Applied Intelligence (2019) 49:1127–1145
https://doi.org/10.1007/s10489-018-1314-z

Improving lazy decision tree for imbalanced classification by using skew-insensitive criteria

Chong Su[1] · **Jie Cao**[2,3]

Published online: 25 October 2018

Abstract
Lazy decision tree (LazyDT) constructs a customized decision tree for each test instance, which consists of only a single path from the root to a leaf node. LazyDT has two strengths in comparison with eager decision trees. One is that LazyDT can build shorter decision paths than eager decision trees, and the other is that LazyDT can avoid unnecessary data fragmentation. However, the split criterion used for constructing a customized tree in LazyDT is information gain, which is skew-sensitive. When learning from imbalanced data sets, class imbalance impedes their ability to learn the minority class concept. In this paper, we use Hellinger distance and K-L divergence as split criteria to build two types of lazy decision trees. An experimental framework is performed across a wide range of imbalanced data sets to investigate the effectiveness of our methods when comparing with the other methods including lazy decision tree, C4.5, Hellinger distance based decision tree and support vector machine. In addition, we also use SMOTE to preprocess the highly imbalance data sets in the experiment and evaluate its effectiveness. The experimental results, which contrasted through nonparametric statistical tests, demonstrate that using Hellinger distance and K-L divergence as the split criterion can improve the performances of LazyDT for imbalanced classification effectively.

Keywords Imbalanced learning · Lazy decision tree · Hellinger distance · K-L divergence · SMOTE

1 Introduction

Decision trees are among the more popular classification methods due to their efficiency, simplicity and interpretability. Algorithms for constructing decision trees such as C4.5 [1] create a single decision tree during the training phase and then use the tree to classify test instances. In contrast, the lazy decision tree (LazyDT) builds a customized tree for each test instance, which consist of only a single path from the root to leaf node [2]. There are two strengths in comparison with eager decision trees. Firstly, the decision paths built by LazyDT are often shorter than paths of eager decision trees. Secondly, LazyDT can overcome the data fragment problem [3]. However, LazyDT has not taken into account the imbalanced data in which one class (the majority class) vastly outnumbers the other (the minority class). Due to the natures of learning algorithms, learning from imbalanced data sets is still a challenging problem and exists widely in many domains including, but not limited to, fraud detection [4], network intrusion [5], medical diagnosis [6] and so on. In order to overcome the class imbalance problem, many methods have been proposed and can be roughly divided into three categories as follows [7–9]. (1) Data-level methods often concentrate on modifying the training set until all the classes are approximately equally represented. The most common methods employed are undersampling of the majority class and oversampling of the minority classes. Since undersampling of the majority class does not take full advantage of the useful information from the majority class, oversampling has received more attention. A very popular approach is SMOTE (Synthetic Minority Oversampling TEchnique), which increases diversity by generating pseudo

✉ Jie Cao
gzh669@163.com

[1] Key Laboratory of Meteorological Disaster, Ministry of Education (KLME), Joint International Reseach Laboratory of Climate and Environment Change (ILCEC), Collaborative Innovation Center on Forecast and Evaluation of Meteorological Disasters (CIC-FEMD), School of Information and Control, Nanjing University of Information Science and Technology, Nanjing 210044, China

[2] Present address: Nanjing 210044, China

[3] School of Mathematical and Statistics, Nanjing University of Information Science and Technology, Nanjing 210044, China

minority class data [10]. Depending on the technique of how synthetic samples will be generated, Several variants of SMOTE have been proposed such as Borderline-SMOTE [11], Adaptive Synthetic Sampling Technique (ADASYN) [12], MSMOTE [13], and MWMOTE [14]. In addition, when a dataset is highly dimensional and highly class imbalanced, existing online feature selection algorithms usually ignore the small classes which can be important in these applications. It is hence a challenge to select the features from highly dimensional and class imbalanced data in an online manner. A new Online Feature Selection based on the Dependency in K nearest neighbors (K-OFSD) uses the information of nearest neighbors to select relevant features [15]. (2) Algorithm-level methods directly modify existing algorithms to alleviate the bias towards the majority class. The most common methods modified are decision trees and support vector machines (SVMs). For decision trees, the main approach is to use the skew-insensitive split criteria to generate the decision tree, which can be seen from the next paragraph for details. For SVMs, one approach is to adjust the class boundary towards the majority class based on the kernel-alignment ideal [16]; the other is to modify twin support vector machine (TSVM) such as maximum margin of twin spheres support vector machine (MMTSSVM) and maximum margin of twin spheres support vector machine with pinball loss (Pin-MMTSSVM). MMTSSVM finds two homocentric spheres by solving a quadratic programming problem and a linear programming problem. The small sphere contains as many positive samples as possible, while most negative samples are pushed outside the large sphere [17]. In contrast with MMTSSVM, Pin-MMTSSVM uses pinball loss instead of hinge loss used in MMTSSVM to improve the generalization performance of MMTSSVM [18]. (3) Hybrid methods can combine the advantages of two previous groups. One strategy is to resample the dataset and then use the standard classifiers such as decision trees, Naive Bayes and SVM etc. The other is combine pre-processing with bagging and boosting. SMOTEBagging [19], SMOTEBoost [20], EasyEnsemble [21] and BalanceCascade [21] are examples of this type of approaches. By contrast, the Hybrid methods have more advantages and vaster development space [22]. Moreover, multi-class imbalance classification is widely applied in many areas. In recent years, a new ensemble learning is proposed to tackle the multi-class imbalance classification, which uses OVO (one vs one) decomposition scheme to divided m-class imbalance classification problem into $m(m-1)/2$ binary subproblems and utilizes a confidence degree matrix to finish the aggregation [23].

In terms of decision trees, the split criteria used in decision trees such as Gini index, information gain and DKM have been proven to be skew-sensitive and the Hellinger distance is proposed as a split criterion to build Hellinger distance decision trees (HDDT) [24, 25]. Meanwhile, the variants of HDDT are proposed to deal with different problems. For example, the Multi-Class HDDT (MHDDT) employs the techniques similar to Error Correcting Output Codes Decomposition (ECOC) to deal with the multi-class imbalance classification [26]; Gaussian Hellinger Very Fast Decision Tree (GH-VFDT) computes the Hellinger distance between two normal distributions P and N straightforwardly, which means it can deal with imbalanced data streams [27]. As for LazyDT, it adopts information gain to build the decision path, which impedes the ability of LazyDT to learn the minority class concept. So for all of these reasons, we use Hellinger distance and K-L divergence as split criteria to build LazyDT respectively, namely lazy decision tree based on Hellinger distance (HLazyDT) and lazy decision tree based on K-L divergence (KLLazyDT). An experimental framework is performed across a wide range of imbalanced data sets to investigate the effectiveness of our methods when comparing with the other methods including lazy decision tree, C4.5, Hellinger distance based decision tree and support vector machine. The data sets used in this experiment are categorized into two groups: highly and lowly imbalanced data sets according to imbalanced ratio. In addition, we also use SMOTE [10] to preprocess the highly imbalanced data sets in the experiment and evaluate its effectiveness since it has become the de facto standard for improving the performances of the decision tree algorithms [28]. The experimental results, which are contrasted through nonparametric statistical tests, demonstrate that using Hellinger distance and K-L divergence as the split criterion to build LazyDT can improve the performances of LazyDT for imbalanced classification effectively.

In summary, the key contributions of this paper are as follows. (1) We analyse the reason why the split criterion used in LazyDT is skew-sensitive. (2) We use Hellinger distance and K-L divergence as the split criteria used in LazyDT, which demonstrates that the modification can improve the performances of LazyDT for imbalanced classification effectively. (3) We investigate the effectiveness of the SMOTE preprocessing.

The rest of this paper is organized as follows. In Section 2, some related works are introduced. Section 3 provides the formulations and some related properties of Hellinger distance and K-L divergence respectively. Section 4 describes Hellinger distance based lazy decision tree (HLazyDT) and K-L divergence based lazy decision tree (KLLazyDT). Section 5 states the used experimental framework and the analysis of experimental results. Finally, the conclusions obtained in this work are shown in Section 6.

2 Related work

As with many lazy algorithms, the first part of training phase is non-existent and all the work of LazyDT is done during the classification of a given test instance. LazyDT, which gets the test instance as part of the input, follows a separate and classify methodology: a feature is chosen and the sub-problem containing the instances with the same feature value as the given instance is then solved recursively. The overall effect is that of tracing a path in an imaginary tree built for the test instance. The heart of LazyDT is information gain that is used as the split criterion to select the split feature. Unlike eager decision trees, LazyDT takes into account the information of not only the training instances but also the test instance. Essentially, LazyDT builds a single decision path for a given test instance, which can avoid splits on the features that are irrelevant for the specific instance. Therefore, LazyDT avoids unnecessary data fragmentation and duplicate subtrees which are shown in Fig. 1 and may produce a shorter and more accurate decision path for the specific instance. On the other hand, three problems need to be solved in the framework of LazyDT. The first is the way to handle missing values. LazyDT will ignore the missing feature and just never branch on a missing value in the test instance. The second problem is that the information gain may be negative or zero. In this case, LazyDT will return the most frequent class in the training set which covered by the current node. The last problem is that there may be no training instances with the same feature value as the given instance. At this time, LazyDT will remount to the parent node and return the most frequent class in the training set. At last, LazyDT is also applied to distributed privacy preserving area in recent years from an application point of view [29].

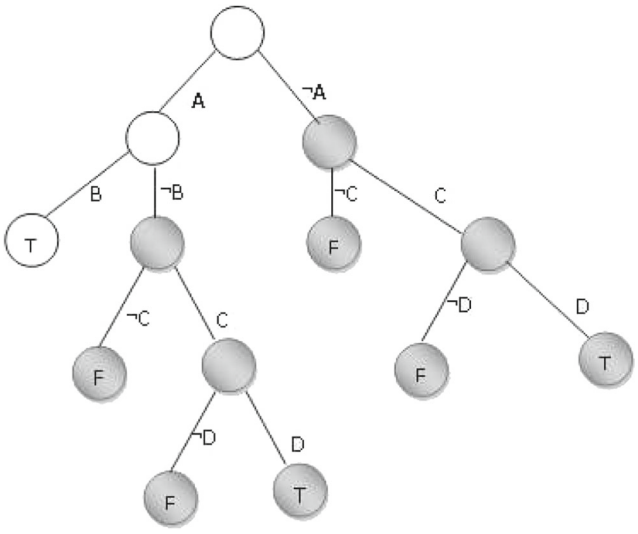

Fig. 1 Data fragmentation and duplicate subtrees in eager decision tree

Since Boosting can improve the performance of various decision trees effectively [30, 31], Fern X Z and Brodley C E proposed a relevance-based boosting lazy decision trees which consist of two steps [32]. Firstly, it builds a customized LazyDT ensemble for each test instance. All the training instances are equally weighted at the beginning. In each iteration, a decision path for the given test instance is produced by applying LazyDT to the training set with instance weights. The instance weight is then adjusted to form a new distribution for the next iteration according to how relevant this instance is to classifying the given test instance and whether its class label is predicted correctly by the current decision path. The relevance level is the depth of the node at which the training instance is discarded when producing the given decision path. Secondly, the relevance-based boosting lazy decision trees also adopt a distance-based pruning strategy to address the problem of over-fitting. In each iteration, the pruning strategy performs a greedy search to select a deletion of feature in the light of the ratio of the heterogeneous distance to the homogeneous distance. The heterogeneous distance measures the distance between the test instance and the closest cluster of instances from the other classes, while the homogeneous distance is defined as the distance between the test instance and the most frequent class in the current set. Although an empirical comparison to the other boosted regular decision trees shows that the relevance-based boosting lazy decision trees achieve comparable accuracy and better comprehensibility, the distance-based pruning strategy is biased against the minority class. Thus, it is difficult for the relevance-based boosting lazy decision trees to deal with the imbalanced data sets.

Due to the fact that LazyDT needs to be re-run independently for each test instance, the main drawback of it is its high computational cost in the classification phase. In order to reduce the computational cost of LazyDT, a batched lazy decision tree was introduced by Guillame-Bert M and Dubrawski A [33]. The batched lazy decision tree substitutes a subtree for a single path and each necessary node is visited only once through a single pass. This algorithm uses the value of the test instance on the split feature to filter the training set and the test set simultaneously in each iteration. Once the termination condition is satisfied, the filtered test instances are labeled as the most frequent class.

In summary, these algorithms mentioned above do not take into account imbalanced learning. The framework of them all takes advantage of information gain as split criterion. It is well known that information gain is a

measure of impurity. As far as the imbalanced data sets are concerned, the distributions of the classes are unbalanced and it means that the impurity of imbalanced data is lower than balanced data. When using information gain as the split criterion to choose the feature in a imbalanced data set, the information gain may be zero or even negative, which will lead to cessation of tree growth. In this way, it is unreasonable that all of the test instances may be classified as the majority class, which decreases the performances of the classifier. This is the reason why information gain is skew-sensitive.

3 Hellinger distance and K-L divergence

3.1 Hellinger distance

Hellinger distance is one of measures which reflects divergences between two probability distributions [34]. Given a countable space Φ, let P and Q be two distributions and Hellinger distance can be defined as follows.

$$D_H(P, Q) = \sqrt{\sum_{\phi\in\Phi}(\sqrt{P(\phi)} - \sqrt{Q(\phi)})^2} \quad . \tag{1}$$

The Hellinger distance carries the following properties.

(1) $D_H(P, Q)$ is in $[0,\sqrt{2}]$.
(2) The Hellinger distance is symmetric and non-negative.
(3) The bigger Hellinger distance is, the better discriminations of the probabilities are.

The Hellinger Distance with respect to a feature for two-class problem is given as follows, which is firstly used as a split criterion in decision trees [24].

$$D_H(X_+, X_-) = \sqrt{\sum_{j=1}^{P}(\sqrt{\frac{|X_{+j}|}{|X_+|}} - \sqrt{\frac{|X_{-j}|}{|X_-|}})^2} \quad , \tag{2}$$

where $|X_+|$ indicates the number of instances that belong to the minority class in the training set and $|X_{+j}|$ specifies the subset of the training set with the minority class and value j for the feature X. Similarly, they are the same explanations of $|X_-|$ and $|X_{-j}|$ but for the majority class. In addition, P is the number of different values in the feature X. Since (2) is not influenced by prior probability, it is insensitive to class distributions.

3.2 K-L divergence

Kullback-Leibler Divergence is an asymmetric measure which reflects divergences between two probability distributions and generally abbreviates to K-L divergence [35]. Suppose Φ is a countable space and both P and Q are two probability distributions in Φ respectively. $D_{kl}(P, Q) = \sum_{\phi\in\Phi}\left(\ln(\frac{P(\phi)}{Q(\phi)}) \cdot P(\phi)\right)$ indicates the K-L divergence of Q from P. Similarly, the K-L divergence of P from Q can be defined as $D_{kl}(Q, P) = \sum_{\phi\in\Phi}\left(\ln(\frac{Q(\phi)}{P(\phi)}) \cdot Q(\phi)\right)$. In order to make K-L divergence satisfy the symmetry, a symmetrised form of K-L divergence is defined as follows.

$$\begin{aligned} D_{kl-symmetry}(P, Q) &= \frac{1}{2} \cdot D_{kl}(P, Q) + \frac{1}{2} \cdot D_{kl}(Q, P) \\ &= \frac{1}{2} \cdot \sum_{\phi\in\Phi}\left(\ln(\frac{P(\phi)}{Q(\phi)}) \cdot P(\phi)\right) \\ &\quad + \frac{1}{2} \cdot \sum_{\phi\in\Phi}\left(\ln(\frac{Q(\phi)}{P(\phi)}) \cdot Q(\phi)\right) \\ &= \frac{1}{2} \cdot \sum_{\phi\in\Phi}((\ln P(\phi) - \ln Q(\phi)) \cdot P(\phi) \\ &\quad + (\ln Q(\phi) - \ln P(\phi)) \cdot Q(\phi)) \\ &= \frac{1}{2} \cdot \sum_{\phi\in\Phi}((P(\phi) \\ &\quad - Q(\phi)) \cdot \ln\left(\frac{P(\phi)}{Q(\phi)}\right)\Big). \end{aligned} \tag{3}$$

Especially, when using the symmetrised form of K-L divergence as the splitting criterion of decision trees for two-class problem, we can ignore the constant term namely $\frac{1}{2}$ and rewrite the (3) as follows.

$$D_{kl-symmetry}(X_+, X_-) = \sum_{j=1}^{P}\left(\left(\frac{|X_{+j}|}{|X_+|} - \frac{|X_{-j}|}{|X_-|}\right) \cdot \ln\left(\frac{\frac{|X_{+j}|}{|X_+|}}{\frac{|X_{-j}|}{|X_-|}}\right)\right). \tag{4}$$

Since all symbols used in the (4) have the same illustrations as the (2), we will not repeat them here and please refer to the Section 3.1 for details. It essentially reflects the divergence between the feature value distributions with respect to two different classes and has the following properties.

(1) $D_{kl-symmetry}(X_+, X_-)$ is bounded in $[0, +\infty)$.
(2) $D_{kl-symmetry}(X_+, X_-)$ satisfy symmetry and non-negativeness.
(3) The bigger $D_{kl-symmetry}(X_+, X_-)$ is, the better discriminations of the feature are.

However, there exists zero probability problem which makes the upper bound of the (4) infinite. Smoothing methods are generally used to avoid this problem. In this paper, we adopt Laplace smoothing namely additive smoothing. For example, assuming $|X_{+j}| = 0$ and there are m_j different values in the feature X, the smoothed conditional probability is defined as follows.

$$P(X_j|+) = \frac{|X_{+j}| + 1}{|X_+| + \lambda \cdot m_j}, \tag{5}$$

where λ is the smoothing parameter. Here, we set $\lambda = 1$, that is to say, Laplace's law of succession [36]. In this way, we can use the (5) to tackle zero probability problem.

4 Hellinger distance and K-L divergence based lazy decision trees

Although LazyDT avoids unnecessary data fragmentation and duplicate subtrees and may produce a shorter and more accurate decision path for the specific instance, it is still difficult for LazyDT to deal with imbalanced problem. There are two main defects which impede LazyDT's ability to learn from imbalanced data. one is that information gain used by LazyDT is skew-sensitive and the other is that LazyDT do not take into account the value of the test instance on the alternative feature in the procedure of selecting the split feature, which may give rise to the problem that there are no instances in the new node when filtering the training set by the value of the test instance on the split feature. In this way, LazyDT will cease to split and return the most frequent class in the training set covered by the current node, which degrades the classification performance of LazyDT. In order to overcome the two defects mentioned above, we use the value of the test instance on the candidate feature to modify Hellinger distance and K-L divergence respectively.

4.1 Hellinger distance based lazy decision trees (HLazyDT)

For the feature X, let v represent the value of a given test instance. Given two-class problem, we can calculate the Hellinger distance of the value v on the feature X as follows.

$$W_{H-v}(X_+, X_-) = \left| \sqrt{\frac{|X_{+v}|}{|X_+|}} - \sqrt{\frac{|X_{-v}|}{|X_-|}} \right| \quad , \tag{6}$$

where $|X_+|$ indicates the number of examples that belong to the minority class in the training set and $|X_{+v}|$ specifies the subset of training set with the minority class and the value v for the feature X. Similarly, they are the same explanations of $|X_-|$ and $|X_{-v}|$ but for the majority class. The bigger is W_{H-v}, the better discriminations of the value v on the feature X are. It reflects the contribution of the value v to the discrimination of the feature X.

As mentioned above, we combine the (2) with the (6) and give the formula of Hellinger distance based split criterion used in HLazyDT as follows.

$$Split_H = \sqrt{W_{H-v}(X_+, X_-) \cdot D_H(X_+, X_-)} \quad , \tag{7}$$

where $D_H(X_+, X_-)$ and $W_{H-v}(X_+, X_-)$ can be calculated according to the (2) and the (6) respectively. Especially, when the value v of the test instance on the feature X does not belong to the set of the values on the feature X in the training set, both the (6) and (7) will be equal to zero. In fact, the (7) calculates the ability of the feature X to discriminate the two classes based not only on all values of the feature X but also on the value of the test instance on the feature X.

The following Algorithm 1 outlines the approach for calculating Hellinger distance and the procedure for inducing HLazyDT. In this algorithm, $T_{y=i}$ indicates the subset of the training set T with class i, $T_{f=j}$ specifies the subset with the value j for the feature f and $T_{f=j,y=i}$ identifies the subset with the class i and has the value j for the feature f. Given two-class problem, class i is drawn from some finite set of classes like the minority class(+) and the majority class(-).

Algorithm 1 HLazyDT

Input: Training Set T, an unlabelled test instance I, Feature Set F
Output: A label for the instance I
1: Create root node
2: If all instances in T have the same class i THEN
3: Return i
4: END If
5: If all instances in T have the same feature values or $F == \varnothing$ THEN
6: Return the most frequent class in T
7: END If
8: $D_{max} \longleftarrow 0; sf \longleftarrow NULL$
9: For each feature f of F DO
10: $v_{fI} \longleftarrow$ get the value of the test instance I on the feature f
11: //ignore the feature f on which there are missing values for the test instance I
12: If v_{fI} is Null THEN
13: CONTINUE
14: END If
15: $D_f \longleftarrow 0$ //the Hellinger distance of the feature f
16: $D_v \longleftarrow 0$ //the Hellinger distance of the test instance I on the feature f
17: $D_g \longleftarrow 0$ //the geomean of D_f and D_S
18: For each value v of f DO //Calculate Hellinger distance
19: $D_f += (\sqrt{\frac{|T_{f=v,y=+}|}{|T_{y=+}|}} - \sqrt{\frac{|T_{f=v,y=-}|}{|T_{y=-}|}})^2$
20: If v == v_{fI} THEN $D_v = \left| \sqrt{\frac{|T_{f=v,y=+}|}{|T_{y=+}|}} - \sqrt{\frac{|T_{f=v,y=-}|}{|T_{y=-}|}} \right|$
21: END For
22: $D_f = \sqrt{D_f}; D_g = \sqrt{D_v \cdot D_f}$
23: If $D_{max} < D_g$ THEN { $D_{max} = D_g; sf = f$ }
24: END For
25: $T^* \longleftarrow \{i | i \in$ T and has the same value as the test instance I on the feature sf }
26: $F \longleftarrow F \backslash sf$ //remove the split feature sf from the feature set F
27: HLazyDT(T^*, I, F)

4.2 K-L divergence based lazy decision trees (KLLazyDT)

Similarly, we firstly calculate K-L divergence of the value v as follows.

$$W_{KL-v}(X_+, X_-) = \left(\frac{|X_{+v}|}{|X_+|} - \frac{|X_{-v}|}{|X_-|}\right) \cdot \ln\left(\frac{\frac{|X_{+v}|}{|X_+|}}{\frac{|X_{-v}|}{|X_-|}}\right), \quad (8)$$

where all symbols have the same illustrations as the (6) and please refer to the Section 4.1 for details. It reflects the discriminations of the value v of the test instance on the feature X. Next, we describe the formula of K-L divergence based split criterion used in KLLazyDT as follows.

$$Split_{KL} = \sqrt{W_{KL-v}(X_+, X_-) \cdot D_{kl-symmetry}(X_+, X_-)} \quad , \quad (9)$$

where $D_{kl-symmetry}(X_+, X_-)$ and $W_{KL-v}(X_+, X_-)$ can be calculated according to the (4) and (8) respectively. Likewisely, if the value v of the test instance on the feature X does not belong to the set of the values on the feature X in the training set, both the (4) and (8) will also be equal to zero. Moreover, it is worth mentioning that we still need use the (5) to smooth the (4) and (8) when zero probability problems come out. The framework of the K-L divergence based lazy decision trees is described in Algorithm 2. Please refer to Algorithm 1 for the illustrations of the related symbols.

Finally, we focus on the time and memory complexity of eager decision trees and lazy decision trees, which have been discussed in [33]. In order to make this paper self-contained for all its related information, we just restate the related discussions. For convenience of discussion, we consider binary splitting for decision trees, a training dataset with n instances and f attributes as well as a test dataset with t instances and f attributes. We assume that there are a minimum of p training instances in expanding nodes. (1) The time complexity of an eager decision tree model is $O(\sum_{i=0}^{\lg \frac{n}{p}} 2^i \cdot f \cdot \frac{n}{2^i}) = O(n \cdot f \cdot \lg \frac{n}{p})$ and the time complexity of a lazy decision tree model is $O(t \cdot \sum_{i=0}^{\lg \frac{n}{p}} f \cdot \frac{n}{2^i}) = O(t \cdot f \cdot n)$ on average. Note that $\lg \frac{n}{p}$ is the average depth of exploration, 2^i is the average number of nodes at depth i, and $\frac{n}{2^i}$ is the average number of training instances contained in a node at depth i. Thus, unless the number of unlabeled observations is small, lazy decision trees can be slower in prediction mode than the equivalent eager decision trees. (2) Building an eager decision tree requires memories for the stack operations used during the recursive construction of the tree as well as some storages for the final trained model, while Lazy decision trees only require the stack. The stack size and model size of decision trees are $O(\sum_{i=0}^{\lg \frac{n}{p}} \frac{n}{2^i}) = O(n)$ and $O(\frac{n}{p})$ (the number of non-leaf nodes in a decision tree) on average respectively. According to the notation mentioned above, the required memories of an eager decision tree is $O(n) + O(\frac{n}{p})$ on average in the training phase and $O(\frac{n}{p})$ in the testing phase, while Lazy decision trees only require $O(n)$. It means that eager decision tree requires more memories in the training phase than lazy decision trees, while in the testing phase it yet requires fewer memories than lazy decision trees.

Algorithm 2 KLLazyDT

Input: Training Set T, an unlabelled test instance I, Feature Set F
Output: A label for the instance I
1: Create root node
2: If all instances in T have the same class i THEN
3: Return i
4: END If
5: If all instances in T have the same feature values or $F == \varnothing$ THEN
6: Return the most frequent class in T
7: END If
8: $D_{max} \longleftarrow 0$; $sf \longleftarrow NULL$
9: For each feature f of F DO
10 $v_{fI} \longleftarrow$ get the value of the test instance I on the feature f
11: If v_{fI} is Null THEN CONTINUE
12: $m \longleftarrow 0$; $smoothing \longleftarrow none$ //not using Laplace smoothing by default
13: For each value v of f DO //deciding whether or not to using Laplace smoothing
14: If ($|T_{f=v,y=+}| == 0$) THEN $smoothing \longleftarrow +$
15: If ($|T_{f=v,y=-}| == 0$) THEN $smoothing \longleftarrow -$
16: m=m+1 //the number of different values in the feature f
17: END For
18: $D_f \longleftarrow 0$; $D_v \longleftarrow 0$; $D_g \longleftarrow 0$
19: For each value v of f DO //Calculate K-L divergence
20: If ($smoothing == +$) THEN $P_+ = \frac{|T_{f=v,y=+}|+1}{|T_{y=+}|+m}$
21: Else If ($smoothing == -$) THEN $P_- = \frac{|T_{f=v,y=-}|+1}{|T_{y=-}|+m}$
22: Else $\{P_+ = \frac{|T_{f=v,y=+}|}{|T_{y=+}|}, \quad P_- = \frac{|T_{f=v,y=-}|}{|T_{y=-}|}\}$
23: END If
24: $D_f += (P_+ - P_-) \cdot \ln(\frac{P_+}{P_-})$
25: If (v == v_{fI}) THEN $D_v = (P_+ - P_-) \cdot \ln(\frac{P_+}{P_-})$
26: END For
27: $D_g = \sqrt{D_v \cdot D_f}$; If $D_{max} < D_g$ THEN $\{D_{max} = D_g ; sf = f\}$
28: END For
29: $T^* \longleftarrow \{i | i \in$ T and has the same value as the test instance I on the feature sf $\}$
30: $F \longleftarrow F \backslash sf$
31: KLLazyDT(T^*, I, F)

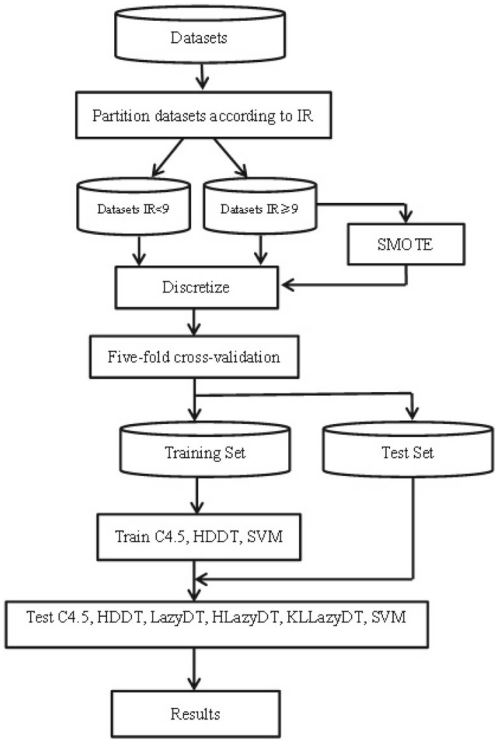

Fig. 2 The framework of the whole experiment

5 Experiments

The whole experiment is divided into the following steps. Firstly, we categorize the experimental data sets into three groups: lowly ($IR < 9$) imbalanced data sets, highly ($IR \geq 9$) imbalanced data sets and highly imbalanced data sets with SMOTE preprocessing. Secondly, we use uniform-frequency method to discretize the datasets used in the experiment. Finally, we use five-fold cross-validation to train and test the methods which participate in the experiment and obtain and analyze the experimental results. In order to depict the whole procedure of our experiment clearly, we give the whole framework of our experiment as shown in Fig. 2. The methods' abbreviations in Fig. 2 are shown in Table 1.

5.1 Experiment tool

In this section, we present the set up of the experimental framework that is used to develop the analysis of our proposals. Moreover, we use KEEL (Knowledge Extraction based on Evolutionary Learning)[1] to finish the whole experiment, which empowers the user to assess the behavior of evolutionary learning and soft computing based techniques for all kinds of data mining problems: regression, classification, clustering, pattern mining and so on. KEEL is an open source Java framework (GPLv3 license) that provides a number of modules to perform a wide variety of data mining tasks. It includes tools to perform data management, add a new algorithm, design of multiple kind of experiments, statistical analyses, etc and provides a simple GUI to design experiments with different data sets and computational intelligence algorithms in order to assess the behaviour of the algorithms. The latest version of it is KEEL3.0, which includes new modules for semi-supervised learning, multi-instance learning, imbalanced classification and subgroup discovery [37]. In our experiment, We first use KEEL3.0 to implement Hellinger distance based lazy decision tree and K-L divergence based lazy decision tree respectively and then use the imbalanced classification and datasets provided by KEEL3.0 to finish the whole experiment.

5.2 Methods

In this experiment, we compare Hellinger distance based lazy decision tree (HLazyDT) and K-L divergence based lazy decision tree (KLLazyDT) with the other methods including lazy decision tree (LazyDT), C4.5, Hellinger distance based decision tree (HDDT) and support vector machine (SVM) across lowly and highly imbalanced data sets. Thereinto, C4.5 is a popular algorithm for eager decision trees which was used in the experiment of lazy decision tree [1]. Since imbalanced classification is the focus of this experiment, we use the version of C4.5 with unpruning in this experiment, which was proven to be an effective strategy [38]. Similarly, HDDT is an eager decision tree which is designed for imbalanced classification [24, 25]. Additionally, SVM is a popular approach for data classification and has been successfully applied in various applications [39]. Since SVM does not belong to the family of decision trees, we only use SVM as the comparison object in the experiment and the statistical analysis of the experimental results will not include it. Finally, all methods used in this experiment, their parameters setting as well as their abbreviations are shown in Table 1. We will use these abbreviations of all methods to denote for the rest of this paper.

5.3 Datasets and data partitions

In this experiment, we select data sets from KEEL data set repository[2] and categorized these data sets into two groups: lowly ($IR < 9$) and highly ($IR \geq 9$) imbalanced data sets according to the ratio of the number of examples from the majority class to the minority class(IR). Although there is no consensus in the literature about when a data

[1] http://www.keel.es

[2] http://www.keel.es

Table 1 All methods used in the experiment and their parameters settings and abbreviations

Methods	Parameters Setting	Abbreviations
C4.5	Unprune, InstancePerLeaf=2	C4.5
Hellinger distance based decision tree	Unprune	HDDT
Lazy Decision Tree		LazyDT
Hellinger distance based lazy decision tree [a]		HLazyDT
K-L divergence based lazy decision tree [a]		KLLazyDT
Support vector machine	kernel function=RBF, $C = 100$, $\epsilon = 0.001$, $\gamma = 0.01$	SVM

Items with a [a] are our approaches in this paper

set is considered highly imbalanced, we will consider that the ratio of the number of examples from the majority class to the minority class (IR) above 9 represents a highly imbalanced data set in this paper due to the fact that ignoring the minority class examples by a classifier still obtains an error of 0.1 in accuracy. The characteristics of data sets are summarized in Tables 2 and 3 where we denote the number of examples, the number of minority class examples, the number of features, the number of continuous features, the number of discrete features and imbalanced ratio by "Examples", "Minority", "Features", "Continuous", "Discrete" and "IR" respectively. These data sets are two-class problem.

On the other hand, when evaluating each of these approaches on the data sets, we firstly make use of the equal-frequency bins to discretize all used data sets since the family of the lazy decision trees only deals with the discrete data. Secondly, in order to explore what effect SMOTE [10] have on these methods used in this experiment, we also use SMOTE to preprocess the highly imbalanced data sets and compare the performances of these methods. Since our goal is to explore SMOTE to obtain the balanced data sets, the SMOTE levels (SL) for minority class can be calculated as follows.

$$SL = (\frac{n - n_+}{n_+} - 1) \times 100\%. \tag{10}$$

Table 2 Lowly imbalanced data sets used in the experiment

NO	Data sets	Examples	Minority	Features	Continuous	Discrete	IR
1	ecoli-0_vs_1	220	77	7	7	0	1.86
2	ecoli1	336	77	7	7	0	3.86
3	ecoli2	336	52	7	7	0	5.46
4	ecoli3	336	35	7	7	0	8.6
5	glass0	214	70	9	9	0	2.06
6	glass-0-1-2-3_vs_4-5-6	214	71	9	9	0	3.2
7	glass1	214	76	9	9	0	1.82
8	glass6	214	29	9	9	0	6.38
9	haberman	306	81	3	3	0	2.78
10	iris0	150	50	4	4	0	2
11	new-thyroid1	215	35	5	5	0	5.14
12	new-thyroid2	215	35	5	5	0	5.14
13	page-blocks0	5472	559	10	10	0	8.79
14	pima	768	268	8	8	0	1.87
15	segment0	2308	329	19	19	0	6.02
16	vehicle0	846	199	18	18	0	3.25
17	vehicle1	846	217	18	18	0	2.9
18	vehicle2	846	218	18	18	0	2.88
19	vehicle3	846	212	18	18	0	2.99
20	wisconsin	683	239	9	9	0	1.86
21	yeast1	1484	429	8	8	0	2.46
22	yeast2	1484	163	8	8	0	8.1

Table 3 Highly imbalanced data sets used in the experiment

NO	Data sets	Examples	Minority	Features	Continuous	Discrete	IR
1	cleveland-0_vs_4	177	13	13	13	0	12.62
2	ecoli-0-1_vs_2-3-5	244	24	7	7	0	9.17
3	ecoli-0-1_vs_5	240	20	6	6	0	11
4	ecoli-0-1-3-7_vs_2-6	281	7	7	7	0	39.14
5	ecoli-0-1-4-6_vs_5	280	20	6	6	0	13
6	ecoli-0-1-4-7_vs_2-3-5-6	336	29	7	7	0	10.59
7	ecoli-0-1-4-7_vs_5-6	332	25	6	6	0	12.28
8	ecoli-0-2-3-4_vs_5	202	20	7	7	0	9.1
9	ecoli-0-2-6-7_vs_3-5	224	22	7	7	0	9.18
10	ecoli-0-3-4_vs_5	200	20	7	7	0	9
11	ecoli-0-3-4-6_vs_5	205	20	7	7	0	9.25
12	ecoli-0-3-4-7_vs_5-6	257	25	7	7	0	9.28
13	ecoli-0-4-6_vs_5	203	20	6	6	0	9.15
14	ecoli-0-6-7_vs_3-5	222	22	7	7	0	9.09
15	ecoli-0-6-7_vs_5	220	20	6	6	0	10
16	ecoli4	336	20	7	7	0	15.8
17	glass-0-1-4-6_vs_2	205	17	9	9	0	11.06
18	glass-0-1-5_vs_2	172	17	9	9	0	9.12
19	glass-0-1-6_vs_2	192	17	9	9	0	10.29
20	glass-0-1-6_vs_5	184	9	9	9	0	19.44
21	glass-0-4_vs_5	92	9	9	9	0	9.22
22	glass-0-6_vs_5	108	9	9	9	0	11
23	glass2	214	17	9	9	0	11.59
24	glass4	214	13	9	9	0	15.47
25	glass5	214	9	9	9	0	22.78
26	poker-8_vs_6	1477	17	10	10	0	85.88
27	poker-8-9_vs_5	2075	25	10	10	0	82
28	poker-8-9_vs_6	1485	25	10	10	0	58.4
29	poker-9_vs_7	244	8	10	10	0	29.5
30	vowel0	988	90	13	13	0	9.98
31	yeast-0-2-5-6_vs_3-7-8-9	1004	99	8	8	0	9.14
32	yeast-0-2-5-7-9_vs_3-6-8	1004	99	8	8	0	9.14
33	yeast-0-3-5-9_vs_7-8	506	50	8	8	0	9.12
34	yeast-0-56-7-9_vs_4	528	51	8	8	0	9.35
35	yeast-1_vs_7	459	30	7	7	0	14.3
36	yeast-1-2-8-9_vs_7	947	30	8	8	0	30.57
37	yeast-1-4-5-8_vs_7	693	30	8	8	0	22.1
38	yeast-2_vs_4	514	51	8	8	0	9.08
39	yeast-2_vs_8	482	20	8	8	0	23.1
40	yeast4	1484	51	8	8	0	28.1
41	yeast5	1484	44	8	8	0	32.73
42	yeast6	1484	35	8	8	0	41.4

Finally, we performed five-fold cross-validation. In this procedure, each data set is randomly partitioned into five equal sized and disjoint subsets, a single subset is retained as the validation data for testing the model and the remaining four subsets are used as the training data. The cross-validation process is then repeated five times and the mean of five repetitions is considered as the final results.

5.4 Evaluation measure

Since the behavior of different evaluation measures can result in potentially different conclusions, it is suggested to use more evaluation measures to evaluate classifier especially when learning from imbalanced datasets [40]. Therefore, we employee the area under the receiver oper-

ating characteristic(ROC) curve and G-mean to compare the performance of all methods used in this experiment respectively which are two popular evaluation measures for imbalanced problem.

(1) The area under the ROC Curve(AUC) can be calculated as follows [41].

$$AUC = \frac{2S_0 - n_+(n_+ + 1)}{2n_+n_-}, \tag{11}$$

where n_+ and n_- indicate instances of the minority class and the majority class respectively. S_0 is the sum of ranks of the minority class instances.

(2) G-mean is computed as the geometric mean of the true positive and true negative rates, i.e.,

$$G - mean = \sqrt{\frac{TP}{TP + FN} \cdot \frac{TN}{TN + FP}}, \tag{12}$$

where TP indicates the number of true positive instances, TN denotes the number of true negative instances, FP means the number of false positive instances, and FN is the number of false negative instances.

5.5 Statistical tests for performance comparison

For the sake of the analysis of the experimental results, non-parametric tests are used for statistical comparisons of all methods used in this experiment due to the fact that the initial conditions that guarantee the reliability of the parametric tests may not be satisfied [42, 43]. In this experiment, we use adjusted p-value Holm procedure to find which methods are distinctive among a $1 \times n$ comparison. In this procedure, we firstly obtain the average ranks of the methods according to the Friendman procedure. Especially, when in the case of a tie, the average rank is assigned to each method. For instance, if the performance of two methods tie for first, a rank of 1.5 is assigned to each, or if three tie for first, a rank of 2 is assigned to each and so on. Secondly, we need choose a base method or a control method. In our experiment, we choose the method with minimum average rank as the base method and arrange the others in ascending order according to the average ranks. The test statistics for comparing the ith with the base method is as follows.

$$Z = \frac{R_i - R_b}{\sqrt{\frac{K(K+1)}{6N}}}, \tag{13}$$

Table 4 The AUC results across lowly imbalanced data sets

NO	Data Sets	C4.5	HDDT	LazyDT	HLazyDT	KLLazyDT	SVM
1	ecoli-0_vs_1	0.964	0.953	0.951	0.983	0.970	0.948
2	ecoli1	0.827	0.835	0.823	0.821	0.777	0.823
3	ecoli2	0.837	0.811	0.804	0.843	0.848	0.880
4	ecoli3	0.696	0.797	0.734	0.745	0.763	0.786
5	glass0	0.806	0.772	0.767	0.749	0.785	0.770
6	glass-0-1-2-3_vs_4-5-6	0.879	0.897	0.887	0.913	0.878	0.860
7	glass1	0.719	0.748	0.705	0.689	0.698	0.770
8	glass6	0.903	0.841	0.883	0.884	0.889	0.894
9	haberman	0.550	0.527	0.555	0.566	0.577	0.563
10	iris0	0.990	1.00	0.990	0.985	0.985	1.00
11	new-thyroid1	0.860	0.937	0.929	0.960	0.963	0.932
12	new-thyroid2	0.923	0.957	0.932	0.992	0.977	0.966
13	page-blocks0	0.896	0.913	0.894	0.905	0.906	0.923
14	pima	0.659	0.673	0.684	0.693	0.673	0.676
15	segment0	0.985	0.994	0.989	0.991	0.990	0.994
16	vehicle0	0.888	0.930	0.914	0.908	0.916	0.964
17	vehicle1	0.672	0.656	0.656	0.676	0.673	0.742
18	vehicle2	0.886	0.914	0.856	0.946	0.950	0.981
19	vehicle3	0.642	0.635	0.614	0.662	0.685	0.715
20	wisconsin	0.944	0.942	0.941	0.958	0.947	0.966
21	yeast1	0.612	0.667	0.622	0.643	0.623	0.654
22	yeast2	0.841	0.845	0.836	0.812	0.825	0.833
	Avg.ROC	**0.817**	**0.829**	**0.817**	**0.833**	**0.832**	**0.847**

where R_i and R_b are the average ranks computed through the Friendman test for the ith method and the base method respectively. K is the number of methods to be compared and N is the number of data sets used in the comparison. The Z value is used to find the corresponding probability(P-value) from the table of normal distribution, which is then compared with an appropriate level of significance α. In adjusted P-value Holm procedure, it starts with the most significant P value. The ith adjusted P-value (APV_i) is defined as follows.

$$APV_i = min\{v; 1\}, where\ v = max\{(k-j)P_j, 1 \le j \le i\}. \quad (14)$$

If $APV_i < \alpha$, the null hypothesis, namely there is no significant difference in the ranks of two methods, is rejected. As soon as a certain null hypothesis cannot be rejected, all remaining hypotheses are retained as well.

5.6 Experiment results and analysis

In this section, we present the empirical analysis of our methods including HLazyDT and KLLazyDT in order to determine their robustness in three different scenarios namely lowly imbalanced data sets, highly imbalanced data sets and highly imbalanced data sets with SMOTE preprocessing. We firstly compare the average AUCs and G-means among these different approaches and then we also compare the average ranks among these different approaches at 95% confidence interval according to AUCs and G-means respectively. If there are statistically significant difference between the base approach and the others, we put a "$\surd$" sign on the corresponding method, which are shown in Table 10.

The experimental results are shown in Tables 4, 5, 6, 7, 8, and 9 respectively. Tables 4, 6, and 8 show the AUC results for each method across three types of the datasets used in this experiment and the rest tables show the G-mean results for each method across three types of the datasets. To compare the average AUC results and G-mean results clearly, we also make the charts which are shown in Figs. 3 and 4 respectively. Finally, The observations can be described as follows.

(1) When comparing the results across the lowly imbalanced data sets, HLazyDT and KLLazyDT outperform the other trees except for SVM according to AUC, which also can be seen in Fig. 3. As shown in Table 10, we note that HLazyDT and KLLazyDT perform statistically significantly better than LazyDT at 95% confidence interval according to the average ranks of AUCs.

Table 5 The G-means results across lowly imbalanced data sets

NO	Data Sets	C4.5	HDDT	LazyDT	HLazyDT	KLLazyDT	SVM
1	ecoli-0_vs_1	0.962	0.952	0.949	0.983	0.966	0.946
2	ecoli1	0.812	0.824	0.814	0.802	0.762	0.814
3	ecoli2	0.829	0.798	0.791	0.816	0.840	0.872
4	ecoli3	0.638	0.759	0.681	0.716	0.739	0.764
5	glass0	0.801	0.763	0.762	0.740	0.773	0.765
6	glass-0-1-2-3_vs_4-5-6	0.877	0.893	0.883	0.909	0.876	0.851
7	glass1	0.706	0.740	0.692	0.650	0.690	0.763
8	glass6	0.892	0.821	0.872	0.885	0.880	0.883
9	haberman	0.467	0.44	0.466	0.507	0.535	0.426
10	iris0	0.990	1.00	0.990	0.985	0.985	1.00
11	new-thyroid1	0.833	0.931	0.923	0.961	0.961	0.925
12	new-thyroid2	0.921	0.955	0.930	0.977	0.977	0.965
13	page-blocks0	0.892	0.910	0.890	0.902	0.904	0.921
14	pima	0.653	0.659	0.674	0.681	0.673	0.666
15	segment0	0.990	0.994	0.989	0.990	0.990	0.994
16	vehicle0	0.885	0.929	0.912	0.890	0.911	0.963
17	vehicle1	0.652	0.633	0.625	0.658	0.649	0.732
18	vehicle2	0.885	0.912	0.850	0.938	0.949	0.981
19	vehicle3	0.603	0.596	0.568	0.643	0.669	0.700
20	wisconsin	0.944	0.941	0.941	0.943	0.947	0.966
21	yeast1	0.574	0.649	0.588	0.621	0.600	0.610
22	yeast2	0.830	0.834	0.824	0.786	0.811	0.820
	Avg.G-means	**0.802**	**0.815**	**0.801**	**0.817**	**0.822**	**0.833**

Similarly, when focusing on the G-mean results, we can obtain the same results as AUC which can be seen from Table 5 and Fig. 4.

(2) When comparing the results across the highly imbalanced data sets, both Tables 6 and 7 show that HLazyDT and KLLazyDT outperform the other trees except for SVM according to AUC or G-mean. This

Table 6 The AUC results across highly imbalanced data sets

NO	Data Sets	C4.5	HDDT	LazyDT	HLazyDT	KLLazyDT	SVM
1	cleveland-0_vs_4	0.631	0.660	0.592	0.622	0.778	0.741
2	ecoli-0-1_vs_2-3-5	0.623	0.733	0.755	0.850	0.870	0.844
3	ecoli-0-1_vs_5	0.802	0.821	0.857	0.911	0.932	0.886
4	ecoli-0-1-3-7_vs_2-6	0.591	0.739	0.643	0.684	0.725	0.637
5	ecoli-0-1-4-6_vs_5	0.767	0.780	0.788	0.833	0.883	0.813
6	ecoli-0-1-4-7_vs_2-3-5-6	0.707	0.744	0.727	0.767	0.824	0.795
7	ecoli-0-1-4-7_vs_5-6	0.709	0.706	0.709	0.849	0.882	0.853
8	ecoli-0-2-3-4_vs_5	0.798	0.822	0.823	0.856	0.853	0.886
9	ecoli-0-2-6-7_vs_3-5	0.815	0.853	0.835	0.815	0.798	0.730
10	ecoli-0-3-4_vs_5	0.828	0.869	0.828	0.908	0.908	0.914
11	ecoli-0-3-4-6_vs_5	0.820	0.819	0.818	0.859	0.884	0.856
12	ecoli-0-3-4-7_vs_5-6	0.758	0.812	0.800	0.854	0.841	0.800
13	ecoli-0-4-6_vs_5	0.773	0.832	0.850	0.848	0.867	0.859
14	ecoli-0-6-7_vs_3-5	0.805	0.858	0.847	0.853	0.840	0.855
15	ecoli-0-6-7_vs_5	0.880	0.876	0.880	0.893	0.885	0.870
16	ecoli4	0.866	0.817	0.891	0.891	0.903	0.891
17	glass-0-1-4-6_vs_2	0.485	0.553	0.480	0.574	0.614	0.638
18	glass-0-1-5_vs_2	0.551	0.630	0.558	0.595	0.585	0.645
19	glass-0-1-6_vs_2	0.538	0.603	0.549	0.599	0.626	0.669
20	glass-0-1-6_vs_5	0.841	0.747	0.747	0.891	0.924	0.894
21	glass-0-4_vs_5	0.994	0.894	0.994	0.832	0.809	1.00
22	glass-0-6_vs_5	0.980	0.978	0.880	0.925	0.895	0.950
23	glass2	0.549	0.572	0.580	0.533	0.543	0.594
24	glass4	0.585	0.884	0.821	0.794	0.794	0.868
25	glass5	0.938	0.695	0.795	0.988	0.973	0.745
26	poker-8_vs_6	0.532	0.521	0.499	0.531	0.529	0.625
27	poker-8-9_vs_5	0.517	0.536	0.536	0.557	0.515	0.499
28	poker-8-9_vs_6	0.498	0.518	0.499	0.617	0.615	0.68
29	poker-9_vs_7	0.535	0.587	0.541	0.596	0.689	0.646
30	vowel0	0.895	0.938	0.906	0.949	0.955	0.994
31	yeast-0-2-5-6_vs_3-7-8-9	0.709	0.673	0.670	0.668	0.688	0.714
32	yeast-0-2-5-7-9_vs_3-6-8	0.821	0.826	0.809	0.837	0.817	0.876
33	yeast-0-3-5-9_vs_7-8	0.571	0.554	0.592	0.670	0.640	0.649
34	yeast-0-5-6-7-9_vs_4	0.710	0.703	0.678	0.709	0.710	0.757
35	yeast-1_vs_7	0.557	0.613	0.602	0.589	0.615	0.631
36	yeast-1-2-8-9_vs_7	0.509	0.509	0.511	0.539	0.516	0.497
37	yeast-1-4-5-8_vs_7	0.503	0.511	0.492	0.521	0.527	0.527
38	yeast-2_vs_4	0.831	0.826	0.817	0.844	0.869	0.850
39	yeast-2_vs_8	0.696	0.742	0.797	0.696	0.702	0.673
40	yeast4	0.569	0.555	0.549	0.574	0.584	0.564
41	yeast5	0.759	0.769	0.761	0.747	0.765	0.848
42	yeast6	0.596	0.665	0.666	0.709	0.691	0.740
	Avg.ROC	**0.701**	**0.722**	**0.714**	**0.747**	**0.759**	**0.762**

result mentioned above also can be seen from Figs. 3 and 4. Moreover, we can find from Table 10 that HLazyDT and KLLazyDT perform statistically significantly better than the other trees at 95% confidence interval according to both the average ranks of AUC and G-mean.

Table 7 The G-means results across highly imbalanced data sets

NO	Data Sets	C4.5	HDDT	LazyDT	HLazyDT	KLLazyDT	SVM
1	cleveland-0_vs_4	0.486	0.526	0.366	0.409	0.671	0.617
2	ecoli-0-1_vs_2-3-5	0.463	0.666	0.691	0.841	0.863	0.824
3	ecoli-0-1_vs_5	0.784	0.783	0.846	0.907	0.930	0.876
4	ecoli-0-1-3-7_vs_2-6	0.194	0.534	0.338	0.391	0.527	0.340
5	ecoli-0-1-4-6_vs_5	0.641	0.673	0.683	0.815	0.877	0.783
6	ecoli-0-1-4-7_vs_2-3-5-6	0.655	0.689	0.679	0.728	0.810	0.775
7	ecoli-0-1-4-7_vs_5-6	0.645	0.636	0.645	0.830	0.868	0.838
8	ecoli-0-2-3-4_vs_5	0.764	0.795	0.804	0.870	0.841	0.875
9	ecoli-0-2-6-7_vs_3-5	0.779	0.824	0.800	0.777	0.771	0.681
10	ecoli-0-3-4_vs_5	0.816	0.855	0.816	0.902	0.905	0.905
11	ecoli-0-3-4-6_vs_5	0.804	0.792	0.802	0.847	0.875	0.845
12	ecoli-0-3-4-7_vs_5-6	0.725	0.789	0.778	0.838	0.830	0.775
13	ecoli-0-4-6_vs_5	0.749	0.811	0.842	0.837	0.855	0.852
14	ecoli-0-6-7_vs_3-5	0.786	0.850	0.839	0.843	0.830	0.845
15	ecoli-0-6-7_vs_5	0.869	0.862	0.869	0.881	0.873	0.856
16	ecoli4	0.853	0.707	0.884	0.884	0.898	0.884
17	glass-0-1-4-6_vs_2	0.093	0.332	0.093	0.345	0.550	0.492
18	glass-0-1-5_vs_2	0.304	0.406	0.304	0.369	0.368	0.508
19	glass-0-1-6_vs_2	0.245	0.396	0.244	0.364	0.509	0.531
20	glass-0-1-6_vs_5	0.736	0.621	0.621	0.872	0.917	0.794
21	glass-0-4_vs_5	0.994	0.794	0.994	0.723	0.704	1.00
22	glass-0-6_vs_5	0.979	0.978	0.862	0.915	0.885	0.941
23	glass2	0.159	0.260	0.270	0.304	0.301	0.362
24	glass4	0.341	0.864	0.785	0.760	0.800	0.855
25	glass5	0.929	0.795	0.595	0.978	0.968	0.536
26	poker-8_vs_6	0.115	0.100	0.000	0.115	0.114	0.379
27	poker-8-9_vs_5	0.090	0.178	0.178	0.267	0.090	0.000
28	poker-8-9_vs_6	0.000	0.099	0.000	0.393	0.369	0.521
29	poker-9_vs_7	0.138	0.277	0.138	0.198	0.470	0.421
30	vowel0	0.888	0.934	0.900	0.948	0.955	0.994
31	yeast-0-2-5-6_vs_3-7-8-9	0.656	0.598	0.592	0.601	0.631	0.651
32	yeast-0-2-5-7-9_vs_3-6-8	0.800	0.809	0.787	0.814	0.802	0.866
33	yeast-0-3-5-9_vs_7-8	0.364	0.296	0.399	0.610	0.574	0.555
34	yeast-0-5-6-7-9_vs_4	0.650	0.637	0.604	0.662	0.658	0.718
35	yeast-1_vs_7	0.277	0.440	0.414	0.421	0.560	0.534
36	yeast-1-2-8-9_vs_7	0.080	0.089	0.081	0.195	0.160	0.000
37	yeast-1-4-5-8_vs_7	0.080	0.098	0.000	0.161	0.192	0.115
38	yeast-2_vs_4	0.812	0.808	0.798	0.833	0.862	0.838
39	yeast-2_vs_8	0.553	0.614	0.753	0.552	0.566	0.514
40	yeast4	0.291	0.310	0.291	0.369	0.396	0.326
41	yeast5	0.710	0.727	0.677	0.696	0.723	0.823
42	yeast6	0.388	0.571	0.569	0.633	0.560	0.684
	Avg.G-means	**0.540**	**0.591**	**0.563**	**0.636**	**0.664**	**0.655**

(3) When we use SMOTE to preprocess the highly imbalanced data sets, it can improve the performances of all methods used in the experiment. As shown in Table 10, HDDT performs statistically significantly better than C4.5, LazyDT and KLLazyDT at 95% confidence interval according to both the average ranks of AUC and G-mean. In addition, Tables 8 and 9 also show that HLazyDT and KLLazyDT outperform

Table 8 The AUC results across highly imbalanced data sets with SMOTE

NO	Data Sets	C4.5	HDDT	LazyDT	HLazyDT	KLLazyDT	SVM
1	cleveland-0_vs_4	0.959	0.957	0.945	0.963	0.963	0.988
2	ecoli-0-1_vs_2-3-5	0.884	0.923	0.889	0.952	0.959	0.964
3	ecoli-0-1_vs_5	0.909	0.943	0.918	0.934	0.932	0.984
4	ecoli-0-1-3-7_vs_2-6	0.991	0.993	0.991	0.993	0.987	0.976
5	ecoli-0-1-4-6_vs_5	0.944	0.958	0.946	0.958	0.956	0.981
6	ecoli-0-1-4-7_vs_2-3-5-6	0.887	0.921	0.887	0.918	0.919	0.953
7	ecoli-0-1-4-7_vs_5-6	0.925	0.941	0.920	0.945	0.938	0.979
8	ecoli-0-2-3-4_vs_5	0.931	0.941	0.929	0.940	0.934	0.967
9	ecoli-0-2-6-7_vs_3-5	0.926	0.938	0.928	0.938	0.938	0.975
10	ecoli-0-3-4_vs_5	0.911	0.947	0.917	0.944	0.956	0.969
11	ecoli-0-3-4-6_vs_5	0.924	0.951	0.924	0.943	0.935	0.970
12	ecoli-0-3-4-7_vs_5-6	0.927	0.927	0.903	0.931	0.935	0.961
13	ecoli-0-4-6_vs_5	0.924	0.942	0.929	0.951	0.956	0.981
14	ecoli-0-6-7_vs_3-5	0.932	0.954	0.940	0.935	0.945	0.970
15	ecoli-0-6-7_vs_5	0.945	0.951	0.942	0.942	0.938	0.975
16	ecoli4	0.979	0.985	0.981	0.987	0.983	0.991
17	glass-0-1-4-6_vs_2	0.806	0.859	0.829	0.896	0.870	0.926
18	glass-0-1-5_vs_2	0.806	0.827	0.813	0.852	0.852	0.916
19	glass-0-1-6_vs_2	0.757	0.809	0.757	0.814	0.800	0.934
20	glass-0-1-6_vs_5	0.994	0.970	0.969	0.989	0.989	0.994
21	glass-0-4_vs_5	0.994	0.994	0.994	0.994	0.994	1.000
22	glass-0-6_vs_5	0.995	0.957	0.950	0.985	0.985	1.000
23	glass2	0.766	0.820	0.779	0.845	0.822	0.929
24	glass4	0.963	0.985	0.970	0.970	0.945	0.990
25	glass5	0.993	0.988	0.983	0.990	0.995	0.995
26	poker-8_vs_6	0.996	1.000	0.996	1.000	1.000	0.985
27	poker-8-9_vs_5	0.960	0.979	0.963	0.973	0.966	0.975
28	poker-8-9_vs_6	0.999	1.000	0.998	1.000	0.999	0.979
29	poker-9_vs_7	0.981	0.991	0.972	0.972	0.979	0.998
30	vowel0	0.988	0.994	0.982	0.982	0.990	0.998
31	yeast-0-2-5-6_vs_3-7-8-9	0.825	0.852	0.824	0.840	0.834	0.856
32	yeast-0-2-5-7-9_vs_3-6-8	0.918	0.934	0.922	0.927	0.925	0.939
33	yeast-0-3-5-9_vs_7-8	0.776	0.810	0.765	0.805	0.804	0.848
34	yeast-0-5-6-7-9_vs_4	0.852	0.875	0.842	0.873	0.864	0.919
35	yeast-1_vs_7	0.829	0.869	0.839	0.871	0.867	0.913
36	yeast-1-2-8-9_vs_7	0.860	0.909	0.865	0.891	0.879	0.919
37	yeast-1-4-5-8_vs_7	0.844	0.892	0.852	0.869	0.856	0.906
38	yeast-2_vs_4	0.938	0.950	0.937	0.945	0.944	0.974
39	yeast-2_vs_8	0.923	0.918	0.890	0.918	0.907	0.917
40	yeast4	0.907	0.926	0.909	0.913	0.916	0.950
41	yeast5	0.968	0.979	0.969	0.977	0.976	0.983
42	yeast6	0.931	0.945	0.930	0.940	0.931	0.946
	Avg.ROC	**0.916**	**0.933**	**0.914**	**0.933**	**0.930**	**0.959**

Table 9 The G-means results across highly imbalanced data sets with SMOTE

NO	Data Sets	C4.5	HDDT	LazyDT	HLazyDT	KLLazyDT	SVM
1	cleveland-0_vs_4	0.959	0.955	0.942	0.956	0.962	0.987
2	ecoli-0-1_vs_2-3-5	0.883	0.922	0.888	0.952	0.954	0.963
3	ecoli-0-1_vs_5	0.909	0.942	0.918	0.934	0.931	0.984
4	ecoli-0-1-3-7_vs_2-6	0.991	0.993	0.991	0.993	0.987	0.976
5	ecoli-0-1-4-6_vs_5	0.944	0.958	0.946	0.946	0.956	0.981
6	ecoli-0-1-4-7_vs_2-3-5-6	0.885	0.920	0.887	0.915	0.918	0.953
7	ecoli-0-1-4-7_vs_5-6	0.925	0.941	0.920	0.939	0.936	0.979
8	ecoli-0-2-3-4_vs_5	0.931	0.941	0.928	0.922	0.936	0.967
9	ecoli-0-2-6-7_vs_3-5	0.926	0.938	0.928	0.938	0.938	0.975
10	ecoli-0-3-4_vs_5	0.910	0.947	0.907	0.941	0.955	0.969
11	ecoli-0-3-4-6_vs_5	0.924	0.951	0.924	0.932	0.934	0.970
12	ecoli-0-3-4-7_vs_5-6	0.926	0.927	0.903	0.931	0.935	0.961
13	ecoli-0-4-6_vs_5	0.923	0.941	0.929	0.950	0.956	0.981
14	ecoli-0-6-7_vs_3-5	0.932	0.953	0.939	0.934	0.944	0.970
15	ecoli-0-6-7_vs_5	0.945	0.950	0.942	0.932	0.937	0.975
16	ecoli4	0.979	0.985	0.981	0.987	0.983	0.990
17	glass-0-1-4-6_vs_2	0.803	0.856	0.827	0.879	0.869	0.925
18	glass-0-1-5_vs_2	0.804	0.824	0.808	0.850	0.851	0.915
19	glass-0-1-6_vs_2	0.754	0.806	0.749	0.810	0.798	0.933
20	glass-0-1-6_vs_5	0.994	0.971	0.969	0.989	0.988	0.994
21	glass-0-4_vs_5	0.994	0.994	0.994	0.994	0.994	1.000
22	glass-0-6_vs_5	0.995	0.956	0.948	0.985	0.985	1.000
23	glass2	0.764	0.819	0.778	0.840	0.822	0.928
24	glass4	0.962	0.984	0.970	0.970	0.945	0.990
25	glass5	0.993	0.987	0.983	0.990	0.995	0.995
26	poker-8_vs_6	0.996	1.000	0.996	1.000	1.000	0.985
27	poker-8-9_vs_5	0.960	0.979	0.959	0.970	0.966	0.975
28	poker-8-9_vs_6	0.999	1.000	0.998	1.000	0.999	0.979
29	poker-9_vs_7	0.981	0.991	0.972	0.972	0.968	0.998
30	vowel0	0.988	0.994	0.982	0.982	0.980	0.998
31	yeast-0-2-5-6_vs_3-7-8-9	0.824	0.852	0.824	0.835	0.830	0.856
32	yeast-0-2-5-7-9_vs_3-6-8	0.918	0.934	0.922	0.926	0.915	0.939
33	yeast-0-3-5-9_vs_7-8	0.775	0.809	0.764	0.803	0.798	0.847
34	yeast-0-5-6-7-9_vs_4	0.852	0.874	0.841	0.864	0.863	0.919
35	yeast-1_vs_7	0.828	0.869	0.839	0.868	0.866	0.912
36	yeast-1-2-8-9_vs_7	0.859	0.909	0.865	0.881	0.876	0.919
37	yeast-1-4-5-8_vs_7	0.843	0.892	0.852	0.862	0.853	0.904
38	yeast-2_vs_4	0.938	0.950	0.936	0.945	0.929	0.974
39	yeast-2_vs_8	0.923	0.918	0.887	0.911	0.904	0.916
40	yeast4	0.907	0.926	0.908	0.913	0.916	0.950
41	yeast5	0.967	0.979	0.969	0.975	0.976	0.983
42	yeast6	0.931	0.945	0.930	0.932	0.930	0.946
	Avg.G-means	**0.915**	**0.933**	**0.913**	**0.930**	**0.928**	**0.959**

C4.5 and LazyDT except for SVM and HDDT. However, HLazyDT and KLLazyDT still achieve the comparable performance (neither significantly worse) than HDDT according to the average AUCs and G-means, which can be observed from Figs. 3 and 4 clearly.

Finally, we provide the following analyses and conclusions.

(1) It is obvious that the AUC and G-mean results not just associate with class imbalance ratio, which can be observed from Tables 4, 5, 6, 7, 8, and 9. As it turns out, data set complexity such as overlapping, lack of representative data, small disjuncts, and others is the primary determining factor of classification deterioration [7, 8].

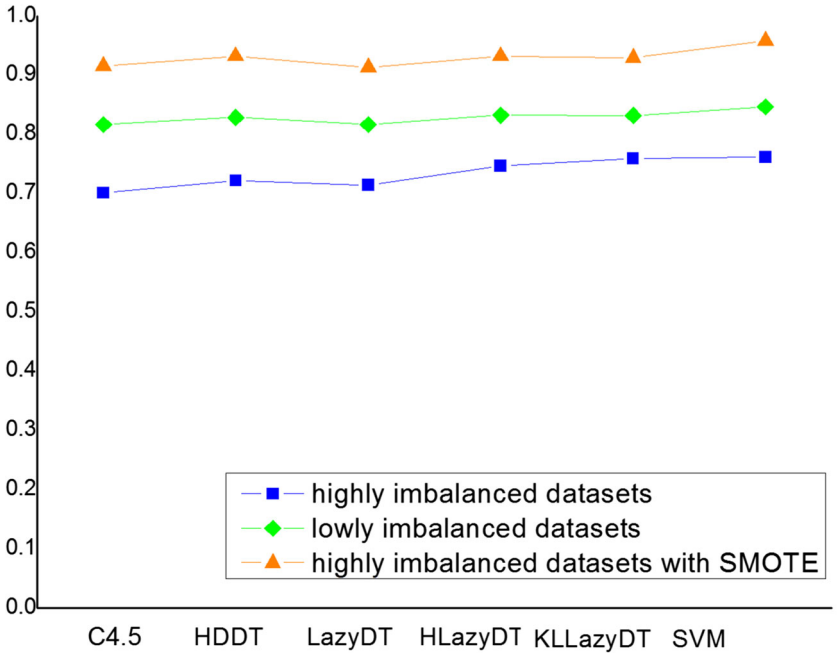

Fig. 3 Comparison of the average ROC results for each method across all data sets

(2) SMOTE preprocessing is very helpful to improve the performances of all methods used in this experiments when learning from highly imbalanced data sets. The main reason is that SMOTE creates artificial data based on the feature space similarities between the existing minority class instances and helps in broadening the decision region for a classifier and improving generalization [10].

(3) The information gain ratio used by C4.5 prefers to choose the feature with fewer values and is skew-sensitive. Thus, C4.5 can not always select the best split feature. LazyDT uses the information gain to choose the split feature, which is also skew-sensitive. Moreover, LazyDT does not take into account the value of the test instance on the candidate feature and this strategy may cause LazyDT to remount to the previous level node and return the most frequent class, which degrades the classification performance to some extent. On contrast, HDDT, HLazyDT and KLLazyDT that all use skew-insensitive splitting criteria can produce more branches to find more fine-grained differences in the datasets. Therefore, HDDT,

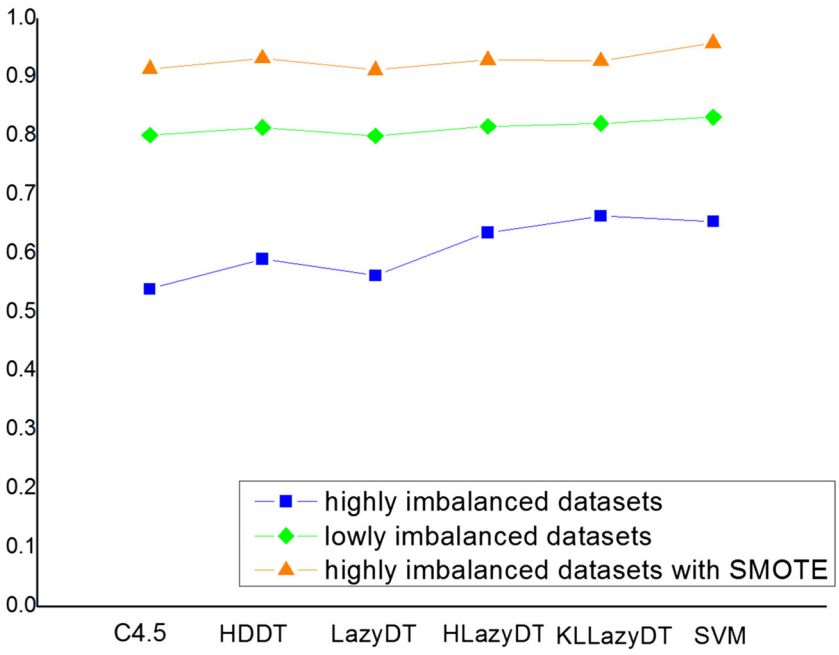

Fig. 4 Comparison of the average G-means results for each method across all data sets

Table 10 Average Friedman rankings and APVs of various decision trees using Holm's procedure in ROC and G-means

Datasets Type	Value Type	C4.5	HDDT	LazyDT	HLazyDT[a]	KLLazyDT[a]
Lowly Imbalanced Datasets	Ranking_R	3.477	2.591	3.863	2.568	**2.500**
	APV_R	0.121	1.000	**0.017** √	1.000	-
	Ranking_G	3.250	2.613	3.863	2.704	**2.568**
	APV_G	0.458	1.000	**0.027** √	1.000	-
Highly Imbalanced Datasets	Ranking_R	3.8577	3.155	3.583	2.333	**2.071**
	APV_R	**0.000** √	**0.003** √	**0.000** √	0.447	-
	Ranking_G	3.940	3.071	3.809	2.190	**1.988**
	APV_G	**0.000** √	**0.003** √	**0.000** √	0.555	-
Highly Imbalanced Datasets with SMOTE	Ranking_R	4.000	**1.869**	4.298	2.167	2.667
	APV_R	**0.000** √	-	**0.000** √	0.390	**0.041** √
	Ranking_G	3.845	**1.786**	4.179	2.417	2.774
	APV_G	**0.000** √	-	**0.000** √	0.067	**0.008** √

Items with a [a] are approaches modified by this paper

Ranking_R and Ranking_G denote average rankings on ROC and G-means respectively

APV_R and APV_G denote adjusted p-values using Holm's procedure in ROC and G-means respectively

HLazyDT and KLLazyDT can better distinguish between minority and majority class than LazyDT and C4.5. On the other hand, we note that HLazyDT and KLLazyDT significantly outperform HDDT across the highly imbalanced datasets. Since the number of the minority class instances in the highly imbalanced datasets is very rare, the values of the minority class instances on the some feature in the test set may not be include in the training set, which may cause C4.5, LazyDT and HDDT to remount to the previous level node and return the most frequent class, which degrades the classification performance to some extent. Furthermore, the minority class instances in the highly imbalanced datasets are indeed abundant in small disjuncts. Eager decision trees including C4.5 and HDDT tend to miss out on small disjuncts because they ignore certain instances when they occur infrequently, such as instances in very small disjuncts, while Lazy decision trees do not ignore any instances. This is the reason why we can still observe that the performances of LazyDT are superior to C4.5 according to both AUC and G-mean. Consequently, lazy learning methods are recommended when small disjuncts abound [44].

(4) In our experiment, we observe that the performances of SVM are superior to all kinds of decision trees according to the two evaluation measures, namely AUC and G-mean. This result is consistent with the conclusion in [45, 46]. It demonstrates that SVM can achieve better generalization than decision trees to deal with nonlinear problems, while decision trees can suffer from overfitting easily.

6 Conclusion

In this paper, we use Hellinger distance and K-L divergence as the split criteria to modify lazy decision trees i.e., HLazyDT and KLLazyDT to deal with the problem of imbalanced classification. An experiment framework using five-fold cross-validation is performed across a wide range of imbalanced data sets to investigate the effectiveness of our methods when comparing with the other methods including lazy decision tree, C4.5, Hellinger distance based decision tree and support vector machine according to ROC and G-mean evaluation measures respectively. Adjusted p-value Holm's post-hoc procedure of the Friendman test is used to determine the significance of the ranks across multiple data sets. Based on the experiment results, we demonstrate that it can improve the performances of LazyDT for imbalanced classification to use Hellinger distance and K-L divergence as the split criterion . According to the experimental results, we find that HLazyDT and KLLazyDT outperform the other decision trees for all imbalanced datasets used in our experiment. Especially, when using SMOTE to preprocess the highly imbalanced datasets, HLazyDT and KLLazyDT still achieve the comparable performance than HDDT. On the other hand, the performances of SVM are superior to all kinds of decision trees, which reveals that SVM can achieve better generalization than decision trees, while decision trees can suffer from overfitting easily. Finally, we recommend using HLazyDT and KLLazyDT to deal with imbalanced learning when limiting the approaches to decision trees. Meanwhile, we also consider that it is a good choice to apply HLazyDT and KLLazyDT to

imbalanced data stream learning such as medical data-mining and privacy preserving area. In future work, we wish to extend our methods to deal with continuous features and multi-class imbalanced problem. Additionally, ensemble methods have been proven to improve decision trees to solve imbalanced leaning effectively [47] and a new ensemble strategy based on optimizing decision directed acyclic graph is proposed to deal with multi-class classification problems [48], which inspires us to use ensemble methods to further improve the ability of HLazyDT and KLLazyDT to tackle imbalanced learning.

Acknowledgements We would like to acknowledge support for this project from China Postdoctoral Science Foundation (2016M600430), the National Social Science Foundation of China (16ZDA054), Jiangsu Provincial 333 Project (BRA2017396), Six Major Talents PeakProject of Jiangsu Province (XYDXXJS-CXTD-005) and Philosophy and social science in colleges and universities in Jiangsu Province outstanding innovation team (2015ZSTD006). The authors also would like to express our gratitude to the donors of the different data sets and the maintainers of the KEEL Data set Repository.

References

1. Quinlan JR (2014) C4.5: Programs for machine learning. Elsevier, Amsterdam
2. Friedman JH, Kohavi R, Yun Y (1996) Lazy decision trees. AAAI/IAAI 1:717–724
3. Bagallo G, Haussler D (1990) Boolean feature discovery in empirical learning. Mach Learn 5(1):71–99
4. Mahmoudi N, Duman E (2015) Detecting credit card fraud by modified fisher discriminant analysis. Expert Syst Appl 42(5):2510–2516
5. Khor KC, Ting CY, Phon-Amnuaisuk S (2014) The effectiveness of sampling methods for the imbalanced network intrusion detection data set. Recent Advances on Soft Computing and Data Mining. Springer, Cham, pp 613–622
6. Wan X, Liu J, Cheung WK (2014) Learning to improve medical decision making from imbalanced data without a priori cost. BMC medical informatics and decision making 14(1):111
7. He H, Garcia EA (2009) Learning from imbalanced data. IEEE Trans Knowl Data Eng 12(9):1263–1284
8. López V, Fernández A, García S (2014) An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. Inf Sci 250:113–141
9. Krawczyk B (2016) Learning from imbalanced data: Open challenges and future directions. Progress in Artificial Intelligence 5(4):221–232
10. Chawla NV, Bowyer KW, Hall LO (2002) SMOTE: Synthetic minority over-sampling technique. J Artif Intell Res 16:321–357
11. Han H, Wang WY, Mao BH (2005) Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning. International Conference on Intelligent Computing. Springer, Berlin, Heidelberg, pp 878–887
12. He H, Bai Y, Garcia EA (2008) ADASYN: Adaptive Synthetic sampling approach for imbalanced learning. IEEE International Joint Conference on Neural Networks, pp 1322–1328
13. Hu S, Liang Y, Ma L (2009) MSMOTE: Improving classification performance when training data is imbalanced. IEEE 2nd International Workshop on Computer Science and Engineering, pp 13–17
14. Barua S, Islam MM, Yao X (2014) MWMOTE–Majority weighted minority oversampling technique for imbalanced data set learning. IEEE Trans Knowl Data Eng 26(2):405–425
15. Zhou P, Hu X, Li P (2017) Online feature selection for high-dimensional class-imbalanced data. Knowl-Based Syst 136:187–199
16. Wu G, Chang EY (2005) KBA: Kernel Boundary alignment considering imbalanced data distribution. IEEE Trans Knowl Data Eng 17(6):786–795
17. Xu Y (2017) Maximum margin of twin spheres support vector machine for imbalanced data classification. IEEE Trans Cybern 47(6):1540–1550
18. Xu Y, Wang Q, Pang X (2018) Maximum margin of twin spheres machine with pinball loss for imbalanced data classification. Appl Intell 48(1):23–34
19. Wang S, Yao X (2009) Diversity analysis on imbalanced data sets by using ensemble models. IEEE Symposium on Computational Intelligence and Data Mining, pp 324–331
20. Chawla NV, Lazarevic A, Hall LO (2003) SMOTEBOost: Improving prediction of the minority class in boosting. European conference on principles of data mining and knowledge discovery. Springer, Berlin, Heidelberg, pp 107–119
21. Liu XY, Wu J, Zhou ZH (2009) Exploratory undersampling for class-imbalance learning. IEEE Trans Syst Man Cybern B (Cybernetics) 39(2):539–550
22. Longadge R, Dongre S (2013) Class imbalance problem in data mining review. International Journal of Computer Science and Network 1305:1707
23. Zhang Z, Krawczyk B, Garcìa S (2016) Empowering one-vs-one decomposition with ensemble learning for multi-class imbalanced data. Knowl-Based Syst 106:251–263
24. Cieslak DA, Chawla NV (2008) Learning decision trees for unbalanced data. Joint European Conference on Machine Learning and Knowledge Discovery in Databases. Springer, Berlin, Heidelberg, pp 241–256
25. Cieslak DA, Hoens TR, Chawla NV (2012) Hellinger distance decision trees are robust and skew-insensitive. Data Min Knowl Disc 24(1):136–158
26. Hoens TR, Qian Q, Chawla NV (2012) Building decision trees for the multi-class imbalance problem. Pacific-asia Conference on Knowledge Discovery and Data Mining. Springer, Berlin, Heidelberg, pp 122–134
27. Lyon RJ, Brooke JM, Knowles JD (2014) Hellinger distance trees for imbalanced streams. IEEE International Conference on Pattern Recognition, pp 1969–1974
28. Chawla NV, Cieslak DA, Hall LO (2008) Automatically countering imbalance and its empirical relationship to cost. Data Min Knowl Disc 17(2):225–252
29. Zhang H (2012) Lazy decision tree method for distributed privacy preserving data mining. International Journal of Advancements in Computing Technology 4(14):458–465
30. Quinlan JR (1996) Bagging, boosting, and c4.5. AAAI/IAAI 1:725–730
31. Dietterich TG (2000) An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. Mach Learn 40(2):139–157
32. Fern XZ, Brodley CE (2003) Boosting lazy decision trees. In: Proceedings of the 20th International Conference on Machine Learning ICML, pp 178–185
33. Guillame-Bert M, Dubrawski A (2016) Batched Lazy Decision Trees. arXiv:1603.02578
34. Rao CR (1995) A review of canonical coordinates and an alternative to correspondence analysis using Hellinger distance. Qű,estiió 19(1):23–63
35. Kullback S, Leibler RA (1951) On information and sufficiency. Ann Math Stat 22(1):79–86

36. Christopher DM, Prabhakar R, Hinrich S (2008) Introduction to information retrieval. An Introduction To Information Retrieval 151(177):5
37. Triguero I, González S, Moyano JM (2017) KEEL 3.0: An open source software for multi-stage analysis in data mining. International Journal of Computational Intelligence Systems 10(1):1238–1249
38. Chawla NV (2003) C4.5 and imbalanced data sets: investigating the effect of sampling method, probabilistic estimate, and decision tree structure. In: Proceedings of the ICML, 3:66
39. Cortes C, Vapnik V (1995) Support vector machine. Mach Learn 20(3):273–297
40. Raeder T, Forman G, Chawla NV (2012) Learning from imbalanced data: evaluation matters. Data mining: Foundations and intelligent paradigms. Springer, Berlin, Heidelberg, pp 315–331
41. Hand DJ, Till RJ (2001) A simple generalisation of the area under the ROC curve for multiple class classification problems. Mach Learn 45(2):171–186
42. Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7(Jan):1–30
43. García S, Herrera F (2008) An extension on "statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons. J Mach Learn Res 9(Dec):2677–2694
44. Van Den Bosch A, Weijters A, Van Den Herik HJ (1997) When small disjuncts abound, try lazy learning: A case study. Proceedings of the Seventh Belgian-Dutch Conference on Machine Learning, pp 109–118
45. Caruana R, Niculescu-Mizil A (2006) An empirical comparison of supervised learning algorithms. ACM Proceedings of the 23rd international conference on Machine learning, pp 161–168
46. Fernández-Delgado M, Cernadas E, Barro S (2014) Do we need hundreds of classifiers to solve real world classification problems. J Mach Learn Res 15(1):3133–3181
47. Banfield RE, Hall LO, Bowyer KW (2007) A comparison of decision tree ensemble creation techniques. IEEE Trans Pattern Anal Mach Intell 29(1):173–180
48. Zhou L, Fujita H (2017) Posterior probability based ensemble strategy using optimizing decision directed acyclic graph for multi-class classification. Inf Sci 400:142–156