Appl Intell (2015) 42:544–565
DOI 10.1007/s10489-014-0610-5

A dissimilarity-based imbalance data classification algorithm

Xueying Zhang · Qinbao Song · Guangtao Wang · Kaiyuan Zhang · Liang He · Xiaolin Jia

Published online: 22 November 2014

Abstract Class imbalances have been reported to compromise the performance of most standard classifiers, such as Naive Bayes, Decision Trees and Neural Networks. Aiming to solve this problem, various solutions have been explored mainly via balancing the skewed class distribution or improving the existing classification algorithms. However, these methods pay more attention on the imbalance distribution, ignoring the discriminative ability of features in the context of class imbalance data. In this perspective, a dissimilarity-based method is proposed to deal with the classification of imbalanced data. Our proposed method first removes the useless and redundant features by feature selection from the given data set; and then, extracts representative instances from the reduced data as prototypes; finally, projects the reduced data into a dissimilarity space by constructing new features, and builds the classification model with data in the dissimilarity space. Extensive experiments over 24 benchmark class imbalance data sets show that, compared with seven other imbalance data tackling solutions, our proposed method greatly improves the performance of imbalance learning, and outperforms the other solutions with all given classificationalgorithms.

Keywords Dissimilarity-based classification · Class imbalance · Software defect prediction · Feature selection · Prototype selection

X. Zhang (✉) · Q. Song (✉) · G. Wang · K. Zhang · L. He · X. Jia
Department of Computer Science and Technology, Xi'an Jiaotong University, 28 Xian-Ning West Road, Xi'an, Shaanxi 710049, People's Republic of China
e-mail: xueyingzhang.725@stu.xjtu.edu.cn
e-mail: qbsong@mail.xjtu.edu.cn

G. Wang
e-mail: gt.wang@stu.xjtu.edu.cn

K. Zhang
e-mail: sxzky@stu.xjtu.edu.cn

L. He
e-mail: lhe@mail.xjtu.edu.cn

X. Jia
e-mail: xlinjia@mail.xjtu.edu.cn

1 Introduction

A data set is imbalanced if the examples of one class outnumbers those of the others. In practice, the imbalance issue is often encountered by numerous real-world machine learning applications, such as text classification [13, 80, 81], speech recognition [46], software defect prediction [32, 34, 35, 61], and bioinformatics and biomedical decision making [48, 76].

The skew class distribution of imbalance data hinders the performance of most standard classification algorithms that work well on data sets with even class distribution, such as Naive Bayes, IB1, C4.5 [30, 74], Logistic Regression [70], Neural Networks and Support Vector Machines [30]. Therefore, the imbalance data problem has attracted much attention of the authoritative machine learning and data mining workshops such as AAAI'2000, ICML'2003 and ACM SIGKDD'2004, and a number of imbalance data dealing with methods have been proposed in the data level and the algorithm level.

The data level solutions concentrate on two points: 1) rebalancing the imbalanced class distribution by sampling or creating new examples, such as random undersampling (RUS) [41], random oversampling (ROS) [42], and synthetic minority over-sampling technique (SMOTE) [8]. It's worth noting that there are still some issues with these solu-

tions, such as some important information may be discarded by undersampling, and duplications and uncertainties introduced by oversampling might lead to overfitting and new problems; 2) feature selection [11, 34, 47, 69, 71, 72, 81] is used to avoid overfitting for the high dimensional imbalance data set. However, feature selection just reduces the dimensionality via removing the unimportant and useless features rather than enhances the discriminant ability of features in essence.

The algorithm level solutions primarily focus on exploring some suitable and robust classification algorithms for dealing with the imbalance learning problems, including one-class learning [64, 70, 75], ensemble learning methods [6, 33, 68], and cost-sensitive analysis [14]. But there are some issues with these methods, for instance, one-class learning fails to build a classifier when the features lack discriminant ability; too much computation time is consumed by ensemble learning methods, and an appropriate cost matrix needs to be determined beforehand by cost-sensitive analysis, etc.

In an imbalance problem, the most obvious characteristic is the skewed class distribution. Nevertheless, theoretical and experimental studies presented in [3, 30, 73] indicate that the skewed data distribution is not the only factor that influences the performance of a traditional classification algorithm in identifying rare events. Simultaneously, small sample size, high dimensionality and the problem complexity will hinder the learning performance as well, because it is difficult to build a good classification model over the high degree of features with limited samples. Essentially, a classification model is built based on the relationship between features and classes. When the imbalance data is high dimensionality and small size, the features are lack of the discriminant ability for classes, and further lead to the degradation in classification performance, specially on the minority class. Nevertheless, the existing solutions pay more attention on re-balancing the skewed class distribution or algorithm adaption but less on studying how to improve the discriminate ability of features in the imbalance data sets.

In order to build a better classification model for imbalance learning problems, it is necessary to construct new features with high discriminant ability instead of original features. Fortunately, Pekalska and Duin [17, 63] have presented a dissimilarity-based representation method which can improve the discriminant ability of features via pairwise dissimilarities between examples, because it not only captures the statistical information but also preserves the structural information of data sets. This dissimilarity-based representation method is originally proposed to depict the characteristics of unstructured or incomplete data sets, recently, it has been successfully applied to describe structural data sets and is capable of building good classifiers [58, 59, 62, 63], especially in pattern recognition. Inspired by these work, we believe that dissimilarity-based representation can be employed for handling the imbalance learning problems as well, although there is no research work on applying this method to solve the imbalance learning problems.

Based on the dissimilarity-based representation, the original data set is projected into the dissimilarity space, in which general classifiers are trained, i.e. the dissimilarity-based classification algorithm (DBC) [59]. The DBC primarily consists of tree parts, they are prototype selection, dissimilarity transformation and classification. Taking into account that those redundancy and useless features (such as noise or irrelevant features) in an imbalance data set might affect the quality and efficiency of prototype selection and dissimilarity transformation of DBC, we proposed an expanded dissimilarity-based classification method (EDBC) for solving the imbalance learning problems, in which feature selection is carried out beforehand to filter those unimportant features out of the original data sets. In our experimental study, three feature selection methods, three prototype selection methods and two distance measures are employed for accomplishing dissimilarity transformation on 24 imbalanced data sets; seven state-of-art solutions (RUS, ROS, SMOTE, Bagging, Boosting, MetaCost and EM1vs1) are compared with our proposed method EDBC in terms of AUC suggested in Refs. [25, 26] under five standard classification algorithms (Naive Bayes, Random Forest, IB1, Multilayer Perceptron and Logistic Regression). The experimental results show that our proposed method EDBC greatly improves the performance of classifiers on the imbalance data sets and stands out in comparison with the other solutions.

The rest of this paper is organized as follows: Section 2 reviews the previous research work related with the conventional DBC method. Section 3 presents the expanded dissimilarity-based imbalance classification method. Section 4 reports the experimental procedure and analyzes the results. Finally, Section 5 summarizes the study and draws the conclusion.

2 Previous work

In the past decades, the issues of the imbalance learning problem have been discussed and reviewed [2, 7, 9, 22, 25, 30, 40]. These methods can be grouped into two categories: in data level and in algorithm level.

The data level methods focus on adjusting the original imbalanced class distribution via sampling or generation of new examples or reducing the dimensionality of the high dimensional imbalance data sets by feature selection. More popular solutions that use various types of sampling

methods were explored to alter the skew class distribution. Random under-sampling [41] randomly deletes some examples of the majority class, meanwhile it also removes some information important for afterwards classification. Random over-sampling [42] replicates the minority class examples. Since over-sampling makes exact copies of the minority class examples, the duplication leads to overfitting. To overcome this problem, synthetic minority over-sampling technique (SMOTE) [8] was proposed to over-sample the minority class by creating new synthetic minority examples. For the high dimensional imbalance data set, to avoid overfitting, feature selection [11, 34, 71, 72, 81] is confirmed to be more important than the choice of the classification algorithm.

The algorithm level methods concentrate on the modification of the existing classification algorithms to suit for dealing with the imbalance learning problems, including cost-sensitive learning, recognition-based learning and ensemble learning. 1) Cost-sensitive learning [45] approaches to minimize the total misclassification cost via adjusting the misclassification cost for each class. MetaCost [14] is one classical algorithm of this kind, which makes an arbitrary classification algorithm cost-sensitive via wrapping a cost-minimizing procedure around. 2) Recognition-based learning [64], Ripper [70, 75] and auto association [29] provide the discrimination model created on the examples of the target class alone. They have been proved to be particularly useful on extremely unbalanced data sets composed of a high dimensional noisy feature space. 3) Ensemble learning is often employed to reduce the variance and bias through summarizing the results of many classification algorithms on the imbalanced data. Representatively, Bagging [6] produces an aggregated predictor to strengthen the classification ability of one base classification algorithm via generating multiple versions of a classification algorithm. Boosting [40] aims to identify the accurate weights for all training examples by iteratively adjusting them according to the classification results. Chawla et al. [10] proposed to combine the SMOTE method with Boosting in order to balance the skewed class distribution and then learn better and broader decision regions for the minority class. Additionally, a novel coding-based multiclass algorithm [68] was presented to convert the imbalanced binary class problem into a balanced multi-class problem and then build a binary classification algorithm on each pair of two classes with the one-against-one coding scheme.

From the solutions mentioned above, we know that the existing solutions pay more attention on adjusting the skewed class distribution or exploring new algorithms but less on improving the discriminant ability of features for imbalance learning. In fact, if the original features are lack of discriminant ability, then it is not powerful enough to only carry out feature selection. Alternatively, replacing the original features, Pekalska et al. [63] proposed that the dissimilarities to a subset of the examples in the historical data are more effective for representing a data set, because it not only reserves the statistical information but also captures the geometry and structure information of the data set. Duin et al. [17] first introduced the relational discriminant analysis method in view of a proximity description of data. After that, Duin and Pekalska [52, 57, 58, 62, 63] demonstrated that dissimilarity-based representation can improve classification performance and provided the framework of the DBC method consisted of prototype selection, dissimilarity transformation and classification. To optimize the DBC method, Kim et al. [37–39] proposed to use prototype reduction schemes for improving the quality of prototype selection. In the recent years, the DBC method has been recognized and widely applied in various fields of real word, such as detecting the seismic signals [49], face recognition [50] and medical image computing [67], etc. However, there is no research work on applying this method for solving the imbalance learning problem. To build a good classifier, we employ the dissimilarity-based method to classify the imbalance data sets.

3 Dissimilarity-based imbalance data classification method

In this section, we first provide an overview of the proposed method in Section 3.1, and then respectively state each step of the proposed expand algorithm in detail, involving feature selection in Section 3.2, prototype selection in Section 3.3 and dissimilarity transformation in Section 3.4.

3.1 Overview of the proposed method

In the traditional way of imbalance learning, the classifiers are built in the original feature space. However, an alternative way is to construct classification models on dissimilarity representations, in which each example is described by pairwise dissimilarity relations between examples in original data sets and the representative examples. This way becomes especially useful when the original data is described by many features or when experts cannot formulate the attributes explicitly, because they are able to provide a dissimilarity measure which can be considered as a connection between perception and higher-level knowledge, being a crucial factor in the process of human recognition and categorization [18, 21].

The original dissimilarity-based classification algorithm (DBC) consists of prototype selection, dissimilarity transformation and classification. Using this method to classify the imbalance data sets, the redundancy and irrelative features in the data sets may compromise the performance of

prototype selection methods and even disturb the dissimilarity transformation, finally results in bad classification performance. Aiming to further alleviate the disturbance of useless features on prototype selection and dissimilarity transformation of the dissimilarity-based classification algorithm, we expand the original DBC method with feature selection as the expanded dissimilarity-based Classification algorithm (EDBC), which consists of two parts: *Model Construction* and *Classification*. Figure 1 shows the details.

1. *Model Construction*

   For a given historical imbalance data set, an important feature subset to the target concept is firstly selected (i.e. *Feature Selection*), and then the data is reduced via reserving these selected features only (i.e. *Data Reduction*). Secondly, the representative examples are selected from the reduced data for each class and a prototype set is obtained (i.e. *Prototype Selection*), and the reduced data is projected into the dissimilarity space via computing the dissimilarity between examples of the reduced data and the prototype set (i. e. *Dissimilarity Transformation*). Finally, the classification model is constructed by a specific classification algorithm on the dissimilarity-based imbalance data set.
2. *Classification*

   For a new imbalanced data set, with the feature subset and the prototype set selected in *Model Construction*, its dimensionality is reduced and the corresponding dissimilarity-based data set is created via computing dissimilarities between examples in prototype set and training data. Finally, the classification model built in the *Model Construction* is employed to classify the dissimilarity-based imbalance data set.

3.2 Feature selection

When learning imbalance problems, some redundancy and useless features existed in the imbalance data sets may hinder the generalization ability of the given learning algorithm [79]. It is inadequate to handle the high dimensional imbalance data sets only via employing the sampling techniques and adapted algorithms, Putten and Someren [69] carried out the experiments on the high dimensional imbalance data sets in the CoIL Challenge 2000 project, and they draw the conclusion that the feature selection is much more important than the selection of the classification algorithm, which contributes to avoiding over fitting.

In the preparation stage, the purpose of employing the feature selection [23] is to filter those irrelative and redundancy features, in order to alleviate and even avoid the curse of dimensionality, reduce storage and memory requirements, increase mining accuracy, and even enhance the comprehensibility of the classification results. According to the evaluation process of feature selection, the feature selection strategy can be divided into three types, they are filter, wrapper and embedded. The wrapper and embedded feature selection methods evaluate the feature subset depending on the performance of the special classification algorithm. Although they can get a valid feature subset supporting for the classification algorithm, they lack of generalization and efficiency due to the high complexity and the strong dependence on the classification algorithm. On the contrary, filter methods are independent to the classification algorithm, they get the feature subset via scoring the correlation, χ^2, the information gain and the symmetrical uncertainty between features with lower complexity and reduce the possibility of over fitting.

Aiming to offset the pernicious effects from the useless features on the afterward prototype selection and dissimilarity transformation process, rather than improve the classification algorithm performance directly, so the filter-based feature selection methods are chosen to expend the original DBC method. For solving the class imbalance learning problems, some popular filter-based metrics have been systemically studied and compared with each other [11, 19, 34, 71, 72, 77, 81], in this paper, we adopt three classical feature selection methods to improve the dissimilarity-based classification algorithm, they are correlation-based Feature Selection (CFS) [24], FCBFS [78] and FAST [66].

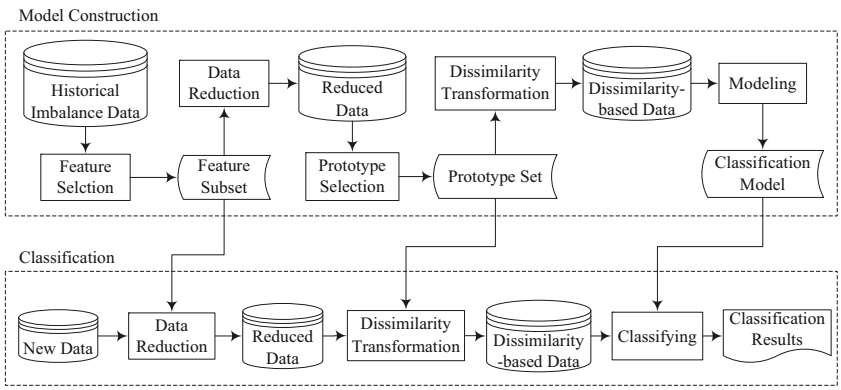


Fig. 1 The framework of the proposed method

CFS [24] evaluates the worth of a subset of attributes by considering the individual predictive ability of each feature along with the degree of redundancy between them. Subsets of features that are highly correlated with the class while having low intercorrelation are preferred.

FCBFS [78] is a fast filter method which can identify relevant features as well as redundancy among relevant features without pairwise correlation analysis, in which the symmetrical uncertainty attribute evaluator is used to evaluate the correlation between features.

FAST [66] is a clustering-based feature selection method with high efficiency and effectiveness. It first adopts the efficient minimum-spanning tree clustering method to divide the features into clusters, and then select the most representative features that are strongly related to the target concept as the feature subset. The clustering-based strategy of FAST has a high probability of producing a subset of useful and independent features.

3.3 Prototype selection

In the DBC method, prototypes, which are the representative examples characterizing the typicality of a class, are selected as the references for dissimilarity transformation. Aiming to get such a prototype set, a number of methods have been investigated, such as the random selection methods *RandomC* [60] and *KCentres* [16], the mode seeking methods *ModeeSeek* [12] and *LinProg* [5], and the feature selection methods *FeaSel* [27], *KCentres-LP* and *EdiCon* [57].

Of these methods, random selection [51, 52, 56, 60] is first proposed and it is the most simple methods for prototype selection, because it just needs to sample randomly with replacement for the specified times (the number of prototypes needs to be extracted). Furthermore, it has been validated to have the capability of working well whenever it is implemented globally or separately for each class. Pekalska, et al. [57] carried an extensive experimental comparisons of some available prototype selection methods in the dissimilarity-based classification algorithm, they suggested that a systematic prototype selection method may be better than random selection and discovered that *KCentres* performs well in general.

Apart from random selection and *KCentres*, the clustering-based Jarvis-Patrick clustering (JPC) algorithm [31, 53] is a good choice to be used for selecting prototypes from the imbalance data set in the proposed EDBC algorithm. JPC is good at dealing with noise, outliers, clusters of different sizes, shapes and densities. High-dimensional data is particularly good at finding tight clusters of strongly related examples. In the JPC algorithm, the shared nearest neighbor (SNN) similarity is used to measure the proximity between examples instead of direct similarity, such as distance measures and correlation coefficient. The SNN similarity is useful since it addresses some problems that occur with direct similarity. At the same time, the SNN similarity takes into account the context of an example via the number of shared nearest neighbors, so it can handle the situation in which an example happens to be relatively close to another example but belongs to a different class.

Algorithm 1: Computing SNN similarity()

```
Find the k-nearest neighbors of all examples.
if two examples, x_i and x_j are not among the k-nearest
   neighbors of each other then
    similarity(x_i, x_j) ← 0
else
    similarity(x_i, x_j) ← number of shared neighbors
end
end
```

The prototypes are diverse with different prototype selection methods. Assume there are i classes in imbalance data set: $\omega_1, \omega_2, ..., \omega_i$. Let D_t be a training set and let D_i be the training data consisted of examples belong to the class ω_i. Each method selects R examples for the prototype set P. The above algorithms are all applied to each class separately, then r_i examples are chosen such that $R = \sum r_i$. The detailed prototype selection procedures are described as follows:

1. *Random Selection* (*RC*). Random selection of r_i examples from the training data set D_i of class ω_i and get the representation set R_i with replacement, the final representation set $R = \sum R_i$.
2. *KCentres* (*KC*). This algorithm is applied to each class ω_i. It tries to choose r_i examples from the class ω_i. The algorithm proceeds as below:

   (1) Randomly select an initial set $R_i = \{p_1^i, p_2^i, ..., p_{ri}^i\}$ consisted of r_i examples from the i^{th} training data set D_i;
   (2) For each $x \in D_i$, find its nearest neighbor in R_i. Let J_j, $j = 1, 2, ..., r_i$, be a subset of D_i consisting of examples that owns the same nearest neighbor p_j^i in R_i;
   (3) For each J_j find its center c_j, that is the example for which the maximum distance to all other examples in J_j is minimum, that is the radius of J_j;
   (4) For each center c_j, if $c_j \neq p_j^i$, then p_j^i is replaced by c_j in R_i. If an replacement is done, then return to (2) step, otherwise stop the iteration. The final representation set R consists of all sets R_i.

3. Jarvis-Patrick clustering (JPC). For each class ω_i, it works as follows:

   (1) Compute the SNN similarity between two points of ω_i with Algorithm 1, and connect those pairs of examples with nonzero SNN similarity, finally the SNN similarity graph is obtained;
   (2) Sparsify the SNN similarity graph via cutting down the links between examples whose SNN similarity is smaller than the given threshold;
   (3) Find the connected components (clusters) of the sparsified SNN similarity graph.
   (4) Select the centers of clusters to create the prototype set R_i for the class ω_i. The final prototype set R consists of all sets R_i.

In the clustering process of KC and JPC, the Euclidean distance [54] is used to measure the proximity between examples as default option. Practically, there are many other proximity measures, such as Jaccard coefficient, cosine similarity, the extended Jaccard coefficient, Dynamic Time Warping (DTW) [4, 65] and Optimal Subsequence Bijection (OSB) [43], etc. Noting that, the type of proximity measure should fit the type data. For many types of dense, continuous data, metric distance measures such as Euclidean distance are often used. For the sparse, asymmetric data, it is more suitable to employ the cosine, Jaccard measures and the extended Jaccard coefficient. DTW and OSB are appropriate to be used for measuring the proximity of examples in time series. Besides, for searching similar web pages, V. Loia, et al. [55] proposed a proximity fuzzy C-means (P-FCM) incorporating a measure of similarity or dissimilarity as user's feedback on the clusters, in which the Euclidean distance is used within the standard FCM algorithm.

Furthermore, Andy and Matthew [44] proposed that the (i,j) element of the proximity matrix produced by Random Forest can used to represent the similarity between examples i and j, they regarded the examples in the same terminal nodes as the similar observations. However, the issues are that some examples may be misclassified into wrong terminal nodes when building Random Forest to be used for each class, and it is difficult to use this method for each class.

Aiming to select the representative examples for each class, the simple and valid proximity measure is preferred. In this paper, the EDBC algorithm is proposed to solve the structural binary imbalance data sets, which are often consisted of dense and continues attributes, so the Euclidean distance is employed to measure the proximity between examples in the cluster-based prototype selection methods KC and JPC.

3.4 Dissimilarity transformation

In the dissimilarity transformation showed in Fig. 2, the reduced data is projected into the dissimilarity space via pairwise dissimilarity relations between examples in the reduced data and the prototype set, where the metric distance measure is often employed to represent the dissimilarity between examples.

Assuming that $D = \{x_1, x_2, ..., x_n\}$ is the reduced data consists of n examples, where $x_i = \{a_{i1}, a_{i2}, ..., a_{im}, c_i\}$ is the ith example with $m + 1$ attributes (m independent

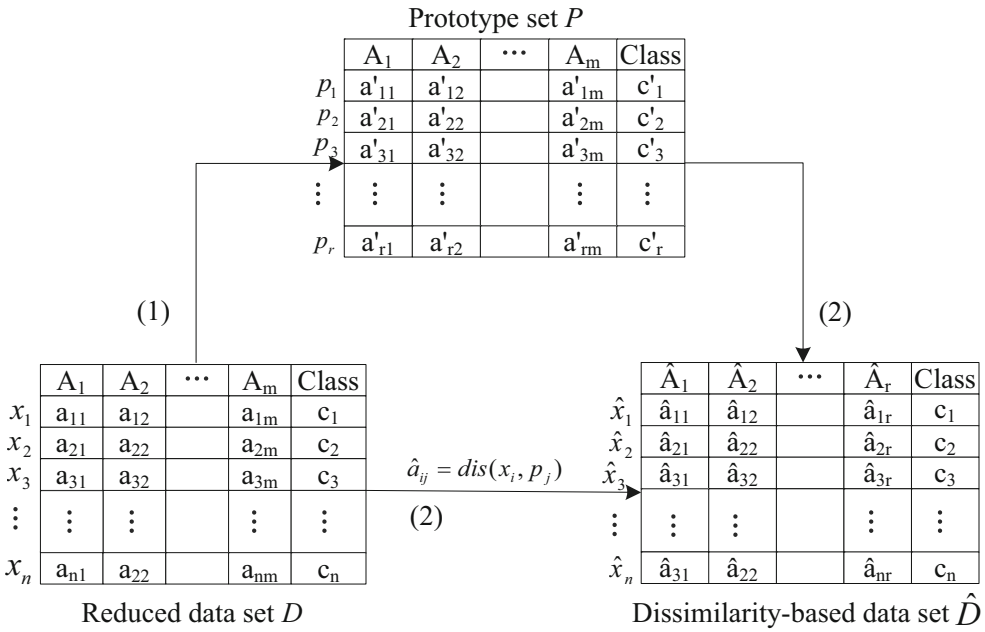


Fig. 2 The procedure of dissimilarity transformation

attributes and the class attribute); $P = \{p_1, p_2, ..., p_r\}$ denotes the prototype set of r representative examples, where $p_i = \{a'_{i1}, a'_{i2}, ..., a'_{im}, c'_i\}$ is the ith prototype selected from the reduced data D. Then the dissimilarity-based data $\hat{D} = \{\hat{x}_1, \hat{x}_2, ..., \hat{x}_n\}$ can be obtained as follows:

1. $\hat{D}$ consists of n examples and each example $\hat{x}_i = \{\hat{a}_{i1}, \hat{a}_{i2}, ..., \hat{a}_{ir}, c_i\}$ is described with r attributes;
2. For the example $\hat{x}_i$, $\hat{a}_{ij}$ is the dissimilarity between the example x_i and the prototype p_j, it is computed by

$$\hat{a}_{ij} = dis(x_i, p_j) = (\sum_{k=1}^{m} |a_{ik} - a'_{jk}|^l)^{\frac{1}{l}}.$$

Where l is a positive integer, it can be 1, 2, ..., ∞.

$l = 1$. Manhattan distance (Hamming distance, City block, taxicab and L_1 norm) .

$l = 2$. Euclidean distance (L_2 norm) are often used for measuring the dissimilarity of dense, continuous data.

$l = \infty$. Supremum (L_{max} or L_∞ norm) distance. This is a the maximum difference between any attribute of the examples.

A distance measure d is called a metric when it fulfills the following conditions:

- Reflectivity: $d(x, x) = 0$.
- Positivity: $d(x, y) > 0$ if x is distinct from y.
- Symmetry: $d(x, y) = d(y, x)$.
- Triangle inequality: $d(x, y) < d(x, z) + d(z, y)$ for every z.

As Pekalska, et al. [60] suggested that, reflectivity and positivity are crucial to define a proper dissimilarity measure. It is unacceptable when a dissimilarity measure is zero for two different objects, since it would violate the compactness hypothesis [1, 15], which states that objects that are similar, are also close in their representations. Besides, it is difficult to interpret the negative dissimilarities between examples. Therefore, in the dissimilarity transformation, the metric Euclidean distance and Manhattan distance based on sums of differences between measurements are adopted to measure the dissimilarity between examples. After dissimilarity transformation, the reduced imbalance data set is projected into the dissimilarity space, in which the classification model is built on the imbalance data sets.

3.5 Complexity of the proposed method

The proposed EDBC consists of feature selection, prototype selection, dissimilarity transformation and classification, thus its complexity depends on the sum of the complexity of the method adopted in each step mentioned above.

Supposing an historical imbalance data set D with n instances and m features, then the complexity of each step in EDBC algorithm is listed as below:

(i) **Complexity of feature selection**

Since the feature selection is employed to improve the quality of prototype selection and dissimilarity rather than effect the classification algorithm directly, the filter-based feature selection methods are selected to expend the dissimilarity-based classification algorithm. The filter-based feature selection aims to remove those redundancy and irrelative features. In the process of feature selection, the computation overhead is taken on evaluating the correlation between each feature and the target feature, and that between features, thus the complexity of feature selection is $T_{FS} = O(m^2)$.

(ii) **Complexity of prototype selection**

When using different methods to select r prototypes, their complexity are diverse. Random selection method just proceeds sampling with replacement for r times and then obtains the r prototypes, its complexity is $T_{RC} = O(r)$. Prototype selection with KCentres method is accomplished by iterative clustering for t times until the r centers become fixed, its complexity is $T_{KC} = O(r \cdot n \cdot t)$. When applying Jarvis-Patrick clustering to select prototypes, its computation overhead is mainly taken on the calculation of the dissimilarity between examples, thus its complexity is $T_{JPC} = O(n^2)$.

(iii) **Complexity of dissimilarity transformation**

In the process of dissimilarity transformation, the primary task is to project the reduced data set into the dissimilarity space via computing the dissimilarity between each examples in the reduced data and r prototypes, so the complexity of dissimilarity transformation is $T_{DT} = O(n \cdot r)$.

Summarizing the complexity of feature selection, prototype selection, dissimilarity transformation and classification, we can achieve the total complexity of the proposed EDBC algorithm $T_{EDBC} = T_{FS} + T_{PS} + T_{DT} + T_C$, that is $T_{EDBC} = O(m^2) + O(r)/O(r \cdot n \cdot t)/O(n^2) + O(n \cdot r) + O(C(n, r))$, in which T_{PS} and T_C respectively denotes the complexity of prototype selection and classification on the reduced data, and $T_C = O(C(n, r))$.

According to the basic principle of each other imbalance solutions, we could deduce that the complexity of RUS is $T_{RUS} = O(C_{min}) + O(C(n', m)), n' = 2C_{min}$, the complexity of ROS is $T_{ROS} = O(C_{max}) + O(C(n', m)), n' = 2C_{max}$, the complexity of SMOTE in which the minority class is over-sampled by creating "synthetic" examples rather than by over-sampling with replacement is

$T_{SMOTE} = O(SMOTE(k, n', t)) + O(C(n', m)), n' = 2C_{max}$, in which k is the number of nearest neighbors in minority class and t denotes the iterations, and the complexity of Bagging, Boosting, MetaCost and EM1vs1 all depends on the iterations of ensemble learning, which can be represented as $T_{Ensemble} = O(t \cdot C(n, m))$.

Since RUS, ROS and SMOTE just proceed the random sampling of majority and minority class with replacement or generation of new minority examples, without further computation, their complexity may be lower than that of the proposed EDBC algorithm. On the contrary, those ensemble learning methods are accomplished via repeated iteration analysis, which are very time-consuming, so their computational complexity will be far higher than that of the proposed EDBC algorithm.

4 Empirical study

In this section, we carry out the extensive experiments on the public imbalance data sets in order to validate the effectiveness of our proposed EDBC algorithm. At first, we introduce the statistical information of the empirical imbalance data sets in Section 4.1. Secondly, we design the contents of experiments so as to comprehensively analyze the performance of EDBC in Section 4.2. Thirdly, we set the available methods for each step of EDBC in detail in Section 4.3. Fourthly, we described the detailed experimental process in Section 4.4. Finally, we analyze and discuss the experimental results corresponding to the investigations proposed in experimental design in Section 4.5.

4.1 Data sets

For the purpose of evaluating the performance and effectiveness of the proposed method EDBC and allowing other researchers to confirm our experimental results, we collect 24 binary imbalanced data sets for classification, in which half of the data sets are available from UC Irvine Machine Learning Repository [20] from different areas (biology, medicine and software engineering), and the rest are software defect prediction data sets in the form of some metrics derived of software source codes.

Table 1 shows the detailed statistical information of these data sets. I, F respectively denotes the number of instances and the number of features. C_{min}, C_{maj} and IR respectively represents the number of examples in the minority class, the number of examples in the majority class and the imbalance ratio with the meaning of how skewed the class distribution of each data sets is, which is calculated from the ratio of the corresponding C_{maj} and C_{min} of each data set and. Aiming to confirm the effectiveness of our proposed EDBC algorithm on the imbalance data sets, the imbalance ratio of 24 empirical data sets we collected are greater than 3.

4.2 Experimental design

In the experiment, two investigations are conducted to evaluate the classification performance of our proposed EDBC algorithm.

(1) Investigation 1: Can the EDBC algorithm improve imbalance learning with conventional classification algorithms?

This investigation aims to explore whether the EDBC algorithm can improve the imbalance learning performance with the conventional classification algorithms and how the improvement on imbalance classification performance with EDBC compared with other imbalance handling methods. Aiming to achieve this purpose, we respectively compare the performance of each given classification algorithm with the proposed EDBC algorithm to that on the original empirical data sets and those with other popular imbalance solutions.

(2) Investigation 2: What and how factors will affect the performance of EDBC as the imbalance ratio alters?

This investigation aims to learn what factors impact the performance of EDBC and how the performance of EDBC will be affected by each factor as the imbalance ratio changes.

In the proposed EDBC algorithm, there are four factors that may affect its classification performance. They are the determinations of feature selection methods, prototype selection methods, number of prototypes and dissimilarity measures. When observing the effect from each factor, it is advisable to compare the differences in classification performance of the EDBC algorithm with various settings of one factor under the condition of keeping other factors fixed. And so on, we can obtain the effect from all factors on EDBC when solving the imbalance learning problems.

4.3 Experimental setup

In this section, we individually setup the methods adopted in each step of EDBC, they are feature selection methods, prototype selection methods and the number of prototypes, the distance measures in dissimilarity transformation and the classification algorithms.

(1) Feature selection methods

In the process of feature selection, three representative feature selection methods were applied to choose

Table 1 Summary of 24 imbalanced binary data sets

ID	Data	I	F	C_{min}	C_{maj}	IR
1	abalone9-18	731	9	42	689	16.4
2	ant1.3	115	64	20	95	4.75
3	ant1.4	163	64	38	125	3.29
4	ant1.5	266	64	32	234	7.31
5	ant1.7	681	64	165	516	3.13
6	camel1.4	720	64	134	586	4.37
7	camel1.6	791	64	170	621	3.65
8	ecoliIM	336	8	77	259	3.36
9	ecoliIMU	336	8	35	301	8.6
10	ecoliOM	336	8	20	316	15.8
11	glassNW	214	10	51	163	3.2
12	hepatitis	155	20	32	123	3.84
13	ivy1.4	209	64	15	194	12.93
14	ivy2.0	294	64	37	257	6.95
15	jedit4.2	344	64	48	296	6.17
16	new-thyroid	215	6	35	180	5.14
17	synapse1.0	139	64	15	124	8.27
18	tomcat6.0	732	64	77	655	8.51
19	vehicleVAN	846	19	199	647	3.25
20	vowelZ	990	14	90	900	10
21	xalan2.4	634	64	108	526	4.87
22	xerces1.2	291	64	43	248	5.77
23	xerces1.3	302	64	65	237	3.65
24	yeastCYT-POX	483	9	20	463	23.15

those important features from the raw training data, they were the classical correlation-based feature selection CFS, the correlation-based filter solution FCBFS and the clustering-based feature subset selection algorithm FAST.

(2) Prototype selection methods

In the process of prototype selection, three prototype selection methods are employed to select the representative examples, including the random sampling for each class (RC), the KCentres (KC) and the Jarvis-Patrick Clustering (JPC).

In the process of prototype selection, the number of prototypes determines the dimensionality of the new imbalance data mapped in dissimilarity space. Too small, a much lower dimensionality will lead to overfitting; too large, it will increase the complexity of computation and introduce some similar prototypes, on the contrary.

As suggested by Pekalska, et. al [57], the number of prototypes should be in the range of $3-10$ % of training data. Meanwhile, considering a trade off between classification performance and computation complexity, we set the numbers of selected prototypes to be $r = LogI$ (I is the number of instances)[36] and $r = 20$ for each training data.

(3) Distance measures

In the process of dissimilarity transformation, Euclidean distance and Manhattan distance are utilized to compute the pairwise dissimilarity between examples in the reduced imbalance data set and the set of prototypes.

(4) Classification algorithms

After the dissimilarity transformation, the classification models are built on the new imbalance data sets in the dissimilarity space. In the dissimilarity classification, five traditional classification algorithms are applied to construct the classification model on each imbalance data set, involving the probability-based Naive Bayes (NB), the instance-based nearest neighbor (IB1) [30, 74], the tree-based Random Forest (RF) [36], the network-based Multilayer Perceptron (MLP) [25, 28, 30] and the linear-based Logistic Regression (LR) [70], which have been implemented in WEKA.

Procedure Experimental Process

```
1  D_s = {D_1, D_2, ..., D_24} - The set of historical binary imbalanced data sets;
2  PS = {RC, KC, JPC} - The set of prototype selection methods;
3  FS = {CFS, FCBFS, FAST} - The set of feature selection methods;
4  Dist = {Euclidean, Manhattan} - The set of dissimilarity measures;
5  R = {LogI, 20} - the size of the representative set (I denotes the size of training data);
6  Classifiers = {Naive Bayes, Random Forest, IB1, Multilayer Perceptron, Logistic Regression} - The set of
   traditional classification algorithms;
7  TIMES = 5 - Times of iteration, FOLDS = 10 - Number of bins;
8  for each C ∈ Classifiers do
9      for each D ∈ D_s do
10         for j = 1 to TIMES do
11             generate FOLDS bins from D;
12             for i = 1 to FOLDS do
13                 for each fs ∈ FS do
14                     test = bins(i);
15                     train = D \ test ;
16                     subset = feasel(train, fs) //feature selection
17                     redTrain = dataRed(train, subset);
18                     redTest = dataRed(test, subset);
19                     D'_{min&maj} = divideClass(train');
20                     for each r ∈ R do
21                         for each ps ∈ PS do
22                             for each dis ∈ Dist do
23                                 P = proSel(D'_{min&maj}, ps, r) //prototype selection
24                                 train^ = DisTrans(redTrain, P, dis) //dissimilarity
                                     transformation
25                                 classifier = learn(C, train^) //classifier construction
26                                 test^ = DisTrans(redTest, P, dis);
27                                 AUC = apply classifier to classify test data test^;
```

4.4 Experimental process

In the experimental process, we employ the EDBC algorithm on each binary imbalanced data set with all combinations generated by three kinds of prototype selection methods, three types of feature selection methods, two classical metric distance measures and five traditional classification algorithms. The details of our experimental process is described in Procedure *Experimental Process*.

Aiming to further validate the effectiveness of the proposed EDBC algorithm, we additionally apply seven popular imbalance handling methods to learning on the empirical data sets and then acquire their performance on each data set for the subsequent comparison with EDBC. The seven imbalance solutions are random under sampling (RUS), random over sampling (ROS), synthetic minority over-sampling technique (SMOTE), Bagging, Boosting, cost-sensitive learning (MetaCost) and EM1vs1.

For evaluating the performance of the proposed EDBC algorithm and other imbalance solutions, the 5×10 cross-validation strategy is realized in the experimental process. For each 10-fold cross-validation, one raw imbalance data was randomly divided into 10 equal folds and EDBC is trained on the rest of the nine folds and tested on the specified fold for each fold. Aiming to obtain reliable and stable classification performance, the 10-fold cross-validation strategy is repeated for 5 times and examples are randomly ordered for each iteration. The average AUC of 50 iterations is used as the measure for evaluating the classification performance of EDBC on the imbalance data sets.

4.5 Experimental results and analysis

In this section, we first present the results of performance comparison between our proposed method EDBC and other existing imbalance solutions in Section 4.5.1, as a response to the question proposed in Investigation 1, and then we analyze the impact of employing different setting for each step on the performance of the proposed EDBC algorithm in Section 4.5.2 with the intend to answer the question raised by Investigation 2.

4.5.1 Performance comparison

For each given classification algorithm, we extensively compared its classification performance with EDBC and that with other seven imbalance data handling methods (RUS, ROS, SMOTE, Bagging, Boosting, MetaCost and EM1vs1) in terms of AUC. Tables 2-6 shows the details of comparison results for Naive Bayes, Random Forest, IB1, Multilayer Perceptron and Logistic Regression, respectively.

Table 2 The AUC values of different imbalance data handling methods with Naive Bayes

Data	EDBC	Bench	RUS	ROS	SMOTE	Bagging	Boosting	MetaCost	EM1vs1
abalone9-18	0.96	0.98	0.76	0.76	0.75	0.76	0.82	0.73	0.73
ant1.3	0.82	0.67	0.69	0.67	0.67	0.71	0.61	0.66	0.73
ant1.4	0.71	0.57	0.53	0.56	0.58	0.58	0.57	0.46	0.58
ant1.5	0.81	0.78	0.77	0.79	0.74	0.79	0.71	0.75	0.78
ant1.7	0.76	0.78	0.77	0.78	0.77	0.77	0.74	0.75	0.77
camel1.4	0.69	0.72	0.71	0.72	0.72	0.72	0.69	0.68	0.71
camel1.6	0.68	0.69	0.68	0.68	0.69	0.68	0.65	0.57	0.68
ecoliIM	0.99	0.88	0.92	0.92	0.93	0.92	0.95	0.90	0.88
ecoliIMU	0.95	0.92	0.94	0.93	0.94	0.94	0.93	0.91	0.91
ecoliOM	1	0.93	0.99	1	1	0.99	0.97	0.96	0.98
glassNW	0.99	0.74	0.93	0.94	0.94	0.95	0.95	0.93	0.92
hepatitis	0.95	0.65	0.87	0.89	0.88	0.89	0.86	0.87	0.85
ivy1.4	0.81	0.57	0.61	0.55	0.54	0.71	0.56	0.77	0.71
ivy2.0	0.80	0.79	0.79	0.78	0.80	0.79	0.72	0.77	0.77
jedit4.2	0.80	0.74	0.74	0.73	0.73	0.74	0.71	0.72	0.74
new-thyroid	1	0.79	1	1	1	1	1	1	0.99
synapse1.0	0.78	0.72	0.73	0.72	0.72	0.79	0.70	0.76	0.80
tomcat6.0	0.79	0.78	0.80	0.78	0.78	0.79	0.72	0.78	0.79
vehicleVAN	0.93	0.99	0.82	0.82	0.82	0.82	0.94	0.76	0.76
vowelZ	0.99	1	0.98	0.98	0.98	0.98	0.97	0.96	0.93
xalan2.4	0.73	0.72	0.72	0.73	0.73	0.73	0.69	0.71	0.72
xerces1.2	0.70	0.68	0.68	0.67	0.68	0.69	0.64	0.66	0.67
xerces1.3	0.71	0.61	0.62	0.62	0.61	0.62	0.63	0.55	0.62
yeastCYT-POX	0.84	0.94	0.84	0.83	0.79	0.83	0.82	0.78	0.80
Average	0.84	0.78	0.79	0.79	0.78	0.80	0.77	0.77	0.78

*Bench - the classification performance without any imbalance handling solution.

Table 2 shows the classification performance of Naive Bayes on each imbalance data set with different imbalance data handling methods. From it we observe that, 4 out of 8 imbalance data handling methods can improve the performance of Naive Bayes, and EDBC performs best. Compared with SMOTE and EM1vs1, the performance of Naive Bayes has been improved by EDBC by 7.69 %; compared with RUS and ROS, the performance of Naive Bayes has been improved by EDBC by 6.33 %; compared with Bagging, the performance of Naive Bayes has been improved by EDBC by 5 %; and compared with Boosting and MetaCost, the performance of Naive Bayes has been improved by EDBC by 9.09 %.

Table 3 shows the classification performance of Random Forest on each empirical imbalance data set with different imbalance handling methods. From this table, we could find that 4 out of 8 imbalance handling methods improved the performance of Random Forest, in which both EDBC and Bagging are the best methods with the same average AUC. Furthermore, comparing the proposed EDBC algorithm with the other six imbalance solutions, the classification performance of Random Forest with EDBC is improved by 3.70 % compared to that with RUS, ROS and MetaCost, by 6.33 % compared to that with Boosting, by 2.44 % compared to SMOTE and by 1.2 % compared to EM1vs1, respectively.

Table 4 reveals the classification performance of IB1 on each imbalance data set with 8 different imbalance handling methods. From it we can observe that, 7 out of 8 imbalance handling methods greatly increased the performance of IB1 on the imbalance data sets, in which the proposed EDBC algorithm ranks the first place with the highest classification performance on average. Relatively, the classification performance of IB1 with EDBC is higher by 18.57 % than that with ROS, by 15.28 % than that with RUS, by 16.90 % than that with SMOTE and MetaCost, by 11.69 % than that with Bagging, by 12.16 % than that with Boosting and by 9.21 % than that with EM1vs1, respectively.

Table 5 discloses the classification performance of Multilayer Perceptron on each imbalance data set with different

Table 3 The AUC values of different imbalance data handling methods with Random Forest

Data	EDBC	Bench	RUS	ROS	SMOTE	Bagging	Boosting	MetaCost	EM1vs1
abalone9-18	0.83	0.78	0.79	0.79	0.80	0.83	0.82	0.81	0.82
ant1.3	0.82	0.69	0.69	0.66	0.69	0.71	0.66	0.68	0.70
ant1.4	0.71	0.64	0.6	0.63	0.65	0.67	0.58	0.62	0.68
ant1.5	0.81	0.73	0.75	0.77	0.77	0.80	0.73	0.76	0.78
ant1.7	0.76	0.78	0.78	0.77	0.79	0.81	0.77	0.78	0.80
camel1.4	0.72	0.72	0.70	0.71	0.74	0.75	0.71	0.72	0.75
camel1.6	0.69	0.72	0.70	0.71	0.71	0.76	0.71	0.72	0.75
ecoliIM	0.99	0.94	0.94	0.95	0.94	0.95	0.95	0.93	0.93
ecoliIMU	0.94	0.90	0.92	0.90	0.91	0.93	0.93	0.91	0.93
ecoliOM	0.98	0.94	0.96	0.94	0.96	0.98	0.95	0.96	0.98
glassNW	0.99	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.96
hepatitis	0.90	0.85	0.86	0.87	0.88	0.91	0.85	0.87	0.84
ivy1.4	0.81	0.76	0.79	0.72	0.78	0.81	0.68	0.75	0.82
ivy2.0	0.80	0.76	0.75	0.74	0.77	0.80	0.72	0.78	0.81
jedit4.2	0.79	0.75	0.76	0.75	0.75	0.80	0.75	0.75	0.79
new-thyroid	1	0.99	0.99	1	0.99	1	0.99	0.99	0.99
synapse1.0	0.78	0.65	0.68	0.68	0.70	0.75	0.69	0.69	0.74
tomcat6.0	0.79	0.78	0.77	0.77	0.79	0.82	0.74	0.78	0.81
vehicleVAN	0.96	0.99	0.99	0.99	0.99	1	0.99	0.99	0.98
vowelZ	1	1	1	1	1	1	0.94	1	1
xalan2.4	0.73	0.72	0.71	0.71	0.72	0.74	0.70	0.72	0.75
xerces1.2	0.71	0.71	0.67	0.70	0.72	0.72	0.62	0.74	0.70
xerces1.3	0.72	0.73	0.75	0.74	0.75	0.77	0.73	0.73	0.75
yeastCYT-POX	0.89	0.82	0.84	0.84	0.82	0.93	0.85	0.84	0.88
Average	0.84	0.81	0.81	0.81	0.82	0.84	0.79	0.81	0.83

imbalance solutions. From the table we observe that, all the imbalance data handling methods can improve the classification performance of Multilayer Perceptron on the imbalance data sets, among which our proposed EDBC algorithm ranks the first with the highest AUC. Individually, the classification performance of Multilayer Perceptron with EDBC is increased by 7.5 % compared with RUS, Boosting and EM1vs1, by 8.86 % compared with ROS and MetaCost, by 10.26 % compared to SMOTE and by 4.88 % compared to Bagging.

Table 6 uncovers the classification performance of Logistic Regression with 8 different imbalance learning handling methods. From it we observe that, 6 out of 8 imbalance data handling methods increased the performance of Logistic Regression, where EDBC ranks the first again with the best classification performance. By comparison, the classification performance of Logistic Regression with EDBC has been improved by 13.16 % for RUS and ROS, by 11.69 % for SMOTE, Boosting and MetaCost, by 8.86 % for Bagging and by 10.26 % for EM1vs1, respectively.

To sum up, our proposed EDBC algorithm has a capability of building a more effective classification model when solving the imbalance learning problems. Besides, compared with other seven imbalance data handling methods, it is always the most outstanding of all.

For the purpose of validating whether our proposed EDBC algorithm significantly outperforms the other solutions with all classification algorithms, the Wilcoxon signed-rank test was conducted under the significance level 0.05. The alternative hypotheses are that for each classification algorithm, the EDBC method is superior to the compared methods.

The p-values of the hypotheses are all significantly lower than 0.05 except Random Forest. This means that the EDBC algorithm is significantly superior to the other methods with four classification algorithms Naive Bayes, IB1, Multilayer Perceptron and Logistic Regression. When using Random Forest as the classification algorithm, both of EDBC and Bagging are the best imbalance solutions and there is no significant differences between them.

Table 4 The AUC values of different imbalance data handling methods with IB1

Data	EDBC	Bench	RUS	ROS	SMOTE	Bagging	Boosting	MetaCost	EM1vs1
abalone9-18	0.76	0.57	0.66	0.57	0.65	0.68	0.71	0.57	0.75
ant1.3	0.82	0.65	0.63	0.66	0.66	0.72	0.73	0.65	0.71
ant1.4	0.71	0.57	0.54	0.57	0.57	0.54	0.54	0.60	0.54
ant1.5	0.81	0.62	0.63	0.61	0.62	0.70	0.63	0.65	0.67
ant1.7	0.75	0.65	0.66	0.64	0.64	0.69	0.67	0.65	0.69
camel1.4	0.69	0.57	0.58	0.57	0.58	0.61	0.59	0.58	0.62
camel1.6	0.68	0.57	0.59	0.57	0.59	0.64	0.64	0.56	0.65
ecoliIM	0.97	0.82	0.85	0.83	0.84	0.90	0.89	0.84	0.91
ecoliIMU	0.95	0.75	0.83	0.74	0.81	0.89	0.84	0.82	0.92
ecoliOM	0.97	0.88	0.91	0.88	0.91	0.97	0.98	0.88	0.99
glassNW	0.96	0.93	0.94	0.93	0.94	0.97	0.97	0.93	0.96
hepatitis	0.91	0.71	0.76	0.71	0.74	0.77	0.72	0.74	0.76
ivy1.4	0.81	0.52	0.65	0.52	0.54	0.71	0.67	0.53	0.73
ivy2.0	0.73	0.60	0.62	0.61	0.62	0.65	0.65	0.59	0.69
jedit4.2	0.79	0.62	0.61	0.62	0.63	0.65	0.67	0.63	0.68
new-thyroid	1	0.99	0.99	0.99	0.99	1	0.99	0.99	0.99
synapse1.0	0.78	0.49	0.54	0.49	0.50	0.57	0.53	0.59	0.58
tomcat6.0	0.79	0.65	0.66	0.65	0.67	0.69	0.67	0.66	0.75
vehicleVAN	0.96	0.92	0.93	0.92	0.92	0.96	0.97	0.92	0.96
vowelZ	1	1	0.99	1	1	1	1	1	1
xalan2.4	0.73	0.59	0.61	0.59	0.60	0.65	0.62	0.59	0.66
xerces1.2	0.71	0.61	0.57	0.61	0.60	0.60	0.58	0.61	0.61
xerces1.3	0.69	0.65	0.65	0.65	0.64	0.69	0.68	0.66	0.67
yeastCYT-POX	0.88	0.76	0.79	0.76	0.8	0.85	0.87	0.80	0.84
Average	0.83	0.70	0.72	0.70	0.71	0.75	0.74	0.71	0.76

To provide an overall performance evaluation of all given imbalance handling methods, we rank these solutions for each classification algorithm according to the average AUC of 24 imbalance data sets separately, the best performing algorithm getting the rank of 1, the second best rank 2..., as shown in Table 7. In case of ties, the same rank is assigned like both of RUS and ROS ranking 3rd, Bench, SMOTE and EM1vs1 all ranking 5th. From this table we observe that our method EDBC ranks the first place with the minimum average rank. It means that the proposed EDBC method markedly improves the performance of the standard classification algorithms on the imbalance learning problems and surpasses all the other existing solutions.

4.5.2 Sensitivity analysis

As well known that, the kernel of EDBC algorithm consists of feature selection, prototype selection, dissimilarity transformation, so the the methods employed in feature selection and prototype selection, the number of prototypes and the dissimilarity measure adopted in dissimilarity transformation are the crucial influence factors for the classification performance of EDBC.

In this section, we respectively analyze and discuss the effect from each factor on the proposed EDBC algorithm. The details of each factor are illustrated in Figs. 3, 4, 5, and 6, where X-axis denotes the imbalance ratio of each empirical data set and Y-axis represents the classification performance AUC of EDBC with one specified factor.

1 **Sensitivity Analysis of Feature Selection Methods**.

Figure 3 shows the effect from four features selection strategies on the classification performance of the proposed EDBC algorithm for each given classification algorithm as the imbalance ratio altering. From the figure we observe that:

(i) For each given classification algorithm, the classification performance of EDBC is significantly affected by different feature selection methods.

Table 5 The AUC values of different imbalance data handling methods with Multilayer Perceptron

Data	EDBC	Bench	RUS	ROS	SMOTE	Bagging	Boosting	MetaCost	EM1vs1
abalone9-18	0.87	0.89	0.89	0.86	0.83	0.90	0.87	0.90	0.89
ant1.3	0.82	0.56	0.64	0.56	0.57	0.67	0.65	0.67	0.65
ant1.4	0.71	0.56	0.57	0.61	0.6	0.63	0.58	0.61	0.61
ant1.5	0.81	0.75	0.75	0.73	0.75	0.77	0.75	0.75	0.75
ant1.7	0.78	0.75	0.75	0.75	0.75	0.77	0.76	0.76	0.75
camel1.4	0.74	0.67	0.67	0.65	0.66	0.70	0.68	0.69	0.68
camel1.6	0.69	0.66	0.67	0.67	0.66	0.72	0.68	0.68	0.69
ecoliIM	0.99	0.96	0.95	0.95	0.95	0.96	0.94	0.95	0.93
ecoliIMU	0.99	0.94	0.93	0.94	0.91	0.94	0.92	0.93	0.93
ecoliOM	1	0.99	0.98	0.97	0.98	0.98	0.97	0.99	0.99
glassNW	0.99	0.96	0.96	0.95	0.95	0.97	0.97	0.95	0.96
hepatitis	0.94	0.79	0.85	0.83	0.85	0.89	0.84	0.85	0.82
ivy1.4	0.85	0.54	0.72	0.75	0.72	0.84	0.75	0.55	0.80
ivy2.0	0.84	0.65	0.74	0.72	0.72	0.78	0.74	0.73	0.76
jedit4.2	0.79	0.71	0.70	0.72	0.72	0.75	0.73	0.72	0.73
new-thyroid	1	1	1	1	1	1	0.99	1	1
synapse1.0	0.78	0.61	0.65	0.63	0.62	0.68	0.66	0.68	0.69
tomcat6.0	0.83	0.74	0.77	0.76	0.75	0.81	0.74	0.77	0.79
vehicleVAN	0.99	1	1	1	1	1	1	1	0.99
vowelZ	1	0.99	0.99	1	1	1	1	0.98	1
xalan2.4	0.76	0.67	0.70	0.68	0.70	0.74	0.70	0.70	0.72
xerces1.2	0.71	0.66	0.75	0.62	0.57	0.65	0.62	0.62	0.62
xerces1.3	0.72	0.68	0.71	0.66	0.69	0.70	0.70	0.69	0.68
yeastCYT-POX	0.99	0.85	0.78	0.88	0.81	0.85	0.86	0.85	0.76
Average	0.86	0.77	0.80	0.79	0.78	0.82	0.80	0.79	0.80

Generally, FAST performs best of all feature selection methods, while there is no significant improvement and even a little degradation on the classification performance of EDBC using CFS and FCBFS for feature selection. It means that the fast clustering-based feature subset selection method (FAST) is able to find a feature subset that is most relative to the class concept from each imbalance data set, which is conducive to improve the afterward prototype selection and dissimilarity transformation.

(ii) From the fluctuation trend of the classification performance of EDBC with different feature selection methods, we find that the effect of feature selection on EDBC is very significant when $IR < 16$ and it is getting stable and weaker as *IR* increases. Although, FAST significantly outperforms other feature selection methods within EDBC for all given classification algorithms when $IR < 16$, the differences in the performance of EDBC with all feature selection methods also become not so obvious as *IR* creases. That means it is difficult to select those salient features from the extremely imbalance data set with the given feature selection methods.

(iii) Compared to the situation without feature selection (nonFS), the classification performance of EDBC using FAST for feature selection is increased by 11 % for Naive Bayes, 12 % for Random Forest, 23 % for IB1, 6 % for Multilayer Perceptron and 7 % for Logistic Regression, respectively.

From above analysis on the effect from different feature selection methods, we have summarized that feature selection plays a critical role in the proposed method EDBC, especially the FAST method. Additionally, it is necessary to adopt a more valid feature selection method for the the extremely imbalanced data sets.

2 **Sensitivity Analysis of Prototype Selection Methods**.

Figure 4 displays the effect from three prototype selection methods RC, KC and JPC within EDBC on

Table 6 The AUC values of different imbalance data handling methods with Logistic Regression

Data	EDBC	Bench	RUS	ROS	SMOTE	Bagging	Boosting	MetaCost	EM1vs1
abalone9-18	0.89	0.94	0.93	0.95	0.93	0.94	0.84	0.93	0.93
ant1.3	0.82	0.53	0.58	0.51	0.55	0.56	0.63	0.55	0.54
ant1.4	0.71	0.51	0.56	0.50	0.51	0.56	0.57	0.59	0.61
ant1.5	0.81	0.63	0.67	0.62	0.68	0.75	0.70	0.68	0.71
ant1.7	0.79	0.77	0.74	0.76	0.77	0.78	0.75	0.76	0.76
camel1.4	0.76	0.71	0.69	0.71	0.71	0.72	0.67	0.68	0.69
camel1.6	0.72	0.67	0.67	0.67	0.68	0.69	0.67	0.65	0.67
ecoliIM	0.99	0.95	0.94	0.95	0.95	0.95	0.87	0.93	0.90
ecoliIMU	0.96	0.92	0.90	0.91	0.92	0.91	0.88	0.91	0.90
ecoliOM	0.99	0.96	0.96	0.95	0.95	0.97	0.94	0.97	0.99
glassNW	0.99	0.96	0.95	0.96	0.96	0.97	0.97	0.95	0.94
hepatitis	0.94	0.88	0.81	0.88	0.85	0.85	0.87	0.80	0.82
ivy1.4	0.81	0.62	0.67	0.56	0.61	0.83	0.75	0.75	0.82
ivy2.0	0.83	0.64	0.62	0.65	0.64	0.71	0.69	0.64	0.69
jedit4.2	0.80	0.64	0.57	0.63	0.67	0.65	0.66	0.64	0.63
new-thyroid	1	1	1	1	1	1	0.97	1	1
synapse1.0	0.78	0.57	0.59	0.58	0.59	0.59	0.63	0.64	0.67
tomcat6.0	0.83	0.70	0.65	0.69	0.74	0.73	0.69	0.72	0.67
vehicleVAN	0.99	0.99	0.99	0.99	1	1	0.99	0.99	0.99
vowelZ	1	1	0.99	1	1	1	1	0.99	1
xalan2.4	0.76	0.72	0.70	0.72	0.72	0.73	0.68	0.72	0.68
xerces1.2	0.71	0.52	0.58	0.52	0.54	0.61	0.64	0.56	0.63
xerces1.3	0.72	0.57	0.61	0.57	0.56	0.65	0.70	0.59	0.65
yeastCYT-POX	0.99	0.86	0.83	0.86	0.86	0.84	0.83	0.84	0.79
Average	0.86	0.76	0.76	0.76	0.77	0.79	0.77	0.77	0.78

the classification performance of EDBC as the imbalance increases. From the figure we observe that:

(i) The classification performance of EDBC is significantly affected by different prototype selection methods, in which RC and JPC within EDBC achieve the similar classification performance as imbalance increases and both of them are superior to KC on all imbalance data sets.

(ii) When $IR < 16$, the performance of EDBC with RC and JPC fluctuates widely and tends towards stability as IR increases. On the contrary, the performance of EDBC with KC is relatively stable and worse overall.

Table 7 Rank of all imbalance handling methods with different classification algorithms

Classification algorithm	EDBC	Bench	RUS	ROS	SMOTE	Bagging	Boosting	MetaCost	EM1vs1
Naive Bayes	1	5	3	3	5	2	8	8	5
Random Forest	1	5	5	5	4	1	9	5	3
IB1	1	8	5	8	6	3	4	6	2
Multi-Layer	1	9	3	6	8	2	3	6	3
Logistic Regression	1	7	7	7	4	2	4	4	3
Average Rank	1	6.8	4.6	5.8	5.4	2	5.6	5.8	3.2

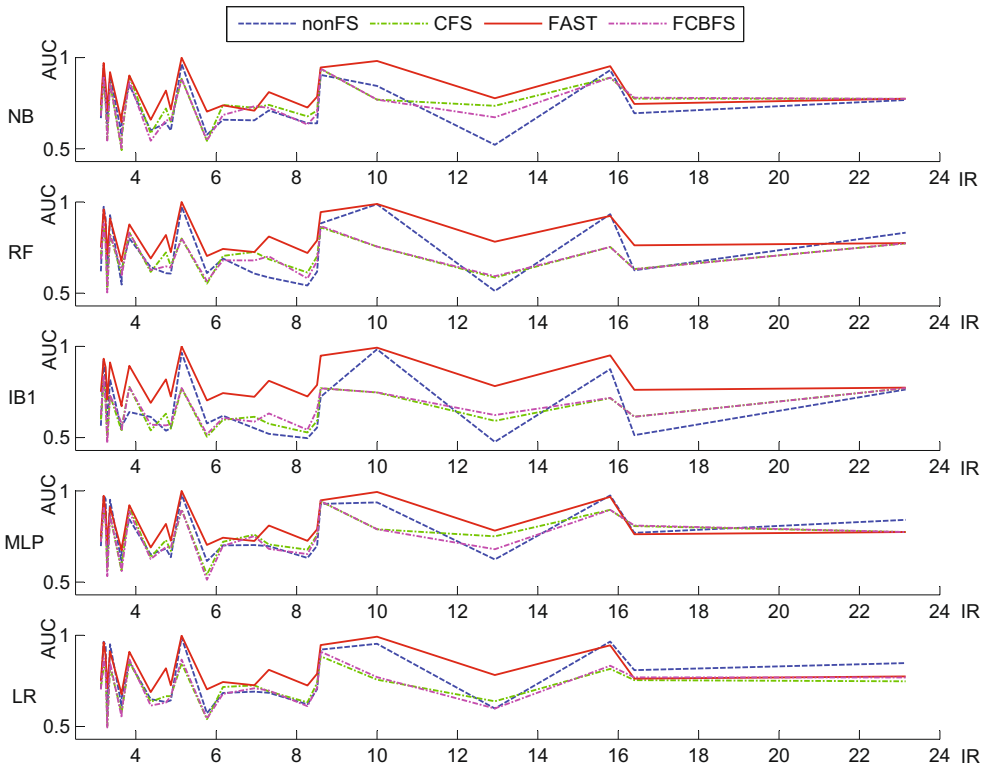


Fig. 3 The sensitivity analysis of four feature selection methods

(iii) Compared to KC, the classification performance of EDBC with RC and JPC is averagely increased by about 12 % for Naive Bayes, 11 % for Random Forest, 15 % for IB1 and Logistic Regression and 14 % for Multilayer Perceptron, respectively.

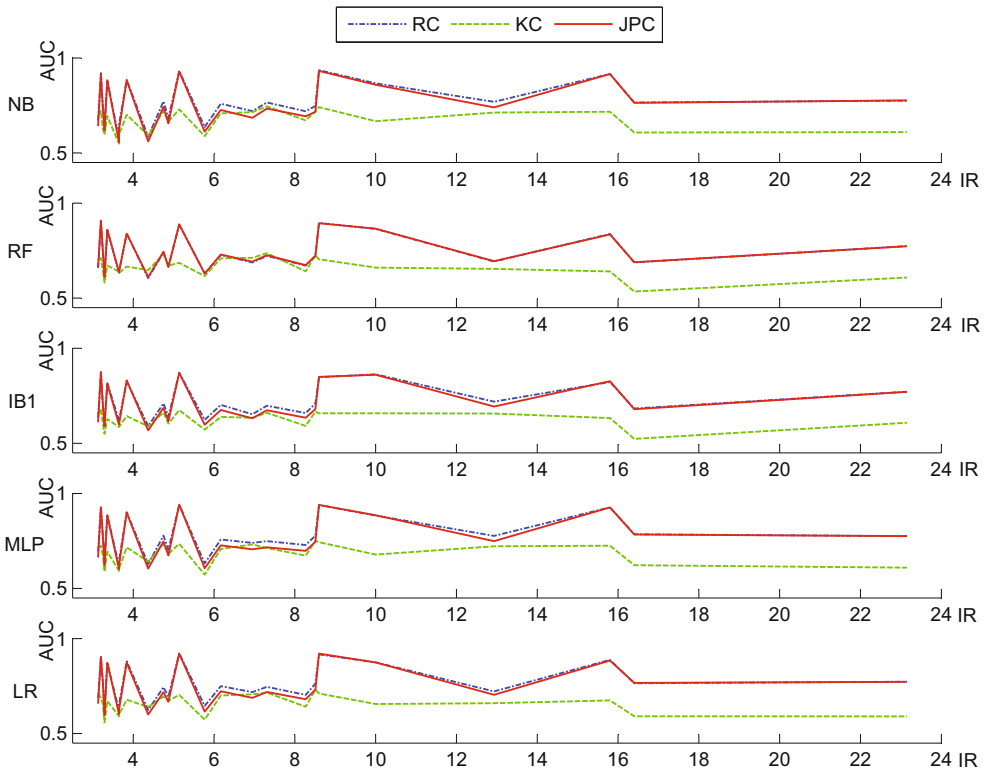


Fig. 4 The sensitivity analysis of three prototype selection methods

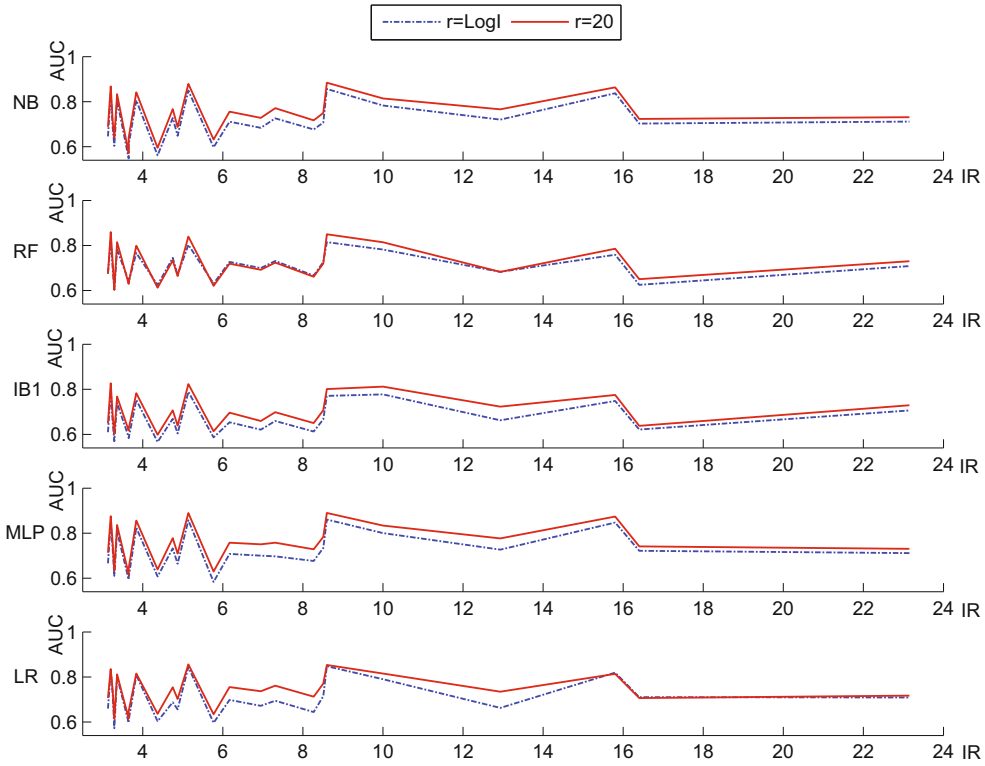


Fig. 5 The sensitivity analysis of two numbers of prototypes

In summary, KC is not suitable for clustering on imbalance data sets, because KC cannot handle non-globular clusters or clusters of different sizes and densities and the data sets containing outliers method, which is the nature of the imbalance data set. While RC proceeds the stratified sampling as the class

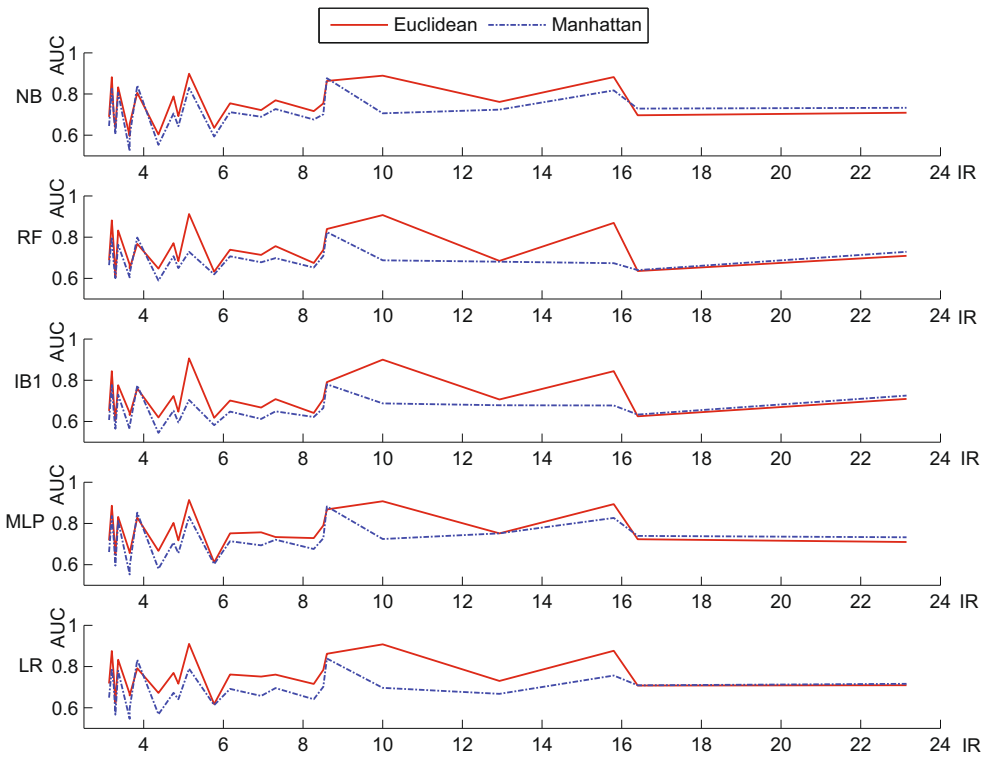


Fig. 6 The sensitivity analysis of two dissimilarity measures

distribution and JPC can compensate the disadvantages of KC, so RC and JPC are very appropriate for selecting the representative examples from imbalance data sets. Furthermore, the advantages of RC and JPC become more and more obvious as *IR* increases.

3 **Sensitivity Analysis of the Numbers of Prototypes**.

Figure 5 shows the impact of the number of prototypes $r = LogI$ and $r = 20$ on the classification performance of EDBC for different classification algorithms. From it we find that, i) the classification performance of EDBC with different numbers of prototypes has the similarity trend as *IR* increases; ii) When $r = 20$, the classification performance of EDBC is averagely higher than that when $r = LogI$ by 5 % for all classification algorithms but Random Forest and iii) when $IR > 16$, the difference between two conditions does not become significant.

It means that the classification performance of EDBC will be degraded with only a few prototypes, because it is difficult to distinguish the examples projected into the dissimilarity space referring a few poor prototypes. Additionally, it also implies that EDBC is not so sensitive to the number of prototypes as *IR* increases. In the practice, taking the computational complexity and classification performance into account, only if the number of prototypes lies in the rational range 3-10% of the data size as suggested by Pekalska, et. al [57], the EDBC algorithm will obtain a stable and approving classification performance.

4 **Sensitivity Analysis of Dissimilarity Measures**

Figure 6 illustrates the effect from two distance measures (Euclidean distance and Manhattan distance) on the classification performance of the proposed EDBC algorithm. From the figure we observe that:

(i) The classification performance of EDBC with Euclidean distance as the dissimilarity measure is superior to that with Manhattan distance. It means that Euclidean distance reflects the real distance between two points, and even profits the construction of classification model in the dissimilarity space. While Manhattan distance is the sum of the lengths of the projections of the line segment between two points, which maybe lead to overlapping of multiple points in the dissimilarity space, e.g. there are many pathes with the same Manhattan distance between two points. It is difficult for EDBC to distinguish those overlapped points in the dissimilarity space via pairwise Manhattan distance.

(ii) The classification performance of EDBC with Euclidean is more volatile than that with Manhattan distance when $IR < 16$, and it tends to stabilize as *IR* increases. Meanwhile the proposed method EDBC with Euclidean distance outperforms that with Manhattan distance by 6 % for Naive Bayes, 8 % for Random Forest, 9 % for IB1, 7 % for Multilayer Perceptron and 10 % for Logistic Regression on average, respectively. As IR increases, the differences between the performance of EDBC with both dissimilarity measures become not so obvious. It implies that the effect from dissimilarity measure on the classification performance of EDBC dies away due to the highly imbalanced class distribution.

Generally, Euclidean distance is more suitable to be used for dissimilarity transformation for EDBC than Manhattan distance.

5 Conclusions

The imbalance learning problem is one challenge of data mining domain. Skewed class distribution often degrades the performance of most traditional classification algorithm, such as Naive Bayes, IB1, C4.5, Logistic Regression, Neural Networks and Support Vector Machines, etc. A number of solutions have been proposed for improving the imbalance classification performance, but they paid more attention on how to rebalance the skewed class distribution and how to find a more suitable classification algorithm, ignoring how to improve the discriminant ability of features in order to increase the performance of traditional classification algorithms on imbalance data sets, essentially.

In this perspective, we have proposed an expanded dissimilarity-based classification algorithm (EDBC) for classifying the imbalance data sets, which proceeds as below:

1. Remove the useless and redundancy features from the original imbalance data via feature selection, aiming to mitigate the effect on the quality of afterward prototype selection and dissimilarity transformation;
2. Select some representative examples for each class from the reduced data, and then produce a prototype set;
3. Project the reduced data into the dissimilarity space through computing the dissimilarity between examples in the reduced data and the prototype set;
4. Construct the classifier on the new data set in the dissimilarity space.

Different from the existing imbalance data handling methods, the proposed EDBC resolves the imbalance learning problems fundamentally via enhancing the discriminant

ability of features with the help of the dissimilarity-based representation. For the purpose of confirming the effectiveness and the efficiency of the proposed EDBC algorithm, we have carried out an extensive empirical study on 24 benchmark imbalance data set, evaluated the performance of EDBC algorithm in terms of AUC and compared it with other commonly used imbalance solutions, including Random under-sampling, Random over-sampling, SMOTE, Bagging, Boosting, MetaCost and EM1vs1 with five traditional classification algorithms Naive Bayes, Random Forest, IB1, Multi-Layer Perceptrons and Logistic Regression over 24 imbalanced data sets.

Firstly, we have compared the average performance of each given classification algorithm with the proposed EDBC algorithm and that with each other imbalance handling method. The classification performance on the imbalance problems has been increased by EDBC by $5 - 9.09$ % for Naive Bayes, $1.2 - 6.33$ % for Random Forest, $9.21 - 18.57$ % for IB1, 4.88-10.26 % Multilayer Perceptron and $8.86 - 13.16$ % for Logistic Regression, respectively. The comparison results shows that our proposed EDBC can greatly increase the performance of all given classification algorithms and outperform other imbalance solutions, overall.

Secondly, we also have analyzed and discussed the sensitivity from the determination of methods employed at each step for our proposed EDBC algorithm, they are feature selection methods, prototype selection methods, the number of prototypes and dissimilarity measures. Through comparative analysis on the classification performance of EDBC on the imbalance data sets, we have achieved a summary for EDBC that FAST is the best choice for feature selection, random selection and Jarvis-Patrick clustering algorithm are more effective for prototype selection with a rational number of prototypes ($3 - 10$ % of training examples) and Euclidean distance is more suitable for measuring the dissimilarity between examples in dissimilarity transformation.

Moreover, we have supplied the computational complexity of the proposed EDBC algorithm and compared it with other imbalance handling method. The comparison results show us that the complexity of the proposed EDBC algorithm may be a little higher than that of RUS and ROS, but it is significantly lower than that of each ensemble learning method. It means that the proposed expanded dissimilarity-based classification method can improve the imbalance learning performance effectively and efficiently.

Acknowledgments This work is supported by the National Natural Science Foundation of China under Grants 61373046 and 61210004.

References

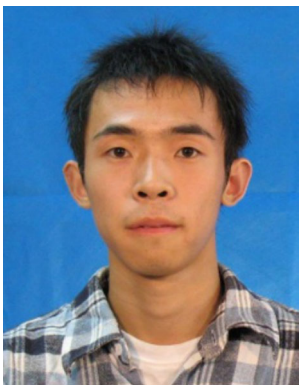
1. Arkadev AG, Braverman ĖM (1967) Computers and pattern recognition. Thompson Book Co, Washington D.C.
2. Barandela R, Sánchez JS, Garcıa V, Rangel E (2003) Strategies for learning in class imbalance problems. Pattern Recog 36(3):849–851
3. Batista GE, Prati RC, Monard MC (2004) A study of the behavior of several methods for balancing machine learning training data. ACM Sigkdd Explorations Newsletter 6(1):20–29
4. Berndt DJ, Clifford J (1994) Using dynamic time warping to find patterns in time series In: KDD workshop, vol 10. Seattle, WA, pp 359–370
5. Bradley PS, Mangasarian OL, Street W (1998) Feature selection via mathematical programming. INFORMS J Comput 10:209–217
6. Breiman L (1996) Bagging predictors. Mach learn 24(2):123–140
7. Chawla NV (2005) Data mining for imbalanced datasets: An overview. In: Data mining and knowledge discovery handbook. Springer, New York, pp 853–867
8. Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP (2002) Smote: Synthetic minority over-sampling technique. J Artif Intell Res 16:341–378
9. Chawla NV, Japkowicz N, Kotcz A (2004) Editorial special issue on learning from imbalanced data sets. ACM Sigkdd Explorations Newsletter 6(1):1–6
10. Chawla NV, Lazarevic A, Hall LO, Bowyer KW (2003) Smoteboost: Improving prediction of the minority class in boosting. In: Knowledge Discovery in Databases: PKDD 2003. Springer, New York, pp 107–119
11. Chen XW, Wasikowski M (2008) Fast A roc-based feature selection metric for small samples and imbalanced data classification problems. In: Proceedings of the 14th ACM SIGKDD international conference on Knowledge very and data mining, pp. 124–132. ACM
12. Cheng Y (1995) Mean shift, mode seeking, and clustering. IEEE Trans Pattern Anal Mach Intel 17(8):790–799
13. Del Castillo MD, Serrano JI (2004) A multistrategy approach for digital text categorization from imbalanced documents. ACM SIGKDD Explorations Newsletter 6(1):70–79
14. Domingos P (1999) Metacost a general method for making classifiers cost-sensitive. In: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining ACM, pp 155–164
15. Duin R (1999) Compactness and complexity of pattern recognition problems. In: International Symposium on Pattern Recognition In Memoriam Pierre Devijver, pp 124–128
16. Duin R, Juszczak P, Paclik P, Pekalska E, De Ridder D, Tax D, Verzakov S (2000) A matlab toolbox for pattern recognition. PRTools version 3
17. Duin R, Pekalska E, Ridder D (1999) Relational discriminant analysis. Pattern Recog Lett 20(11):1175–1181
18. Edelman S (1999) Representation and recognition in vision. MIT press
19. Forman G (2003) An extensive empirical study of feature selection metrics for text classification. J Mach Learn Res 3:1289–1305
20. Frank A, Asuncion A (2010) Uci machine learning repository irvine, ca: University of california. School of Information and Computer Science, vol 213. http://archive.ics.uci.edu/ml
21. Goldstone RL, Son JY (2005) Similarity. Cambridge University Press
22. Guo X, Yin Y, Dong C, Yang G, Zhou G (2008) On the class imbalance problem. In: Fourth International Conference on Natural Computation vol 4 IEEE, pp 192–201
23. Guyon I, Elisseeff A (2003) An introduction to variable and feature selection. The J Mach Learn Res 3:1157–1182

24. Hall MA (1999) Correlation-based feature selection for machine learning. Ph.D. thesis, The University of Waikato
25. He H, Garcia EA (2009) Learning from imbalanced data. IEEE Trans Knowl Data Eng 21(9):1263–1284
26. Huang J, Ling CX (2005) Using auc and accuracy in evaluating learning algorithms. IEEE Trans Knowl Data Eng 17(3):299–310
27. Jain A, Zongker D (1997) Feature selection: Evaluation, application, and small sample performance. IEEE Trans Pattern Anal Mach Int 19(2):153–158
28. Japkowicz N (2000) Learning from imbalanced data sets: a comparison of various strategies. In: AAAI workshop on sets, learning from imbalanced data vol. 68. CA, Menlo Park
29. Japkowicz N (2001) Supervised versus unsupervised binary-learning by feedforward neural networks. Mach Learn 42(1-2):97–122
30. Japkowicz N, Stephen S (2002) The class imbalance problem: A systematic study. Int Data Anal 6(5):429–449
31. Jarvis RA, Patrick EA (1973) Clustering using a similarity measure based on shared near neighbors. IEEE Trans Comput 100(11):1025–1034
32. Jiang Y, Cukic B, Ma Y (2008) Techniques for evaluating fault prediction models. Emp Software Eng 13(5):561–595
33. Joshi MV, Kumar V, Agarwal RC (2001) Evaluating boosting algorithms to classify rare classes: Comparison and improvements. In: Proceedings IEEE International Conference on Data Mining, pp 257–264
34. Khoshgoftaar TM, Gao K (2009) Feature selection with imbalanced data for software defect prediction. In: International Conference on Machine Learning and Applications, IEEE, pp 235–240
35. Khoshgoftaar TM, Gao K, Seliya N (2010) Attribute selection and imbalanced data: Problems in software defect prediction. In: International Conference on Tools with Artificial Intelligence, vol. 1 IEEE, pp 137–144
36. Khoshgoftaar TM, Golawala M, Van Hulse J (2007) An empirical study of learning from imbalanced data using random forest. In: IEEE International Conference on Tools with Artificial Intelligence, vol. 2 IEEE, pp 310–317
37. Kim S, Oommen B (2007) On using prototype reduction schemes to optimize dissimilarity-based classification. Pattern Recog 40(11):2946–2957
38. Kim SW, Gao J (2008) On using dimensionality reduction schemes to optimize dissimilarity-based classifiers. In: Progress in Pattern Recognition, Image Analysis and Applications. Springer, pp 309–316
39. Kim SW, Oommen BJ (2006) On optimizing dissimilarity-based classification using prototype reduction schemes. In: Image Analysis and Recognition. Springer, New York, pp 15–28
40. Kotsiantis S, Kanellopoulos D, Pintelas P (2006) Handling imbalanced datasets: A review. GESTS International. Trans Comput Sci Eng 30(1):25–36
41. Kotsiantis S, Pintelas P (2003) Mixture of expert agents for handling imbalanced data sets. Annals of Mathematics, Computing Teleinformatics 1(1):46–55
42. Kubat M, Matwin S (1997) Addressing the curse of imbalanced training sets: one-sided selection. In: ICML, vol. 97, pp 179–186
43. Latecki LJ, Wang Q, Koknar-Tezel S, Megalooikonomou V (2007) Optimal subsequence bijection. In: Seventh IEEE International Conference on Data Mining, IEEE, pp 565–570
44. Liaw A, Wiener M (2002) Classification and regression by randomforest. Rnews 2(3):18–22
45. Liu XY, Zhou ZH (2006) The influence of class imbalance on cost-sensitive learning: An empirical study. In: In Sixth International Conference on Data Mining IEEE, pp 970–974
46. Liu Y, Chawla N, Shriberg E, Stolcke A, Harper M (2003) Resampling techniques for sentence boundary detection: a case study in machine learning from imbalanced data for spoken language processing. Tech. rep
47. Mladenic D, Grobelnik M (1999) Feature selection for unbalanced class distribution and naive bayes. In: ICML, vol. 99, pp 258–267
48. Novianti PW, Roes KC, Eijkemans MJ (2014) Evaluation of gene expression classification studies: Factors, associated with classification performance. PloS one 9(4) e96:063
49. Orozco M, García ME, Duin RP, Castellanos CG (2006) Dissimilarity-based classification of seismic signals at nevado del ruiz volcano. Earth Sci Res J 10(2)
50. Orozco-Alzate M, Castellanos-Domínguez C (2007) Nearest feature rules and dissimilarity representations for face recognition problems Face Recognition; International Journal of Advanced Robotic Systems, Vienna, Austria, pp 337–356
51. Paclik P, Duin R (2003) Classifying spectral data using relational representation. In: In: Proceedings of the Spectral Imaging Workshop
52. Paclik P, Duin R (2003) Dissimilarity-based classification of spectra: computational issues. Real-Time Imaging 9(4):237–244
53. Pang-Ning T, Steinbach M, Kumar V (2007) Introduction to data mining
54. Pang-Ning T, Steinbach M, Kumar V, et al. (2006) Introduction to data mining. In: Library of Congress
55. Pedrycz W, Loia V, Senatore S (2004) P-fcm: a proximity based fuzzy clustering. Fuzzy Sets Syst 148(1):21–41
56. Pekalska E, Duin R (2002) Dissimilarity representations allow for building good classifiers. Patte Recognition Letters 23(8):943–956
57. Pekalska E, Duin R, Paclik P (2006) Prototype selection for dissimilarity-based classifiers. Pattern Recog 39(2):189–208
58. Pekalska E, Duin RP (2000) Classifiers for dissimilarity-based pattern recognition. In: International Conference on Pattern Recognition
59. Pekalska E, Duin RPW (2006) Dissimilarity-based classification for vectorial representations. In: International Conference on Pattern Recognition, vol. 3, pp 137–140
60. Pekalska E, Paclik P, Duin RP (2002) A generalized kernel approach to dissimilarity-based classification. The J Mach Learn Res 2:175–211
61. Pelayo L, Dick S (2007) Applying novel resampling strategies to software defect prediction. In: In: Conference of the North American Fuzzy Information Processing Society IEEE, pp 69–72
62. Pkekalska E, Duin RP (2002) Dissimilarity representations allow for building good classifiers. Pattern Recog Lett 23(8):943–956
63. Pkekalska E, Duin RP (2005) The dissimilarity representation for pattern recognition: foundations and applications. 64. World Scientific
64. Raskutti B, Kowalczyk A (2004) Extreme re-balancing for svms: a case study. ACM Sigkdd Explorations Newsletter 6(1):60–69
65. Sakoe H, Chiba S (1978) Dynamic programming algorithm optimization for spoken word recognition IEEE. Trans Acoustics Speech Signal Process 26(1):43–49
66. Song Q, Ni J, Wang G (2013) A fast clustering-based feature subset selection algorithm for high-dimensional data. IEEE Trans Knowl Data Eng 25(1):1–14
67. Sørensen L, Loog M, Lo P, Ashraf H, Dirksen A, Duin RP, de Bruijne M (2010) Image dissimilarity-based quantification of lung disease from CT. Springer
68. Sun Z, Song Q, Zhu X (2012) Using coding-based ensemble learning to improve software defect prediction. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews 42(6):1806–1817

69. Van Der Putten P, Van Someren M (2004) A bias-variance analysis of a real world learning problem: The coil challenge 2000. Mach Learn 57(1-2):177–195
70. Van Hulse J, Khoshgoftaar TM, Napolitano A (2007) Experimental perspectives on learning from imbalanced data. In: Proceedings of the 24th international conference on Machine learning. ACM, Corvallis, pp 935–942
71. Van Hulse J, Khoshgoftaar TM, Napolitano A, Wald R (2009) Feature selection with high-dimensional imbalanced data. In: IEEE International Conference on Data Mining Workshops, IEEE, pp 507–514
72. Wasikowski M, Chen X (2010) Combating the small sample class imbalance problem using feature selection. IEEE Trans Knowl Data Eng 22:1388–1400
73. Weiss G (2004) Mining with rarity: a unifying framework. Sigkdd Explorations 6(1):7–19
74. Weiss GM, Provost F (2001) The effect of class distribution on classifier learning: an empirical study Rutgers University
75. William C (1995) Fast effective rule induction. In: Twelfth International Conference on Machine Learning, pp 115–123
76. Yao JK, Dougherty Jr GG, Reddy RD, Keshavan MS, Montrose DM, Matson WR, McEvoy J, Kaddurah-Daouk R (2010) Homeostatic imbalance of purine catabolism in first-episode neuroleptic-naïve patients with schizophrenia. PLoS One 5(3):e9508
77. Yin L, Ge Y, Xiao K, Wang X, Quan X (2013) Feature selection for high-dimensional imbalanced data. Neurocomputing 105:3–11
78. Yu L, Liu H (2003) Feature selection for high-dimensional data: A fast correlation-based filter solution. In: ICML, vol. 3, pp 856–863
79. Yu L, Liu H (2004) Efficient feature selection via analysis of relevance and redundancy. The J Mach Learn Res 5:1205–1224
80. Zheng Z, Srihari R (2003) Optimally combining positive and negative features for text categorization. In: ICML 2003 Workshop
81. Zheng Z, Wu X, Srihari R (2004) Feature selection for text categorization on imbalanced data. ACM SIGKDD Explorations Newsletter 6(1):80–89

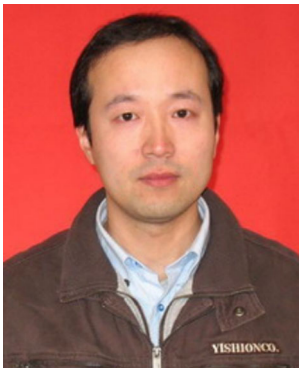
Xueying Zhang received the BS degree from Northwestern Polytechnical University in software and microelectronics in 2009. Currently, She is a Ph.D. student in the Department of Computer science and Technology, Xian Jiaotong University, Xian, China. Her main research interests include data mining and software engineering, especially focusing on classification algorithm improvement and software defect prediction.

Qinbao Song received the Ph.D. degree in computer science from Xian Jiaotong University, Xian, China, in 2001. He is currently a Professor of software technology in the Department of Computer Science and Technology, Xian Jiaotong University, where he is also the Deputy Director of the Department of Computer Science and Technology. He is also with the State Key Laboratory of Software Engineering, Wuhan University, Wuhan, China. He has authored or coauthored more than 80 referred papers in the areas of machine learning and software engineering. He is a board member of the Open Software Engineering Journal. His current research interests include data mining/machine learning, empirical software engineering, and trustworthy software.



Guangtao Wang received the Ph.D. degree in computer science from Xian Jiaotong University, Xian, China, in 2013. He is currently a lecturer in the Department of Computer Science and Technology, Xian Jiaotong University. His research focuses on feature subset selection and algorithm automatic recommendation.

Kaiyuan Zhang received his master degree in Peking University, China, in 2012. He is now a Ph.D candidate at Department of Computer Science and Technology, Xi'an Jiaotong University, China. His research interests include software engineering, data mining and feature subset selection in Data Mining.



Liang He received his Ph.D degree in School of Electronic and Information Engineering from Xi'an Jiaotong University, China, in 2011. He is now a lecturer at the Department of Computer Science and Technology, Xi'an Jiaotong University. His main research interests include data mining and its Application in Software Engineering.

Xiaolin Jia received her Ph.D degree in Computer Software and Theory from Xi'an Jiaotong University, China, in 2006. She is now a Senior Engineer, Xi'an Jiaotong University. Her research interests include software engineering and reliability.