Soft Comput
DOI 10.1007/s00500-014-1291-z

FOCUS

To combat multi-class imbalanced problems by means of over-sampling and boosting techniques

Lida Abdi · Sattar Hashemi

Abstract Imbalanced problems are quite pervasive in many real-world applications. In imbalanced distributions, a class or some classes of data, called minority class(es), is/are under-represented compared to other classes. This skewness in the data underlying distribution causes many difficulties for typical machine learning algorithms. The notion becomes even more complicated when machine learning algorithms are to combat multi-class imbalanced problems. The presented solutions for tackling the issues arising from imbalanced distributions, generally fall into two main categories: data-oriented methods and model-based algorithms. Focusing on the latter, this paper suggests an elegant blend of boosting and over-sampling paradigms, which is called MDOBoost, to bring considerable benefits to the learning ability of multi-class imbalanced data sets. The over-sampling technique introduced and adopted in this paper, Mahalanobis distance-based over-sampling technique (MDO in short), is delicately incorporated into boosting algorithm. In fact, the minority classes are over-sampled via MDO technique in such a way that they almost preserve the original minority class characteristics. MDO, in comparison with the popular method in this field, SMOTE, generates more similar minority class examples to original class samples. Moreover, the broader representation of minority class examples is provided via MDO, and this, in turn, causes the classifier to build larger decision regions. MDOBoost increases the generalization ability of a classifier, since it indicates better results with pruned version of C4.5 classifier; unlike other over-sampling/boosting procedures, which have difficulties with pruned version of C4.5. MDOBoost is applied to real-world multi-class imbalanced benchmarks and its performance is then compared with several data-level and model-based algorithms. The empirical results and theoretical analyses reveal that MDOBoost offers superior advantages compared to popular class decomposition and over-sampling techniques in terms of MAUC, G-mean, and minority class recall.

Keywords Multi-class imbalance · Over-sampling · Mahalanobis distance · Boosting algorithm · Class decomposition techniques

Communicated by E. Lughofer.

L. Abdi · S. Hashemi (✉)
CSE and IT Department, Shiraz University, Engineering Campus Number 2, Mollasadra Ave., Shiraz, Iran
e-mail: s-hashemi@shirazu.ac.ir

L. Abdi
e-mail: l-abdi@cse.shirazu.ac.ir

1 Introduction

In recent years, with the explosive growth of available data in many scientific fields, there is a need for more robust and efficient learning techniques. Among these huge data, the imbalanced distributions are very pervasive. A data set with uneven number of instances for different classes is called an imbalance data set. This skewness in data underlying distribution poses many issues to typical machine learning algorithms. Correctly classifying the rarest or minority class instances is of great importance in imbalanced learning. Consequently, obtaining a classifier with high accuracy for the minority class without severely jeopardizing the accuracy of the majority class is a key point (He and Garcia 2009).

In many real-world applications such as protein fold and weld flaw classifications (Zhao et al. 2008; Liao 2008), there is a presence of multi-class imbalanced distributions. Processing these problems imposes many new challenges and issues which have not been seen in two-class cases.

Zhou and Liu (2006) showed that cost-sensitive learning with multi-class tasks is more difficult than with two-class ones and a higher degree of class imbalance may increase the difficulty. They also revealed that almost all techniques, which are effective on two-class tasks, are ineffective or may cause negative effects on multi-class problems.

Proposed solutions in imbalanced problems are at data level and algorithmic level. Data-level solutions are a kind of pre-process tasks which is applied to balance the skewed distributions directly. These solutions which can be used simply are divided into over-sampling and under-sampling techniques. In over-sampling, the number of instances in minority classes increases to reach a desired level of balance. These synthetic samples, which are added to the original data set, may cause the algorithm to over-fit or over-generalize. SMOTE over-sampling technique is among the most popular over-sampling methods in the literature (Chawla et al. 2002).

On the other hand, under-sampling solutions eliminate some of the majority class instances and in this way, they can help ease the learning process. But these methods which remove some instances of the majority classes may cause lack of useful information and mislead the algorithm.

Algorithmic or model-based solutions try to change the inductive bias of the learner in such a way that the learning process improves remarkably. Feature selection techniques (Alibeigi et al. 2012), cost-sensitive methods, one-class learning [also known as novelty detection and recognition-based methodology (Chawla et al. 2004)] and ensemble learning algorithms are among these techniques. In feature selection techniques, the most informative and the most discriminative features are selected through the feature selection process. Some popular techniques of feature selection in imbalanced tasks are Feature Assessment by Sliding Thresholds (FAST) (Chen and Wasikowski 2008), Signal-to-Noise Correlation Coefficient (S2N) (Golub et al. 1999), Pearson Correlation Coefficient (PCC), etc. In cost-sensitive learning algorithms, a higher cost of misclassification is considered for rare examples. A cost matrix which determines the cost of misclassification for each of the classes plays the essential role in cost-sensitive scenarios; unfortunately, such cost information is not available in most applications.

One-class learning techniques address class imbalance problem by modifying the training mechanism with the more direct goal of better accuracy on the minority classes. Instead of differentiating examples of one class from the others, these methods learn a model using mainly or only a single class of examples. Some related researches of this category include Zhuang and Dai (2006); Trujillo-Ortiz et al. (2007). One-class learning, also, indicates powerful performance in dealing with extremely imbalanced data sets with high-dimensional feature space (Raskutti and Kowalczyk 2004; Lee and Cho 2006).

Ensemble learning algorithms indicate great success in many real-world imbalanced problems. The key idea is to employ several base classifiers instead of one and combine their predictions as a “committee” to reach better accuracy. For theoretical reasons (Dietterich 2000; Polikar 2006), every single learning method has limitations and may perform differently due to insufficient data. Averaging many classifiers can reduce the overall risk of poor predictions (Perrone and Cooper 1993). Besides, ensembles can avoid local optima problem by running local search from different aspects, where a better approximation to the true function is expected. Moreover, each individual of the ensemble can learn from one of the smaller and simpler partitions of data space. Finally, the aggregation of these base classifiers can represent the whole problem better. With these mentioned advantages, ensemble learning algorithms have considerable success in various fields like imbalance learning problems.

Among limited solutions in multi-class imbalance learning in the literature, class decomposition techniques gained a lot of attention, recently. Class decomposition is referred to converting a multi-class problem into several two-class problems which are easier to handle. One against one (OAO) (Hastie and Tibshirani 1998) and one against all (OAA) (Rifkin and Klautau 2004) are among these techniques. In OAO, if there are c classes of data, each of them is considered against every one of the other classes and a two-class classifier is then trained on them. As a result, $c(c-1)/2$ two-class classifiers should be trained. Therefore, if the number of classes is too high the training time would be very long. In OAA case, each of c classes is considered as a positive class and all other classes are considered as negative; a two-class classifier is then trained on this new two-class train data. In this case, if the original training data are imbalanced already, the problem may get even worse. Consequently, class decomposition techniques do not seem to be good solutions in processing multi-class imbalanced problems.

Combination of ensemble learning algorithms and sampling techniques has been used widely in the literature, such as SMOTEBoost (Chawla et al. 2003) and RUSBoost (Seiffert et al. 2010). For instance, SMOTEBoost populates the considered training examples via SMOTE over-sampling techniques in each learning iteration of AdaBoost.M2 algorithm (Freund and Schapire 1996). In fact, this approach creates synthetic samples from the minority class instances; indirectly changing the updating weights and compensating for skewed distribution.

AdaBoost is a popular technique in imbalance learning and can be simply adapted for improving the classification performance due to the following advantages (Weiss 2004; Kotsiantis et al. 2006): (a) it is flexible and can easily be combined with and adapt to other algorithms; (b) it is efficient, because the optimal class distribution and representative instances are searched automatically during the learning process without any extra computational cost; (c) it is accurate, since no majority class samples are ignored (Sun et al. 2007). These appealing properties of AdaBoost encourage it to be integrated with sampling techniques (Seiffert et al. 2008). Boosting and over-sampling solely may not be ideal cases for learning from imbalanced data sets, especially multi-class ones; which are harder to be learned. Combining both of these techniques as a new approach for multi-class imbalanced problems seems effective and desired.

In this paper, we propose a novel technique, called MDOBoost, which applies over-sampling and boosting procedures. MDO, a new over-sampling technique, is inspired by Mahalanobis distance (Mahalanobis 1936) of a random vector from its class mean. In other words, the method MDOBoost employs both MDO over-sampling technique and AdaBoost.M2 algorithm to improve the classification ability in the presence of multi-class imbalanced data sets.

MDO over-sampling technique generates new synthetic samples which have the same Mahalanobis distance from the considered class mean. Mahalanobis distance takes into account the class samples variability and the correlation of the instances of a class. MDO tries to generate samples toward the variability of each minority class examples, and in this way the generated samples almost resemble the properties and characteristics of the original minority class instances. SMOTE and the other proposed over-sampling techniques in the literature deal with all the class examples as equal.

We compare our algorithms with other five popular algorithmic- and data-level techniques which are used widely in multi-class imbalanced literature. Four different classifiers with 14 multi-class UCI (Frank and Asuncion 2010) and KEEL (Alcal et al. 2010) imbalanced data sets are used in our evaluations. MDO over-sampling technique generates synthetic examples which preserve almost the same characteristic as original minority class samples. Moreover, combing the MDO and boosting procedure improves the generalization ability of classifiers significantly. The experiments reveal that MDOBoost performs significantly better than over-sampling and class decomposition techniques in MAUC, G-mean, and minority class recall.

The rest of the paper is organized as follows: Sect. 2 introduces the popular research progresses in class imbalance problems in the literature. Section 3 explains our proposed methods in detail. Our experimental settings and analyses of the results are provided in Sect. 4. Section 5 concludes the paper by a conclusion part and presents our future work.

2 Related work

In this section, several data-oriented and model-oriented approaches in the area of class imbalance problems are presented. Random over-sampling and random under-sampling are two simple data-level or re-sampling solutions which are used widely in many applications. In random over-sampling, randomly chosen instances of the minority class are added to the original data set. It is a simple method, while it has been argued that exact copies of minority class instances can lead the algorithm to over-fit (Weiss and Provost 2001). On the other hand, random under-sampling removes the desired number of majority class instances randomly from the data set. This method may discard useful information pertain to the majority class for better learning (He and Garcia 2009; Weiss and Provost 2001).

Synthetic minority over-sampling technique, SMOTE, which was proposed by Chawla et al. (2002), is a rounded over-sampling technique. In this method, the synthetic samples are generated based on the similarities between original minority class instances in the feature space. SMOTE randomly selects one of the K-nearest neighbours of a regarding minority class sample x_i, denoted as $\hat{x}_i$, and calculates the difference between them. This difference is then multiplied by a random number δ in range $[0, 1]$ and is added to the original considered sample. In other words, synthetic samples are created along the line segment between two specific samples of minority class, i.e. $(x_i, \hat{x}_i)$. SMOTE is reported to have the over-generalization problem (Wang and Japkowicz 2004). Also, it generates synthetic samples regardless of the majority class; consequently, the possibility of over-lapping between classes is very high (Prati et al. 2004).

Borderline SMOTE (Han et al. 2005), which is a modification of the SMOTE, assumes that instances near the classification boundaries (i.e. borderline samples) are more likely to be misclassified. So these examples are more important. Only borderline examples are considered to populate new data by applying SMOTE. If the most neighbours of a minority class sample are of the majority class, the algorithm considers it as a borderline example in "danger".

Adaptive Synthetic Sampling (ADA-SYN) (He et al. 2008) is a kind of adaptive sampling technique that has been proposed to overcome the SMOTE difficulty. This method creates different amount of synthetic samples according to their distribution. It first calculates the number of new instances that need to be generated for the minority class: $G = (|S_{\text{major}}| - |S_{\text{minor}}|) \times \beta$, where $\beta \in [0, 1]$ is a parameter to control the desired level of balance. For each minority class sample $x_i \in S_{\text{minor}}$, the K-nearest neighbours are calculated and the ratio of Γ_i is defined as $\frac{\Delta_i/K}{Z}, \quad i = 1, \ldots, |S_{\text{minor}}|$. In which, Δ_i is the number of examples in the K-nearest neighbours of x_i that belong to majority class S_{major} and Z is a normalization factor. The number of synthetic examples for each $x_i \in S_{\text{minor}}$ is computed as follows: $g_i = \Gamma_i \times G$. Finally, for each $x_i \in |S_{\text{minor}}|$ ADA-SYN generates g_i synthetic data with SMOTE algorithm. In fact, this technique uses the Γ density distribution as a criterion to generate synthetic examples adaptively.

Tomek links which was proposed by Tomek (1976) is an under-sampling method which eliminates the examples in a Tomek link which belongs to the majority class. Tomek links between two pair x_i and x_j, which belong to minority and majority class, respectively, is defined as follows (He and Garcia 2009). The (x_i, x_j) is denoted as a Tomek link if there is no example x_k, such that $d(x_i, x_k) < d(x_i, x_j)$ or $d(x_j, x_k) < d(x_i, x_j)$ in which $d(x_i, x_j)$ stands for the distance between two samples x_i and x_j. Either one in the link is noise or both are near the boundary. This method can clean up over-lapping between classes and establish well-defined classification rules.

Data cleaning techniques are effectively applied in various applications. Condensed nearest neighbour rule (CNN) (Kubat and Matwin 1997) is a data cleaning method that aims to find samples that are far from the decision boundaries. The idea is to find a consistent data subset $\hat{Z} \subseteq Z$, where all the examples in Z can be classified correctly using One-NN classifier in $\hat{Z}$.

One-sided selection (OSS) method (Kubat and Matwin 1997) selects a representative subset of majority class examples E and creates a new preliminary set N with all minority samples. The set N is then refined using a data cleaning technique. Wilson's edited nearest neighbour rule (ENN) (Wilson 1972) removes the examples whose class labels differ from the class of at least two of its three nearest neighbours.

AdaBoost algorithm is very popular in class imbalance learning, since it can be easily adapted to other techniques to improve the classification performance; especially, when it is integrated with re-sampling techniques. Data generation methods are often used to broaden the decision region of the minority class. SMOTEBoost modifies the distribution of the training examples via SMOTE in each round of boosting and enhances the probability of selecting the difficult rare cases (Rifkin and Klautau 2004). Meanwhile, boosting prevents from sacrificing accuracy over the entire data set and is believed to produce higher diversity within the ensemble. The parameter setting in SMOTE is crucial, which may cause over-generalization.

RUSBoost (Seiffert et al. 2010) is another approach of sampling and boosting procedures which provides a simpler and faster alternative to SMOTEBoost. RUSBoost is based on the AdaBoost.M2 algorithm. Prior to constructing the weak hypothesis during each round of boosting, random under-sampling is applied to the training data to achieve a more balanced training data set.

Among algorithmic level solutions, class decomposition methods have been noticed greatly by many researchers. Tan et al. (2003) used OAA and OAO schemes to convert a multi-class problem into several two-class tasks and then built rule-based learners to improve the coverage of minority class instances. Zhao et al. (2008) used SMOTE, OAA, and under-sampling to overcome the problem of multi-class imbalance. Liao (2008) studies several over-sampling and under-sampling techniques with OAA for weld flaw classification problem. Fernandez et al. (2010) combine both OAO and SMOTE in their algorithm. Chen et al. (2006) propose an algorithm which uses OAA for multi-class tasks and then applies some sampling methods which decompose each binary problem further so as to re-balance the train data.

Wang and Yao (2012) used the idea of negative correlation learning in their algorithm. An ambiguity or diversity term which assesses the disagreement degree of the classification within the ensemble is used in this algorithm. Applying this ambiguity term causes the algorithm to generate more different and diverse ensemble members during boosting iteration of AdaBoost algorithm.

Sun et al. (2006) develop a cost-sensitive boosting algorithm to improve the classification performance of imbalanced data sets with multiple classes. They apply a Genetic Algorithm (GA) to search optimum cost setup of each class. Two different fitness functions, G-mean and F-measure, are used in their algorithm. The acquired cost vector is then integrated in a cost-sensitive version of AdaBoost.M1 namely AdaC2 (Sun et al. 2005, 2007). Empirical evaluations indicated that their proposed boosting algorithm improves the classification performance of multi-class imbalanced data sets, significantly.

Sun et al.'s method (2006) and AdaBoost.NC algorithm (Wang and Yao 2012) consider the multi-class task as a whole, without applying any class decomposition schemes, like our proposed method. Due to the difficulties associated with class decomposition techniques, considering the multi-class imbalanced tasks as a unit problem would be very efficient.

3 The proposed MDOBoost algorithm

In the following section, our proposed MDO over-sampling technique together with MDOBoost algorithm is described in detail.

3.1 MDO: Mahalanobis distance-based over-sampling technique

The proposed over-sampling technique is based on Mahalanobis distance (Mahalanobis 1936). The idea is that "we can generate synthetic instances which have the same Mahalanobis distance from their corresponding class mean". Figure 1 indicates a simulated two-dimensional data. The circle and ellipse contours represent equal Euclidean and Mahalanobis distances to the centre point of the considered class, respectively. In calculation of Mahalanobis distance, the correlations of the data set are also taken into account and in this way, Mahalanobis distance differs from Euclidean distance.

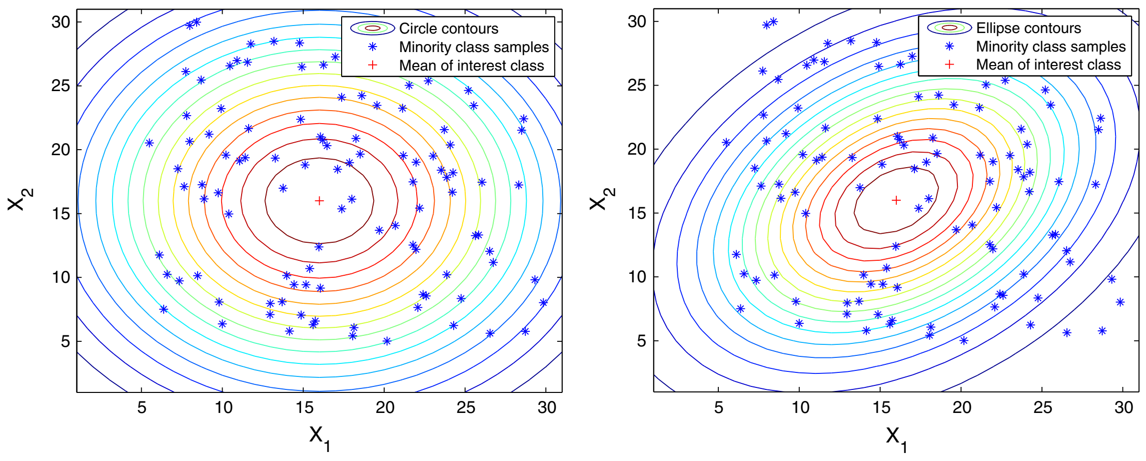

Fig. 1 Simulated data for two variables x_1, x_2 with the circle (*left*) and ellipse contours (*right*), representing equal Euclidean and Mahalanobis distances toward the centre mean, respectively

Definition 1 (*Euclidean distance*) The Euclidean distance between two points $x = (x_1, \ldots, x_N)^T$ and $y = (y_1, \ldots, y_N)^T$ in the N-dimensional space $\Re^N$ is defined as:

$$d_E(x, y) = \sqrt{(x_1 - y_1)^2 + \cdots + (x_N - y_N)^2} = \sqrt{(x - y)^T (x - y)}$$

It follows that all points with the same Euclidean distance of the origin $\|x\|_2 = c$ satisfy $x_1^2 + \cdots + x_N^2 = c^2$ which is the equation of a spheroid. In other words, all components of a considered point x contribute equally to the Euclidean distance of x from the centre. However, in statistics when we want to determine the distance of a variable from its origin, we prefer a distance which for each of the components takes the variability of that variable into account. Components with high variability should receive higher weight than components with low variability. This can be obtained by rescaling the components. Denote $u = (\frac{x_1}{s_1}, \ldots, \frac{x_N}{s_N})$ and $v = (\frac{y_1}{s_1}, \ldots, \frac{y_N}{s_N})$, then define the distance between x and y as

$$d(x, y) = d_E(u, v) = \sqrt{\left(\frac{x_1 - y_1}{s_1}\right)^2 + \cdots + \left(\frac{x_N - y_N}{s_N}\right)^2} = \sqrt{(x - y)^T D^{-1} (x - y)}$$

where $D = \text{diag}(s_1^2, \ldots, s_N^2)$.

Now the norm of x equals $\|x\| = d(x, 0) = d_E(u, 0) = \|u\|_2 = \sqrt{\left(\frac{x_1}{s_1}\right)^2 + \cdots + \left(\frac{x_N}{s_N}\right)^2} = \sqrt{x^T D^{-1} x}$ and all points with the same distance of the origin $\|x\| = c$ satisfy

$$\left(\frac{x_1}{s_1}\right)^2 + \cdots + \left(\frac{x_N}{s_N}\right)^2 = c^2.$$

which is the equation of an ellipsoid centred at the origin with principal axes equal to the coordinate axes. We want to take the **correlation** between variables into account when computing statistical distances. Correlation means that there are associations between the variables. Therefore, we want the axes of ellipsoid to reflect this correlation. This is obtained by allowing the axes of the ellipsoid at a constant distance to rotate. This yields the following general form for the statistical distance of two points:

Definition 2 (*Mahalanobis distance*) The statistical or Mahalanobis distance between two points $x = (x_1, \ldots, x_N)^T$ and $y = (y_1, \ldots, y_N)^T$ in the N-dimensional space $\Re^N$ is defined as:

$$d_S(x, y) = \sqrt{(x - y)^T S^{-1} (x - y)}$$

where S denotes the covariance matrix.

Points with the same distance of the origin $\|x\|_S = c$ satisfy $x^T S^{-1} x = c^2$ which is the general equation of an ellipsoid centred at the origin. In general, the centre of the observations will differ from the origin and we will be interested in the distance of an observation from its centre μ given by

$$d_S(x, \mu) = \sqrt{(x - \mu)^T S^{-1} (x - \mu)}$$

If the covariance matrix is the identity matrix, the Mahalanobis distance reduces to the Euclidean distance.

The minority class is over-sampled by taking each minority class sample and generating synthetic examples along the ellipse contour which passes through the considered minority class sample point. In this way, the synthetic samples are generated toward the variation of the corresponding class and almost preserve the same characteristics as the original class samples.

An ellipse, in mathematics, is a plane curve which results from the intersection of a cone by a plane such that it produces a closed curve. An ellipse in two dimensions with the centre point (h, k) is a closed curve which satisfies the Eq. 1:

$$\frac{(x^2 - h)}{a^2} + \frac{(y^2 - k)}{b^2} = 1. \tag{1}$$

In general, for d-dimensional ellipse with origin of the Cartesian coordinates as its centre point, we can generalize Eq. 1 as follows:

$$\frac{(x_1^2)}{a_1^2} + \frac{(x_2^2)}{a_2^2} + \cdots + \frac{(x_d^2)}{a_d^2} = 1. \tag{2}$$

In our over-sampling approach, d is referred to the dimension of our sample points, a_is, which are the denominators of the ellipse equation, can be considered as variations of the data in each dimension. In other words, we can substitute these coefficients for a new parameter, α times V_i, in which α is a constant parameter for all dimensions and V_i is the corresponding variance of each dimension; V has the same dimension as the data. Consequently, we can rewrite Eq. 2 as follows:

$$\frac{(x_1^2)}{\alpha \cdot V_1} + \frac{(x_2^2)}{\alpha \cdot V_2} + \cdots + \frac{(x_i^2)}{\alpha \cdot V_i} + \cdots + \frac{(x_d^2)}{\alpha \cdot V_d} = 1. \tag{3}$$

In Eq. 3, we substitute all "V" coefficients with variance of data in each dimension to parametrize the ellipse equation and express each radius of the ellipse with the corresponding variance. To reach variance of each dimension, we use eigenvalue decomposition, make the data uncorrelated and get the required variances. Consequently, we can simply generate new synthetic examples via Eq. 3. We used the concept of Mahalanobis distance instead of Euclidean distance to over-sample the minority class instances, because we want to create synthetic samples in such a way that they almost preserve the same characteristic as the original minority class samples. And this goal will be reached when we generate new samples toward the variation of the corresponding class instances. We use Eq. 3 to generate new synthetic examples for the minority classes. Algorithm 1 is the pseudo-code for MDO over-sampling method; also the block diagram of the algorithm is provided in Fig. 2.

Let x_i be a point in the feature space X and y_i be the corresponding class label in a set of class labels Y. Let n_{maj} be the number of majority class instances. In step 1, the samples of class i are determined for over-sampling. In steps 2–4 of the MDO algorithm, each sample set of class i is normalized to have zero mean. In step 5, the principal components of the class i instances are computed via eigenvalue decomposition. In step 6, the instances of the considered class are transformed to Principal Components (PC) space. The corresponding coefficients, which are the variances of the data in each dimension, are computed in step 7. These coefficients together with the α parameter form the denominators of the ellipse Eq. 3. Over-sampling rate parameter Orate$_i$, which is

Algorithm 1 MDO over-sampling technique.

```
Input: Data set $S = \{(x_1, y_1), ..., (x_m, y_m)\} x_i \in X; y_i \in Y = \{1, ..., c\}$; Number of majority class instances $n_{maj}$.
Output: A balanced set of data.

$t$ = Number of minority classes;
$S_{new} = \emptyset$ ;
for $i = 1$ to $t$ do
  $S_i$ = instances of class $i$;
  $n_i$ = number of instances in class $i$;
  $\mu_i = \frac{1}{n_i} \sum_{j=1}^{n_i} x_i^j$; */class mean.
  $Z_i$ = replace each $x_i^j$ with $x_i^j - \mu_i$;
  Find the principal components of $Z_i$ using eigenvalue decomposition;
  $T_i$ = transform $Z_i$ to corresponding PC space and make them uncorrelated;
  $V$ = diagonal(covariance($T_i$)); */A vector of coefficients.
  $Orate_i = n_{maj} - n_i$;
  $S_{temp}$ = MDO-oversampling($T_i, \mu_i, V, Orate_i$);
  Transform the newly generated samples $S_{temp}$ to the original space;
  Add the class mean $\mu_i$ to the new samples;
  Add new samples to $S_{new}$;
end for
Return balanced set of data;

MDO-oversampling($T_i, \mu_i, V, Orate_i$)
$I$ = number of instances in $T_i$;
if $I > Orate_i$ then
  $N = 1$;
  $I = Orate_i$;
else
  $N = (int)(Orate_i / I)$; */ Round up the division result.
end if
$Num_{attrs}$ = number of attributes;
for $l = 1$ to $I$ do
  $x_{temp} = T_i(l)$;
  $x$ = square $x_{temp}$;
  Compute $\alpha$ from ellipse equation 3, with $V$ and $x$;
  $AlphaV = \alpha \times V$; */ A vector results from $\alpha \times V$, which forms the denominators of equation 3.
  for $m = 1$ to $N$ do
    Set $s = 0$;
    for $p = 1$ to $(Num_{attrs} - 1)$ do
      $r$ = choose a random number from
      $[-\sqrt{AlphaV(p)}, \sqrt{AlphaV(p)}]$ for the $p$th dimension of the new sample;
      $s = s + (r^2 / AlphaV(p))$;
    end for
    Solve ellipse equation 3, with $s$ and $AlphaV(last)$ and find the last feature value, $LastFeaVal$;
    Set last feature value randomly from
                    $\{LastFeaVal, -LastFeaVal\}$;
    Add newly generated sample $x_{new}$ to $S_{temp}$;
  end for
end for
If the number of synthetic instances is greater than $Orate_i$, then select $Orate_i$ number of them randomly;
Return set of synthetic samples $S_{temp}$;
```

the difference between the number of majority and minority class instances, is computed in step 8. Synthetic samples with MDO method are generated in step 9. In steps 10–12,

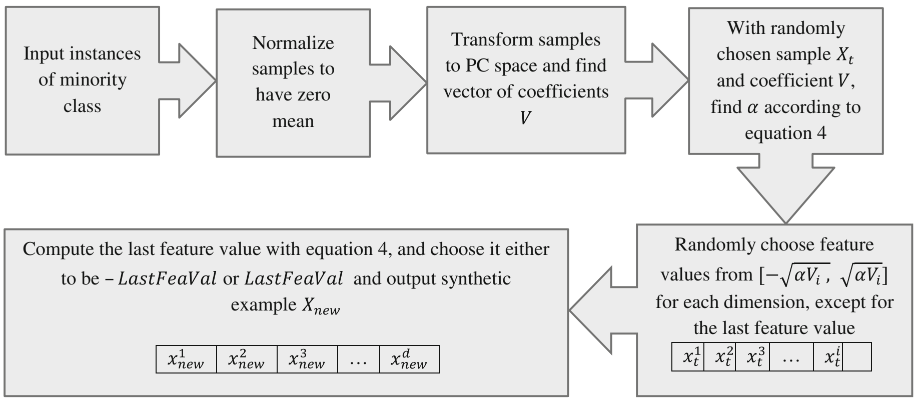
Input instances of minority class
Normalize samples to have zero mean
Transform samples to PC space and find vector of coefficients V
With randomly chosen sample X_t and coefficient V, find α according to equation 4
Randomly choose feature values from $[-\sqrt{\alpha V_i}, \sqrt{\alpha V_i}]$ for each dimension, except for the last feature value
x_t^1 x_t^2 x_t^3 … x_t^i
Compute the last feature value with equation 4, and choose it either to be $-LastFeaVal$ or $LastFeaVal$ and output synthetic example X_{new}
x_{new}^1 x_{new}^2 x_{new}^3 … x_{new}^d

Fig. 2 Overall view of the MDO over-sampling technique. MDO generally is composed of several steps. The process is depicted in the corresponding boxes briefly

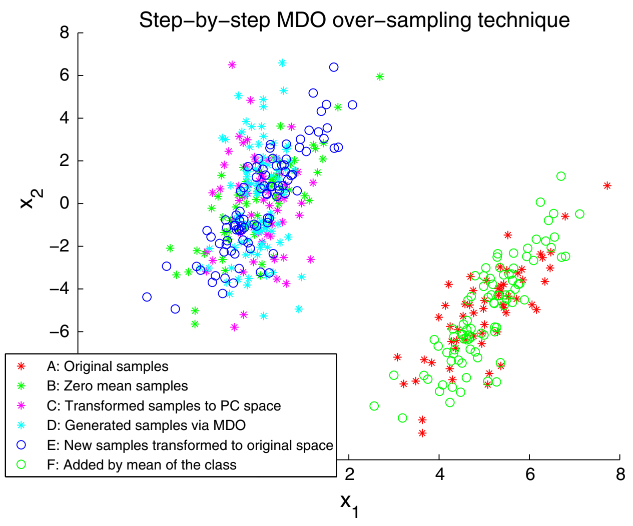

Fig. 3 Step-by-step MDO over-sampling technique just for the considered minority class. *A* Minority class instances in original space. *B* Normalized original samples. *C* Transformed samples to PC space. *D* Generated synthetic samples in PC space. *E* Transformed samples to the original space. *F* Synthetic samples are added by the corresponding class mean

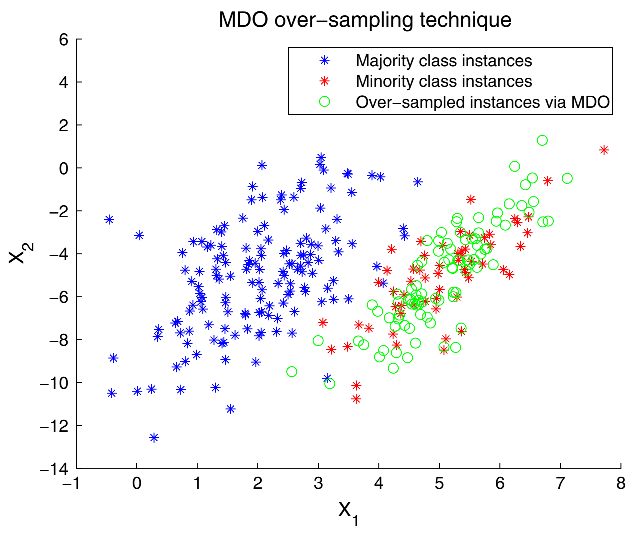

Fig. 4 Two-class artificial data set: minority/majority class samples together with synthetic minority class instances

synthetic samples which are generated in PC space are transformed to the original space and added by the corresponding class mean.

3.2 An illustrative example of the proposed over-sampling technique MDO

Figure 3 illustrates an example of calculating synthetic samples, using MDO over-sampling technique. In this example, a two-dimensional data set with two classes is considered. After normalizing minority class instances and mapping them to PC space, the new samples are generated and transformed to the original space, using Eq. 3. Then, they are added by their class mean. In Fig. 4, the resulting samples together with majority and minority class instances are illustrated.

3.3 Computational complexity of the MDO over-sampling technique

In MDO technique, first the considered instances are normalized to have zero mean. Then, they are mapped to PC space via eigenvalue decomposition; next, synthetic samples are generated in PC space and transformed to the original space. Then, synthetic samples are added by the corresponding class mean. This procedure should be repeated for each considered minority class sample. The most time-consuming part of the algorithm is eigenvalue computation. In general, the complexity of this procedure is $O(n^3)$, but it is just an upper bound. There are many proposed algorithms in the literature which can reduce eigenvalue computation to matrix multiplication or other less expensive methods. For example, Demmel et al. (2007) indicate that all standard linear algebra operations, such as linear equation solving, matrix inversion, (generalized) eigenvalue problems, and the singular value decomposition can be done stably in $O(n^{\omega+\eta})$

operations, for two square $n \times n$ matrices, small real number ω, and any $\eta > 0$. In our case, the dimension of the input data is not very high, so the procedure of eigenvalue computation can be done fast. Creating each synthetic sample needs solving the Eq. 3 two times, which is reduced to solving a linear equation. The first equation, which is the calculation of α, is a linear equation which can be solved in $O(n^{\omega+\eta})$. The second equation, which is the calculation of the last feature value, is just like the previous one except for a square root operation. It can be solved in $O(n^{\omega+\eta})$ as well.

3.4 MDOBoost algorithm

In this section, our MDOBoost approach for multi-class imbalanced problems is described. MDOBoost algorithm applies MDO over-sampling technique in each learning iteration of AdaBoost.M2 algorithm, to enhance the probability of selecting difficult rare examples of the minority classes. AdaBoost.M2 allows the learner to generate more expressive hypotheses whose outputs are vectors in $[0, 1]^c$, rather than a single labels in Y. The yth component of this vector represents a "degree of belief", that the correct label is "y" (Freund and Schapire 1996). The components with values close to 1 or 0 correspond to plausible or implausible labels. This algorithm allows the hypotheses to do well with respect to a more sophisticated error measure which is called pseudo-loss. This pseudo-loss is computed with respect to a distribution over the set of all pairs of examples and incorrect labels. By manipulating this distribution, the boosting algorithm can focus the weak learner not only on hard-to-classify examples but also more specifically on the incorrect labels which are harder to discriminate (Chawla et al. 2003).

In SMOTEBoost and MDOBoost algorithms, through over-sampling techniques SMOTE or MDO, the weight distribution of instances is modified to focus on rare and difficult examples of the minority classes, more than in AdaBoost.M2. Therefore, these combinations of re-sampling and boosting algorithms allow the decision regions of the minority classes to broaden, and the classification performance to improve significantly. MDO over-sampling technique causes a learner to create more accurate and discriminant decision regions over minority classes in data space. Algorithm 2 indicates the pseudo code for MDOBoost algorithm.

Let S be a set of points in the feature space X and y_i be class labels in Y. Let B be a set of mislabels (i, y), $y \neq y_i$ and $|B| = m \times (c - 1)$ in which m is the total number of training examples and c is the number of classes. Let t be an iteration between one and the total number of iterations T, or simply said the number of classifiers in the ensemble. h_t be the base classifier which is trained on iteration t and $h_t(x_i, y)$ be the output of h_t for instance x_i which is a vector of plausibility $\in [0, 1]^c$ for other classes except for x_i's class. $D_t(i)$ be the weight distribution of examples in iteration t.

In step 1 of the algorithm, the weights of the samples are initialized to $1/|B|$. In step 2, the weight of training examples are modified by MDO over-sampling technique to reach a balanced set of training examples. In step 3, a weak learner is trained on these examples using D_t distribution. In step 4, all trained examples are classified with classifier h_t and a vector of plausibility $[0, 1]^c$ for each instance is computed. The pseudo-loss of the classifier h_t is then computed in step 5. Next, the weight of h_t classifier is computed according to $\beta_t = \varepsilon_t/(1-\varepsilon_t)$, so the greater weight is given to hypotheses with lower errors. In step 6, the weight of the training examples are updated and this procedure is repeated for a number of desired iterations. The final output of the ensemble for a test instance is computed using $h_{fin} = \text{argmax}_{y\in Y} \sum_{t=1}^{T} (log \frac{1}{\beta_t}) h_t(x, y)$.

Algorithm 2 The MDOBoost algorithm.

```
Given: Set $S = \{(x_1, y_1), ..., (x_m, y_m)\} x_i \in X$, with labels $y_i \in Y = \{1, ..., c\}$, $|Y| = c \geq 2$, where $c_m, c_m < c$, corresponds to a minority class.
Let $B = \{(i, y) : i = 1, ..., m, y \neq y_i\}$.
Initialize the distribution $D_1$ over the examples, such that
$D_1(i, y) = 1/|B|$ for $(i, y) \in B$.
for $t = 1, 2, 3, ..., T$ do
    Modify distribution $D_t$ by creating $N$ synthetic examples from minority classes $c_m$ using MDO algorithm;
    Train a weak learner using distribution $D_t$;
    Compute weak hypothesis $h_t : X \times Y \to [0, 1]$;
    Compute the pseudo-loss of hypothesis $h_t$ :
    $\varepsilon_t = \sum_{(i,y)\in B} \sum_{y\neq y_i} D_t(i, y)(1 - h_t(x_i, y_i) + h_t(x_i, y));$
    Set $\beta_t = \varepsilon_t/(1 - \varepsilon_t)$ and $w_t = (1/2)(1 - h_t(x_i, y) + h_t(x_i, y_i));$
    Update $D_t : D_{t+1}(i, y) = (D_t(i, y)/Z_t)\beta_t^{w_t}$
    where $Z_t$ is a normalization constant chosen such that $D_{t+1}$ is a distribution;
end for
Output the final hypothesis:
$h_{fin} = \text{argmax}_{y\in Y} \sum_{t=1}^{T} (log \frac{1}{\beta_t}) h_t(x, y)$.
```

4 Experimental settings

The proposed MDOBoost algorithm is compared with other state-of-the-art data-level and model-based techniques. We also used the AdaBoost.M1 algorithm as our ensemble baseline method to compare with other over-sampling and ensemble methods. Moreover, a baseline method, in which no sampling technique is applied, is used as a baseline method for over-sampling techniques.

With respect to the parameter setting, in SMOTE and SMOTEBoost methods, the nearest neighbour parameter K

is set to 5 which is the most accepted value in the literature. The rate of the over-sampling for a class c is the size difference between the largest class and class c. Each minority class is over-sampled as equally as the size of the majority class in all over-sampling and sampling/boosting methods. Therefore, a balanced set of training examples is presented to each learner. Each ensemble consists of 11 base classifiers. Experiments performed using more base learners did not result in significantly different results, so we set the number of classifiers to a lower value. Moreover, to evaluate class decomposition techniques, we compare our results with OAA and OAO, the most frequently used schemes in the multi-class imbalanced literature (Wang and Yao 2012).

4.1 Data sets and base learners

The performance of the methods for multi-class imbalanced problems is evaluated using 14 multi-class benchmark data sets from the UCI (Frank and Asuncion 2010) and KEEL (Alcal et al. 2010) repositories. These data sets have at least one class with significantly smaller number of instances than other classes. Table 1 provides the characteristics of these data sets, including the number of classes, data set sizes, class distributions, number of features, and imbalance ratio (IR). The smallest data set is *Hayes-roth* with 132 instances and the largest one is *Nursery* with 12985 samples. The imbalanced ratio (IR) of these data sets varies between 1.48 and 175.46.

To evaluate the performance of different algorithms, we used several base classifiers for our experiments which are implemented in Weka (Witten and Frank 2005), an open-source data mining tool. These four base classifiers are used since they have different characteristics and produce various degrees of (un)stable models within the ensemble. Base classifiers' characteristics affect the experimental results of the comparative algorithms by generating different degrees of ensemble diversity. The following classifiers were used: (a) C4.5 decision tree (Quinlan 1993) which uses normalized information gain splitting criterion from information theory (Aczl and Darczy 1975). We used the default Weka parameters for this version of the C4.5 decision tree; (b) the second version of C4.5, denoted here as C4.5UP, disables pruning and uses Laplace smoothing (Weiss and Provost 2003); (c) RIPPER (Cohen 1995), Repeated Incremental Pruning to Produce Error Reduction, is a well-known rule-based algorithm. It handles multiple classes by ordering them from the least to the most prevalent and then treating each in order as a distinct two-class problem (Kotsiantis et al. 2007); (d) Naïve Bayes classifier (Mitchell 1997) is a probabilistic learner based on Bayes' theorem with naïve independence assumption. As some data sets have very small classes of data, we perform a fivefold cross-validation to be sure that each fold of test or train contains at least one example of each class.

4.2 Assessment metrics

To evaluate the performance of comparative techniques, four different performance metrics are used. Traditionally, the most widely used metrics of learning algorithms are accuracy and error rate. However, they can be deceiving in certain situations, especially imbalanced learning, and are highly sensitive to changes in data (He and Garcia 2009). Two single-class and two multi-class performance measures are used. Precision and recall (Rijsbergen 1979) are the most prevalently used single-class measures for two-class problems. We computed these metrics for the minority and the majority classes individually. In each case, one of the classes is considered as a positive class and all other classes are considered as negative; the problem is reduced to a two-class task and the single class performance metric can be calculated simply.

Precision is a measure of exactness, i.e. from among the examples labelled as positive, how many are actually labelled correctly (He and Garcia 2009). On the other hand, recall is a measure of completeness, i.e. how many examples of the positive class were labelled correctly. Precision and recall are provided in Eqs. 4 and 5, respectively.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \tag{4}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{5}$$

In which TP is the number of instances of positive class which are classified correctly. FP is the number of instances of negative class which are classified as positive and FN is the number of instances of positive class which are classified as negative.

Extended G-mean (Kubat et al. 1997) for multi-class cases which was adapted by Sun et al. (2006) is provided in Eq. 6. It is defined as the geometric mean of the recall over all classes. Given a *c-class* problem:

$$\text{G-mean} = \left(\prod_{i=1}^{c} R_i\right)^{\frac{1}{c}}. \tag{6}$$

$$R_i = \frac{\text{TP}_i}{\text{TP}_i + \text{FN}_i}, \quad i = 1, \ldots, c.$$

MAUC or M-measure (Hand and Till 2001) is the average of AUC (Bradley 1997) over all pairs of classes. It is defined as follows:

$$\text{MAUC} = \frac{2}{c(c-1)} \sum_{i<j} \frac{[A(i,j) + A(j,i)]}{2}. \tag{7}$$

where $A(i, j)$ is the AUC between class i and j calculated from the ith column of M. M is the $m \times c$ matrix which is provided by the classifier. Each element of M, $t_{p,q}$, indicates the probability of belongingness of instance p to class q. We should be aware that in multi-class cases $A(i, j)$, $A(j, i)$ may

Table 1 Description of 14 UCI and KEEL benchmark data sets with their characteristics; including number of classes, data set size, class distribution, number of features, and Imbalance Ratio (IR)

Data set	Class	Size	Class distribution	No. of features	IR
Hayes-roth	3	132	51/51/30	4	1.7
Lymphography	3	147	81/61/5	18	16.2
Wine	3	178	71/59/48	13	1.48
Glass	6	214	70/76/17/13/9/29	10	8.44
New-thyroid	3	215	150/35/30	5	5
Cleveland	5	303	164/55/36/35/13	13	12.62
Ecoli	5	327	143/77/52/35/20	7	7.15
Dermatology	6	366	112/61/72/49/52/20	34	5.6
Pageblocks	4	545	492/33/8/12	10	61.5
Balance	3	625	49/288/288	4	5.88
Contraceptive	3	1473	629/333/511	9	1.89
Page	5	5473	4913/329/28/88/115	10	175.46
Satimage	6	6435	1533/703/1358/626/707/1508	36	2.45
Nursery	4	12958	4044/4266/4320/328	8	13.17

not be equal, so both of them should be taken into account in calculation of MAUC. As it is obvious from the concept of all these imbalanced assessment metrics, the greater the value of these metrics, the better the classification performance.

4.3 Analyses and observations

To evaluate the statistically significant differences of the results from the comparative methods, we conduct the Friedman test (Friedman 1937) with the corresponding post-hoc test which was recommended by Demsar (2006). The Friedman test is a non-parametric statistical method for ranking all the algorithms over all data sets separately. Ranking the comparative methods over multiple data sets is an appropriate way of evaluation (Demsar 2006). Friedman test compares the mean ranks of the considered techniques. The best performing algorithm gets the rank of 1, the second best the rank of 2, etc. In tie cases, the average ranks are assigned to the considered methods. Therefore, smaller value indicates a higher rank of performance. If the null hypothesis, i.e., all algorithms are equivalent, is rejected then we proceed with the post-hoc test to find out which algorithms actually differ. We use an improved Friedman test which is proposed by Iman and Davenport (1980). The Bonferroni–Dunn test (Dunn 1961) is used as the post-hoc test method. Since the studied data sets are multi-class, we only discuss the recall and precision of the smallest and the largest classes, which are the most typical ones in the data set. Tables 2, 3, 4, 5, 6, 7, 8 and 9 present the results of Friedman and post-hoc tests on MAUC, G-mean, minority class recall (R_{min}), minority class precision (P_{min}), majority class recall (R_{maj}), and majority class precision (P_{maj}) for comparative methods over 14 data sets. In our experiments, the MDOBoost algorithm is selected as the "control" method, where each value in post-hoc tables indicates the difference of the mean ranks between the "control" method and one other technique in corresponding column. The CD value, which stands for critical difference, is determined by the number of competing algorithms and data sets. If their difference of mean ranks is greater than the CD, the performance of these two algorithms is significantly different. We used F_{F} test that is based on Friedman's χ^2_{F} statistic. In most cases there are critical differences, so we reject the null hypothesis and proceed with the post-hoc test with $q_{0.1} = 2.45$ and CD = 2.268.

- The comparative methods with C4.5 classifier: in Tables 2 and 3, the results of Friedman and corresponding post-hoc tests with C4.5 classifier are reported. Table 2 summarizes the average ranks of comparative methods over all data sets. A smaller value shows a higher rank, i.e., better performance. We noticed that the best performing algorithm in terms of MAUC is SMOTEBoost with average rank of 2.2857. The second best performing algorithm is MDOBoost with average rank of 2.3571, which is significantly better than SMOTE, MDO, OAO, and OAA. In terms of G-mean and minority class recall, MDOBoost performs significantly better than class decomposition techniques OAO and OAA with average ranks of 2.8571 and 2.0357, respectively. In minority class precision, there is no significant difference between algorithms. AdaBoost.M1 performs better than other techniques in terms of majority class recall with average rank of 2.7500. In majority class precision, SMOTEBoost performs better than others with average rank of 3.2857. Class decomposition techniques

Table 2 Mean ranks of eight comparative methods over 14 data sets, including the Baseline, SMOTE, MDO, AdaBoost.M1 (Ada.M1), SMOTEBoost (SMB), MDOBoost (MDB), OAO, and OAA with C4.5 as the base classifier

Rank	Baseline	SMOTE	MDO	Ada.M1	SMB	MDB	OAO	OAA
MAUC	4.5000	6.6429	6.6786	2.5357	2.2857	2.3571	5.7143	5.2857
G-mean	4.6429	4.1786	4.6786	4.5357	2.8929	2.8571	5.5714	6.6429
$R_{\min}$	4.7857	4.0714	4.0357	5.0000	3.7857	3.0357	5.8571	5.4286
$P_{\min}$	4.3929	4.6429	5.6429	4.1786	4.6429	3.9286	4.1429	4.4286
R_{maj}	4.7857	5.7143	5.6429	2.7500	4.0714	3.8214	4.1786	5.0357
P_{maj}	5.2500	3.7500	4.0714	4.3214	3.2857	4.6429	5.2143	5.4643

Table 3 Friedman test with the corresponding post-hoc test, Bonferroni–Dunn, with C4.5 as the base classifier

	Friedman	Baseline	SMOTE	MDO	Ada.M1	SMB	OAO	OAA
MAUC	Reject	2.1429	**4.2857**	**4.3214**	0.1786	0.0714	**3.3571**	**2.9286**
G-mean	Reject	1.7857	1.3214	1.8214	1.6786	0.0357	**2.7143**	**3.7857**
$R_{\min}$	Reject	1.7500	1.0357	1.0000	1.9643	0.7500	**2.8214**	**2.3929**
$P_{\min}$	Not reject							
R_{maj}	Reject	0.9643	1.8929	1.8214	1.0714	0.2500	0.3571	1.2143
P_{maj}	Not reject							

The difference of mean ranks between MDOBoost and every other comparative method is computed. CD is critical difference which is set to 2.268. The values greater than CD indicate significant differences between the methods which are highlighted in boldface

Table 4 Mean ranks of eight comparative methods over 14 data sets, including the Baseline, SMOTE, MDO, AdaBoost.M1 (Ada.M1), SMOTEBoost (SMB), MDOBoost (MDB), OAO, and OAA with Naïve Bayes as the base classifier

Rank	Baseline	SMOTE	MDO	Ada.M1	SMB	MDB	OAO	OAA
MAUC	3.9286	3.5357	4.6071	4.5714	3.3571	4.3571	6.7143	4.9286
G-mean	3.6429	4.1071	5.8571	3.6429	3.4286	5.4286	4.0357	5.8571
$R_{\min}$	4.5000	3.5714	4.3571	4.6429	4.2143	5.3571	4.5357	4.8214
$P_{\min}$	4.0000	4.4286	5.3571	3.5357	6.0357	5.4286	3.4643	3.7500
R_{maj}	3.7143	5.8571	5.2500	3.4643	6.1071	4.4643	3.3929	3.7500
P_{maj}	4.0000	4.2500	4.6071	4.4286	3.7500	5.6786	4.0357	5.2500

Table 5 Friedman test with the corresponding post-hoc test, Bonferroni–Dunn, with Naïve Bayes as the base classifier

	Friedman	Baseline	SMOTE	MDO	Ada.M1	SMB	OAO	OAA
MAUC	Reject	0.4286	0.8214	0.2500	0.2143	1.0000	**2.3571**	0.5714
G-mean	Reject	1.7857	1.3214	0.4286	1.7857	2.0000	1.3929	0.4286
$R_{\min}$	Not reject							
$P_{\min}$	Reject	1.4286	1.0000	0.0714	1.8929	0.6071	1.9643	1.6786
R_{maj}	Reject	0.7500	1.3929	0.7857	1.0000	1.6429	1.0714	0.7143
P_{maj}	Not reject							

The difference of mean ranks between MDOBoost and every other comparative method is computed. CD is critical difference which is set to 2.268. The values greater than CD indicate significant differences between the methods which are highlighted in boldface

OAO and OAA have poor performance in almost all cases. AdaBoost.M1 algorithm ranks third in MAUC after SMOTEBoost and MDOBoost. It is known that AdaBoost solely cannot handle class imbalance distributions (Sun et al. 2007; Joshi et al. 2002). As it can be seen from Table 2, low G-mean of AdaBooat.M1 results from its low minority class recall. MDOBoost and SMOTEBoost which use advantages of both data-level and model-based techniques indicate significantly better results than baseline method, over-sampling tech-

Table 6 Mean ranks of eight comparative methods over 14 data sets, including the Baseline, SMOTE, MDO, AdaBoost.M1 (Ada.M1), SMOTEBoost (SMB), MDOBoost (MDB), OAO, and OAA with RIPPER as the base classifier

Rank	Baseline	SMOTE	MDO	Ada.M1	SMB	MDB	OAO	OAA
MAUC	4.5000	6.6429	6.6786	2.5357	2.2857	2.3571	5.7143	5.2857
G-mean	5.0357	4.8929	6.2500	4.1786	2.7500	2.3214	5.0357	5.5357
$R_{\min}$	4.9643	4.4643	4.6071	4.7500	3.0357	2.8571	5.7500	5.5714
$P_{\min}$	5.2143	6.1429	5.7857	3.6071	4.1071	3.5000	3.6071	4.0357
R_{maj}	3.6786	6.6786	6.1786	2.8571	4.3929	3.5357	4.0000	4.6786
P_{maj}	6.1429	3.6786	5.1786	4.9286	3.3929	3.6429	4.8214	4.2143

Table 7 Friedman test with the corresponding post-hoc test, Bonferroni–Dunn, with RIPPER as the base classifier

	Friedman	Baseline	SMOTE	MDO	Ada.M1	SMB	OAO	OAA
MAUC	Reject	2.1429	**4.2857**	**4.3214**	0.1786	0.0714	**3.3571**	**2.9286**
G-mean	Reject	**2.7143**	**2.5714**	**3.9286**	1.8571	0.4286	**2.7143**	**3.2143**
$R_{\min}$	Reject	2.1071	1.6071	1.7500	1.8929	0.1786	**2.8929**	**2.7143**
$P_{\min}$	Reject	1.7143	**2.6429**	**2.2857**	0.1071	0.6071	0.1071	0.5357
R_{maj}	Reject	0.1429	**3.1429**	**2.6429**	0.6786	0.8571	0.4643	1.1429
P_{maj}	Reject	**2.5000**	0.0357	1.5357	1.2857	0.2500	1.1786	0.5714

The difference of mean ranks between MDOBoost and every other comparative method is computed. CD is critical difference which is set to 2.268. The values greater than CD indicate significant differences between the methods which are highlighted in boldface

Table 8 Mean ranks of eight comparative methods over 14 data sets, including the Baseline, SMOTE, MDO, AdaBoost.M1 (Ada.M1), SMOTEBoost (SMB), MDOBoost (MDB), OAO, and OAA with C4.5UP as the base classifier

Rank	Baseline	SMOTE	MDO	Ada.M1	SMB	MDB	OAO	OAA
MAUC	5.2143	5.6071	6.1786	3.5000	2.3571	1.9286	6.5000	4.7143
G-mean	5.7143	4.3571	5.4643	3.8214	2.5000	3.1786	6.0357	4.9286
$R_{\min}$	5.5357	4.0000	4.6786	4.0000	3.1429	3.0714	6.0000	5.5714
$P_{\min}$	5.2857	5.2857	5.7143	3.1429	3.8929	4.4286	4.9643	3.2857
R_{maj}	5.1429	6.4643	5.1786	3.1786	4.6071	4.0714	3.6071	3.7500
P_{maj}	5.6786	4.0357	4.8929	4.0000	2.6429	4.5000	5.6786	4.5714

Table 9 Friedman test with the corresponding post-hoc test, Bonferroni–Dunn, with C4.5UP as the base classifier

	Friedman	Baseline	SMOTE	MDO	Ada.M1	SMB	OAO	OAA
MAUC	Reject	**3.2857**	**3.6786**	**4.2500**	1.5714	0.4286	**4.5714**	**2.7857**
G-mean	Reject	**2.5357**	1.1786	**2.2857**	0.6429	0.6786	**2.8571**	1.7500
$R_{\min}$	Reject	**2.4643**	0.9286	1.6071	0.9286	0.0714	**2.9286**	**2.5000**
$P_{\min}$	Reject	0.8571	0.8571	1.2857	1.2857	0.5357	0.5357	1.1429
R_{maj}	Reject	1.0714	**2.3929**	1.1071	0.8929	0.5357	0.4643	0.3214
P_{maj}	Reject	1.1786	0.4643	0.3929	0.5000	1.8571	1.1786	0.0714

The difference of mean ranks between MDOBoost and every other comparative method is computed. CD is critical difference which is set to 2.268. The values greater than CD indicate significant differences between the methods which are highlighted in boldface

niques, and class decomposition schemes. MDOBoost is slightly better than SMOTEBoost in terms of G-mean, minority class recall, and minority class precision.

- The comparative methods with Naïve Bayes classifier: Tables 4 and 5 summarize the results of comparative algorithms with Naïve Bayes as the base classifier. MDOBoost performs worse than SMOTEBoost in MAUC, G-mean, minority class recall, and majority class precision, but performs better in minority class precision and majority class recall. The best performing algorithm in

MAUC is SMOTEBoost method. Also in G-mean, the best performing algorithm is SMOTEBoost. Meanwhile, it is encouraging to see that MDOBoost does not lose too much performance on neither of the evaluation metrics, where no significant differences are observed. OAO and OAA perform better than SMOTEBoost and MDOBoost in majority class recall and minority class precision; however, they showed performance degradation in G-mean and MAUC. The performance degradation of OAA in G-mean is probably related to recall reduction of the minority class.

- The comparative methods with RIPPER classifier: Tables 6 and 7 summarize the mean ranks of our comparative methods together with the Friedman and post-hoc tests with RIPPER classifier. According to the ranks, the best performing algorithms are SMOTEBoost and MDOBoost in MAUC. Class decomposition techniques and over-sampling methods perform poorly in MAUC. MDOBoost performs significantly better than OAO, OAA, MDO, and SMOTE. In G-mean, minority class recall, and minority class precision, the lower rank is related to MDOBoost with mean ranks of 2.3214, 2.8571, and 3.500, respectively. In majority class recall, AdaBoost.M1 ranked first among other methods with average rank of 2.8571. The second and third best algorithms are MDOBoost and baseline method, respectively. In majority class precision, SMOTEBoost and MDOBoost outperform other algorithms. MDOBoost performs significantly better than over-sampling and class decomposition schemes in MAUC, G-mean, and minority class recall. It seems that, taking advantages of both over-sampling technique MDO and AdaBoost.M2, the MDOBoost algorithm improves the learning ability of the RIPPER classifier significantly, even better than SMOTEBoost algorithm.
- The comparative methods with C4.5UP classifier: Tables 8 and 9 summarize the results of the comparative methods together with the Friedman and post-hoc tests with C4.5UP classifier. According to Table 8, MDOBoost ranked first in MAUC and minority class recall. In terms of G-mean and majority class precision, SMOTEBoost is better than other techniques with mean ranks of 2.500 and 2.6429, respectively. In majority class recall and minority class precision, AdaBoost.M1 outperforms other techniques. MDOBoost performs significantly better than class decomposition schemes in MAUC and minority class recall. The overall performance of our MDOBoost algorithm with C4.5 is better than with C4.5UP.

G-mean is geometric mean of recall over all classes of data. If any of the classes has very low recall, the value of G-mean will decrease. This metric indicates that how well a learner can recognize different class instances correctly. A high G-mean guarantees that all classes are considered. On the other hand, MAUC evaluates the average ability of a classifier in separating class pairs. A high MAUC indicates that a learner is good at separating most class pairs; however, it is still possible that some classes are hard to be distinguished from the other classes. G-mean is more sensitive to single class performance than MAUC.

According to ranks of algorithms over all data sets, the best two performing algorithms in MAUC are MDOBoost and SMOTEBoost. It seems that combining over-sampling techniques with boosting procedure results in better discrimination of class pairs. Since in these two algorithms the minority class instances have modelled better, the broader representation of these examples is provided. These two over-sampling techniques cause the classifier to build larger decision regions which contain nearby minority class points. Therefore, the combination of these methods with boosting makes the learner to recognize class pairs better and results in greater MAUC. In G-mean, the two best performing methods are MDOBoost and SMOTEBoost. Probably, these two algorithms which are tuned for learning from imbalanced data sets can recognize minority class instances better; this, in turn, increases the G-mean metric. In minority class recall, the best performing method is MDOBoost with C4.5, RIPPER, and C4.5UP classifiers and SMOTE over-sampling with Naïve Bayes classifier, respectively.

Over-sampling methods MDO and SMOTE, which improve coverage of the minority class examples (Chawla et al. 2003), have the effect of improving the recall of minority class. Increased generalization ability of a classifier by over-sampling techniques makes the classifier to perform significantly better in recalling minority class instances. In minority class precision, MDOBoost, AdaBoost.M1, and OAA perform better than other techniques. It is probably related to the algorithm, because the training samples in OAA and AdaBoost.M1 cases did not over-sample and the rate of false positive did not increase. Likewise SMOTEBoost, MDOBoost indicates better overall precision in minority class; this, in turn, indicates that the probability of over-fitting or over-generalization of the classifier with over-sampled instances via MDO decreases considerably.

It is reported that over-sampling methods like SMOTE cause degradation in precision due to increased number of false-positive samples (Chawla et al. 2003). These over-samplings cause the algorithm to over-generalize and this, in turn, causes the degradation of precision. In recalling majority class instances, AdaBoost.M1 and OAO algorithms perform better than others. In these methods in which we did not use over-sampling pre-processes, the rate of false negative in majority class did not increase to cause the degradation of this metric. In majority class precision, SMOTEBoost algorithm performs better in all cases.

Pruned and unpruned trees can have varied effects on learning from imbalanced data sets (Chawla 2003). Pruning can be detrimental to learning from imbalanced data sets as it can potentially collapse (small) leaves belonging to the minority class, thus reducing the coverage (Chawla 2003). Using MDO and MDOBoost with pruned version of C4.5 indicated better overall performance than unpruned version. According to the Tables 3 and 9, with the pruned version, MDOBoost significantly outperforms MDO, SMOTE, OAO, and OAA in MAUC, baseline and class decomposition methods in G-mean and minority class precision.

While SMOTEBoost performs better than MDOBoost with C4.5UP, it seems that the learning ability with MDO improves significantly with pruned C4.5. This is the opposite in SMOTE over-sampling. SMOTE indicates better results with unpruned version (Chawla et al. 2003). Due to the way MDO over-samples, new synthetic minority class samples almost preserve the same characteristics as the original class instances. It seems that SMOTE over-samples the examples in such a way that the underlying distribution of the new instances differs considerably than MDO. It is probable that, in most of the considered data sets in this research, the mean of each candidate class can be a good representative for the corresponding class instances. MDO over-samples new examples along the probability contours which have the same Mahalanobis distance from the considered class mean. To investigate the performances of these two over-sampling techniques, we compare these two techniques in next section.

In general, over-sampling methods in the presence of multi-class imbalanced data sets cause the classifier to overfit or over-generalize. Combining boosting procedure with over-sampling techniques have significantly better overall results than class decomposition schemes or over-sampling methods. These methods can improve the overall performance of learning algorithms. Greater values in all metrics over all data sets indicate the good effect of these kinds of algorithms. In other words, the application of sampling/boosting is better than applying boosting or sampling methods alone.

4.4 Investigating the effects of SMOTE and MDO techniques on class distribution

In this section we want to scrutinize the over-sampling techniques SMOTE and MDO, to find out the effects of these techniques on minority class distributions. To investigate these techniques, first we consider one of the minority classes of each data sets; six data sets are used here. The considered class samples are over-sampled as the same size as the considered class size, via SMOTE and MDO. To examine the distribution of samples, we estimate the distribution of the considered class instances before and after over-sampling. One of the minority classes is considered with at least 30 examples, which is a widely utilized minimum on sample size for statistical inference purposes. Since the observations are being drawn from some unknown probability density $P(x)$ in some d-dimensional space, we used Kernel Density Estimation (KDE) (Bishop 2007) with Gaussian kernel function to estimate the underlying distribution. The Gaussian kernel function is given in Eq. 8.

$$K(x) = \frac{1}{(2\pi)^{\frac{d}{2}}} \exp\left(-\frac{1}{2}x^T x\right). \tag{8}$$

The expression of the density estimate KDE is as follows:

$$P_{\mathrm{KDE}}(x) = \frac{1}{N}\sum_{n=1}^{N}\frac{1}{h^d}K\left(\frac{x-x_n}{h}\right). \tag{9}$$

In which h represents the standard deviation of the Gaussian components, N is the number of considered samples, and d is the dimension of the input data. For each class samples and each technique, we estimate the probability density function before and after over-sampling. Then, these two probability densities should be compared. In probability and information theory, the Kullback–Leibler divergence (Kullback and Leibler 1951) is a non-symmetric measure of the difference between two probability distributions P and Q. Typically, P represents the "true" distribution of observations and Q represents the "approximation" of P. In our case, P refers to the distribution of instances before over-sampling and Q refers to the distribution of synthetic examples. For discrete probability distributions P and Q, the KGL divergence of Q from P is defined as follows:

$$D_{\mathrm{KL}}(P||Q) = \sum_i \ln\left(\frac{P(i)}{Q(i)}\right)P(i). \tag{10}$$

Table 10 summarizes our results over six different data sets of our studies. The average values of 100 independent runs are averaged and reported. The K–L divergence of two considered distributions are computed. The lower values indicate the better technique which results in lower difference between true distributions and synthetic ones. The bandwidth parameter h is the same for all the axes and can be calculated with $1.06\sigma N^{\frac{-1}{5}}$, where σ indicates the variance which is set to four different values 0.25, 0.5, 0.75, and 1.

Here, we want to evaluate the distribution of synthetic samples of MDO and SMOTE over-sampling techniques from the view of Normal distribution. As it is obvious, in SMOTE and MDO techniques, different dimensions of the input data are not considered independent. Consequently, multivariate Normality tests should be considered. Henze and Zirkle (1990) introduce a multivariate version of the univariate Normality test. There are many tests for assessing the multivariate Normality in the statistical literature (Lee and Cho 2006). It has been found that the Henze and

Table 10 The results of comparing distributions of original and over-sampled examples over one minority class sample of each data set

Data/sigma	$\sigma = 0.25$	$\sigma = 0.5$	$\sigma = 0.75$	$\sigma = 1.0$
Ecoli	SMOTE: **0.047593**	SMOTE: **0.023913**	SMOTE: **0.016846**	SMOTE: **0.011843**
	MDO: 0.054994	MDO: 0.027335	MDO: 0.019590	MDO: 0.013025
Dermatology	SMOTE: 0.002922	SMOTE: 0.006038	SMOTE: 0.010753	SMOTE: 0.016475
	MDO: **0.000000**	MDO: **0.000000**	MDO: **0.000026**	MDO: **0.000265**
Nursery	SMOTE: 0.004463	SMOTE: 0.009541	SMOTE: 0.017638	SMOTE: 0.024533
	MDO: **0.000000**	MDO: **0.000000**	MDO: **0.000000**	MDO: **0.000000**
New-thyroid	SMOTE: 0.001351	SMOTE: 0.001980	SMOTE: 0.003430	SMOTE: 0.004141
	MDO: **0.000000**	MDO: **0.000000**	MDO: **0.000000**	MDO: **0.000000**
Balance	SMOTE: 0.006850	SMOTE: 0.013336	SMOTE: 0.023850	SMOTE: **0.031928**
	MDO: **0.000816**	MDO: **0.006179**	MDO: **0.019384**	MDO: 0.038976
Page	SMOTE: **0.062518**	SMOTE: **0.057989**	SMOTE: **0.055556**	SMOTE: **0.053595**
	MDO: 0.352834	MDO: 0.352834	MDO: 0.352834	MDO: 0.352834

Lower values which indicate small difference between two distributions are highlighted in boldface

Table 11 The results of Henze–Zirkler's multivariate normality test; comparing distributions of original and over-sampled examples via SMOTE and MDO techniques over all minority class samples of considered data sets

Data	Class distribution	Considered class	SMOTE	MDO	Original distribution
Hayeth-roth	51/51/30	2nd class	**Not Normal**	**Normal**	**Normal**
Wine	71/59/48	2nd class	**Not Normal**	**Normal**	**Normal**
		3rd class	**Not Normal**	**Normal**	**Normal**
New-thyroid	150/35/30	2nd class	**Not Normal**	**Normal**	**Not Normal**
		3rd class	**Not Normal**	**Normal**	**Not Normal**
Ecoli	143/77/52/35/20	2nd class	Not Normal	Not Normal	Not Normal
		3rd class	Not Normal	Not Normal	Not Normal
		4th class	Not Normal	Not Normal	Not Normal
		5th class	Not Normal	Not Normal	Not Normal
Dermatology	112/72/61/52/49/20	2nd class	Not Normal	Not Normal	Not Normal
		3rd class	Not Normal	Not Normal	Not Normal
		4th class	Not Normal	Not Normal	Not Normal
		5th class	Not Normal	Not Normal	Not Normal
		6th class	Not Normal	Not Normal	Not Normal
Balance	288/288/49	2nd class	Not Normal	Not Normal	Not Normal
Page	4913/329/115/88/28	2nd class	Not Normal	Not Normal	Not Normal
		3rd class	Not Normal	Not Normal	Not Normal
		4th class	Not Normal	Not Normal	Not Normal
		5th class	**Not Normal**	**Normal**	**Not Normal**
Nursery	4320/4266/4044/328	2nd class	Not Normal	Not Normal	Not Normal
		3rd class	Not Normal	Not Normal	Not Normal
		4th class	Not Normal	Not Normal	Not Normal

Differences between comparing the distributions of techniques are highlighted in boldface

Zirkler's (Henze and Zirkle 1990) test has a good overall power against alternatives to normality. We used Henze and Zirkler's multivariate normality test which is implemented in Matlab (Trujillo-Ortiz et al. 2007).

We conduct the multivariate Normality test on six benchmark data sets. The considered data sets are over-sampled via SMOTE and MDO as equal as the size of each class. Then, the Henze and Zirkler's multivariate Normality test is conducted over the original and synthetic data of each class.

Table 11 summarizes the results of the performed tests. Over all benchmarks, the distribution of original and synthetic samples do not have Normal distribution, except for Hayes-roth, Wine, New-thyroid, and Page data sets. Over Hayes-roth and Wine data sets, the original samples of the considered classes and synthetic samples of MDO algorithm are Gaussian. Over these two data sets, the distribution of synthetic samples of the MDO algorithm resembles the original minority class distribution more than the synthetic examples of SMOTE algorithm. For New-thyroid and Page data sets, synthetic samples of MDO technique have a Normal distribution. In these two cases, generated samples via SMOTE method resemble the original samples more than MDO samples. The overall results with different classifiers also indicate that SMOTE technique have better overall performance on these two data sets than MDO technique.

Tables 10 and 11, in most cases, MDO in comparison with SMOTE, generates better examples which are more similar to the true class distribution. In these cases, the underlying distribution of minority class instances before and after over-sampling differs less in MDO than in SMOTE. Since the test examples are drawn from almost the same distribution as trained samples, the learned classifier has better performance on test data with MDO algorithm. Using MDO synthetic samples with AdaBoost.M2 procedure in MDOBoost algorithm indicates better overall results. This was, also, revealed by our empirical observations over studied benchmarks.

5 Conclusion and future work

In this paper, a new sampling/boosting algorithm for multi-class imbalanced data sets is proposed. This techniques is called MDOBoost, since in each learning iteration of AdaBoost.M2 algorithm, all minority class instances are over-sampled via our proposed MDO over-sampling technique. MDO, which stands for Mahalanobis distance-based over-sampling technique, generates new synthetic examples which have the same Mahalanobis distance from the considered class mean. In fact, these newly generated examples almost preserve the same characteristics as the original minority class samples.

To examine the performance of the proposed technique, we compared our results with over-sampling methods, class decomposition schemes, and SMOTEBoost algorithm. Fourteen UCI and KEEL multi-class imbalanced data sets and four different classifiers were used. MDOBoost algorithm significantly outperforms over-sampling and class decomposition techniques. MDOBoost showed better results with pruned version of C4.5 classifier, unlike SMOTEBoost which performs better with unpruned version. In other words, MDO over-samples the examples in such a way that they have almost the same characteristics as original minority class samples. All in all, combining sampling and boosting procedures, which benefit from both techniques sampling and boosting, significantly outperform over-sampling and class decomposition schemes and can recognize minority class examples in multi-class cases better.

Future work of this study includes the following: (1) in almost all data sets that we used, we have low dimensionality. If the dimension of the input data is very high, MDO may take a long running time to select good random numbers for different feature values. In these cases, using a preprocess task and decreasing the data dimensionality or finding another solution to use MDO in higher dimensions seem useful. (2) MDO performs well in most of the studied data sets in which the mean of each class is a good representative for the data characteristics. Investigating the ability of MDO over-sampling technique in multi-modal data sets and adapting it to work well with such data distributions, also, remain as our future studies.

References

Alcal J, Fernndez A, Luengo J, Derrac J, Garca S, Snchez L, Herrera F (2010) KEEL data-mining software tool: data set repository, integration of algorithms and experimental analysis framework. J Mult Valued Log Soft Comput

Alibeigi M, Hashemi S, Hamzeh A (2012) DBFS: an effective density based feature selection scheme for small sample size and high dimensional imbalanced data sets. Data Knowl Eng

Aczl J, Darczy Z (1975) On measures of information and their characterizations. New York

Bishop CM (2007) Pattern recognition and machine learning (information science and statistics)

Bradley AP (1997) The use of the area under the roc curve in the evaluation of machine learning algorithms. Pattern Recognit 30(7):1145–1159

Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP (2002) Smote: synthetic minority over-sampling technique. J Artif Intell Res 16:341–378

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, Berlin, pp 107–119

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, vol 3

Chawla NV, Japkowicz N, Kotcz A (2004) Editorial: special issue on learning from imbalanced data sets. SIGKDD Explor Newsl 6(1):1–6

Chen K, Lu BL, Kwok JT (2006) Efficient classification of multi-label and imbalanced data using min-max modular classifiers. In: International joint conference on neural networks, IJCNN'06, IEEE, pp 1770–1775

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 discovery and data mining. ACM, New York, pp 124–132

Cohen WW (1995) Fast effective rule induction. In: ICML, vol 95, pp 115–123

Demmel J, Dumitriu I, Holtz O (2007) Fast linear algebra is stable. Numer Math 108(1):59–91

Demar J (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7:1–30

Dietterich TG (2000) Ensemble methods in machine learning. In: Multiple classifier systems. Springer, Berlin, pp 1–15

Dunn OJ (1961) Multiple comparisons among means. J Am Stat Assoc 56(293):52–64

Elkan C, Noto K (2008) Learning classifiers from only positive and unlabeled data. In: Proceedings of the 14th ACM SIGKDD international conference on knowledge discovery and data mining, pp 213–220

Fernndez A, Del Jesus MJ, Herrera F (2010) Multi-class imbalanced data-sets with linguistic fuzzy rule based classification systems based on pairwise learning. In: Computational intelligence for knowledge-based systems design. Springer, Berlin, pp 89–98

Frank A, Asuncion A (2010) UCI machine learning repository: http://archive.ics.uci.edu/ml

Freund Y, Schapire RE (1996) Experiments with a new boosting algorithm. In: ICML, vol 96, pp 148–156

Friedman M (1937) The use of ranks to avoid the assumption of normality implicit in the analysis of variance. J Am Stat Assoc 32(200):675–701

Golub TR, Slonim DK, Tamayo P, Huard C, Gaasenbeek M, Mesirov JP, Coller H, Loh ML, Downing JR, Caligiuri MA, Bloomfield CD, Lander ES (1999) Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. Science 286(5439):531–537

Henze N, Zirkler B (1990) A class of invariant consistent tests for multivariate normality. Commun Statist Theor Meth 19(10):3595–3618

Han H, Wang WY, Mao BH (2005) Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning. In: Advances in intelligent computing. Springer, Berlin, pp 878–887

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

Hart PE (1968) The condensed nearest neighbour rule. IEEE Trans Inform Theory 14(3):515–516

Hastie T, Tibshirani R (1998) Classification by pairwise coupling. Ann Stat 26(2):451–471

He H, Bai Y, Garcia EA, Li S (2008) ADASYN: adaptive synthetic sampling approach for imbalanced learning. In: IEEE international joint conference on neural networks. IJCNN (IEEE world congress on computational intelligence), IEEE, pp 1322–1328)

He H, Garcia EA (2009) Learning from imbalanced data. IEEE Trans Knowl Data Eng 21(9):1263–1284

Iman RL, Davenport JM (1980) Approximations of the critical region of the fbietkan statistic. Commun Stat Theory Methods 9(6):571–595

Joshi MV, Agarwal RC, Kumar V (2002) Predicting rare classes: can boosting make any weak learner strong? In: Proceedings of the eighth ACM SIGKDD international conference on knowledge discovery and data mining. ACM, New York, pp 297–306

Kotsiantis S, Kanellopoulos D, Pintelas P (2006) Handling imbalanced datasets: a review. GESTS Int Trans Comput Sci Eng 30(1):25–36

Kotsiantis SB, Zaharakis ID, Pintelas PE (2007) Supervised machine learning: a review of classification techniques

Kubat M, Matwin S (1997) Addressing the curse of imbalanced training sets: one-sided selection. In: Proceeding 14th international conference on machine learning, p 179–186

Kubat M, Holte R, Matwin S (1997) Learning when negative examples abound. In: Machine learning: ECML-97. Springer, Berlin, pp 146–153

Kullback S, Leibler RA (1951) On information and sufficiency. Ann Math Stat 22(1):79–86

Lee HJ, Cho S (2006) The novelty detection approach for different degrees of class imbalance. In: Lecture notes in computer science series as the proceedings of international conference on neural information processing, vol 4233, pp 21–30

Liao TW (2008) Classification of weld flaws with imbalanced class data. Expert Syst Appl 35(3):1041–1052

Mahalanobis PC (1936) On the generalized distance in statistics. Proc Natl Inst Sci (Calcutta) 2:49–55

Mitchell TM (1997) Machine learning. McGraw-Hill, New York

Polikar R (2006) Ensemble based systems in decision making. Circuits Syst Mag IEEE 6(3):21–45

Perrone MP, Cooper LN (1992) When networks disagree: ensemble methods for hybrid neural networks (No. TR-61). Brown University Providence RI Institute for Brain and Neural Systems

Prati RC, Batista GE, Monard MC (2004) Class imbalances versus class overlapping: an analysis of a learning system behavior. In MICAI 2004: advances in artificial intelligence. Springer, Berlin, pp 312–321

Quinlan JR (1993) C4. 5: programs for machine learning, vol 1. Morgan kaufmann

Raskutti B, Kowalczyk A (2004) Extreme re-balancing for svms: a case study. SIGKDD Explor 6(1):60–69

Rifkin R, Klautau A (2004) In defense of one-vs-all classification. J Mach Learn Res 5:101–141

Rijsbergen CV (1979) Information retrieval. Butterworths, London

Seiffert C, Khoshgoftaar TM, Van Hulse J, Napolitano A (2008) Improving learner performance with data sampling and boosting. In: Proceeding of the 20th IEEE international conference on tools with artificial intelligence. ICTAI'08, IEEE, vol 1, pp 452–459

Seiffert C, Khoshgoftaar TM, Van Hulse J, Napolitano A (2010) Rusboost: a hybrid approach to alleviating class imbalance. IEEE Trans Syst Man Cybern Part B Cybern 40(1):185–197

Sun Y, Wong AK, Wang Y (2005) Parameter inference of cost-sensitive boosting algorithms. In: Machine learning and data mining in pattern recognition. Springer, Berlin, pp 21–30

Sun Y, Kamel MS, Wang Y (2006) Boosting for learning multiple classes with imbalanced class distribution. In: Proceeding of the sixth international conference on data mining. ICDM'06, IEEE, pp 592–602

Sun Y, Kamel MS, Wong AK, Wang Y (2007) Cost-sensitive boosting for classification of imbalanced data. Pattern Recognit 40(12):3358–3378

Tan A, Gilbert D, Deville Y (2003) Multi-class protein fold classification using a new ensemble machine learning approach

Tomek I (1976) Two modifications of cnn. IEEE Trans Syst Man Cybern 11:769–772

Trujillo-Ortiz A, Hernandez-Walls R, Barba-Rojo K, Cupul-Magana L (2007) HZmvntest: Henze–Zirkler's multivariate normality test. A MATLAB file [WWW document]. http://www.mathworks.com/matlabcentral/fileexchange/loadFile.do?objectId=17931

Wang BX, Japkowicz N (2004) Imbalanced data set learning with synthetic samples. In: Proceeding of the IRIS machine learning workshop

Wang S, Yao X (2012) Multiclass imbalance problems: analysis and potential solutions. IEEE Trans Syst Man Cybern Part B Cybern 42(4):1119–1130

Weiss GM, Provost F (2001) The effect of class distribution on classifier learning: an empirical study. Rutgers University, USA

Weiss GM, Provost FJ (2003) Learning when training data are costly: the effect of class distribution on tree induction. J Artif Intell Res (JAIR) 19:315–354

Weiss GM (2004) Mining with rarity: a unifying framework. ACM SIGKDD Explor Newsl 6(1):7–19

Wilson DL (1972) Asymptotic properties of nearest neighbour rules using edited data. IEEE Trans Syst Man Cybern 2(3):408–421

Witten IH (2005)and Frank, E. Data mining, practical machine learning tools and techniques. Morgan Kaufmann

Zhao XM, Li X, Chen L, Aihara K (2008) Protein classification with imbalanced data. Proteins Struct Funct Bioinform 70(4):1125–1132

Zhou ZH, Liu XY (2006) Training cost-sensitive neural networks with methods addressing the class imbalance problem. IEEE Trans Knowl Data Eng 18(1):63–77

Zhuang L, Dai H (2006) Parameter optimization of kernel-based one-class classifier on imbalance learning. J Comput 1(7):32–40