To Combat Multi-Class Imbalanced Problems by Means of Over-Sampling Techniques

Lida Abdi and Sattar Hashemi

Abstract—Class imbalance problem is quite pervasive in our nowadays human practice. This problem basically refers to the skewness in the data underlying distribution which, in turn, imposes many difficulties on typical machine learning algorithms. To deal with the emerging issues arising from multi-class skewed distributions, existing efforts are mainly divided into two categories: model-oriented solutions and data-oriented techniques. Focusing on the latter, this paper presents a new over-sampling technique which is inspired by Mahalanobis distance. The presented over-sampling technique, called MDO (Mahalanobis Distance-based Over-sampling technique), generates synthetic samples which have the same Mahalanobis distance from the considered class mean as other minority class examples. By preserving the covariance structure of the minority class instances and intelligently generating synthetic samples along the probability contours, new minority class instances are modelled better for learning algorithms. Moreover, MDO can reduce the risk of overlapping between different class regions which are considered as a serious challenge in multi-class problems. Our theoretical analyses and empirical observations across wide spectrum multi-class imbalanced benchmarks indicate that MDO is the method of choice by offering statistical superior MAUC and precision compared to the popular over-sampling techniques.

Index Terms—Multi-class imbalance problems, over-sampling techniques, Mahalanobis distance

1 Introduction

In recent years, with the accelerated developments in science and technology and availability of data, there is a need for more robust and accurate learning algorithms. Existence of imbalanced distributions among these data is very prevalent. In fact, a data set with unequal number of instances for different classes is called imbalanced data set. This skewness in the data underlying distribution causes many problems for typical machine learning algorithms. In particular, correctly classifying the minority class instances is a main issue in processing these data sets. Simply said, the key point of learning is to obtain a classifier which will provide high accuracy for the minority class without severely jeopardizing the accuracy of the majority class [31].

In many real world applications such as weld flaw [37] and protein fold [56] classifications, there is a presence of multi-class imbalanced data sets. These problems impose many new issues and challenges which have not been seen in two-class ones. Zhou and Liu [57] showed that cost sensitive learning with multi-class tasks is more difficult than two-class ones and a higher degree of class imbalance may increase the difficulty. They also revealed that almost all techniques are effective on two-class problems, while most are ineffective and even may cause negative effects on multi-class tasks.

Existing solutions in dealing with class imbalanced problems are at the data level and algorithmic level. Data level solutions are pre-process tasks which are 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. 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 such as cost sensitive methods, ensemble learning algorithms, and one-class learning (also known as novelty detection and recognition-based methodology) [8] are among the proposed ways of dealing with these problems. Cost sensitive methods consider higher misclassification costs for rare examples; however, in many cases these costs are not available. Ensemble learning algorithms showed great success in various learning problems. Bagging [5], Boosting [22], SMOTEBoost [9], and RareBoost [33] are popular in literature. 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 by using mainly or only a single class of examples.

Many typical machine learning algorithms pose many difficulties dealing with uneven data distributions. Although over-sampling techniques are simple to be used as pre-process tasks, they indicated very great success in many applications. By generating artificial data for the minority classes and training a classifier on a balanced data distribution, the learning process improves significantly; consequently, over-sampling techniques have become an

• *The authors are with the Department of Computer Science and Engineering, School of Electrical and Computer Engineering, Shiraz University, Shiraz, Iran. E-mail: l-abdi@cse.shirazu.ac.ir, s-hashemi@shirazu.ac.ir.*

Manuscript received 18 Oct. 2014; revised 1 July 2015; accepted 7 July 2015. Date of publication 21 July 2015; date of current version 3 Dec. 2015.
Recommended for acceptance by X. He.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TKDE.2015.2458858

effective tool to solve the imbalanced problems. García et al. [24] indicates that "over-sampling the minority class consistently outperforms under-sampling the majority class when data sets are strongly imbalanced, whereas there are not significant differences for databases with a low imbalance".

Previous multi-class imbalanced over-sampling methods, which are proposed by Lin et al. [38] and Fernández-Navarro et al. [20], integrate over-sampling and training phase together. In other words, over-sampling is applied during the training phase of the algorithms. In these algorithms, heuristics are applied to determine whether a training example should be learned. Due to the used heuristics and integration of over-sampling and training phase, the training time and the final model complexity would be very high. Also, these heuristics are used to well tune neural network classifiers. Consequently, just neural networks should be employed in these algorithms.

In this paper, a new simple and yet effective over-sampling technique, called MDO, is proposed to deal with the multi-class imbalanced problems. This approach, which is a kind of data level solution, is inspired by Mahalanobis distance [39]. MDO over-samples the minority classes by considering each candidate minority class sample and generating a new synthetic instance which has equal Mahalanobis distance from the considered class mean with the candidate sample. It generates synthetic samples toward the variation of the corresponding class and helps in reducing the overlapping between different class regions. In multi-class problems over-lapping between different class regions is a prevalent issue. By preserving the covariance structure of the data in minority classes, MDO can make new suitable examples which are very useful in learning algorithms. MDO selects those minority class candidates which are settled in the dense areas of the corresponding class. In other words, new synthetic samples are generated toward the probability contours of the samples which have more neighbours in the same class.

"Class imbalance does not seem to be problematic, but when allied to highly over-lapped classes it can significantly decrease the number of minority class examples correctly classified [3]". In most of the proposed over-sampling techniques, the synthetic samples are generated according to the similarities between minority class samples in feature space. Some of these over-sampling techniques should be used with a specific classifier to reach better results. Also, many existing over-sampling techniques are only used or tuned for two-class tasks and they have poor performance in multi-class imbalanced problems. In multi-class cases, decision boundaries of different classes are harder to be learnt. Only considering similarities between samples in feature space or over-sampling the minority classes via borderline examples near boundaries increases the overlapping probability between different class regions. By considering suitable samples in each minority class and generating synthetic samples in dense areas of the feature space, MDO can reduce the risk of overlapping between different class regions. MDO not only improves the generalization ability of a classifier remarkably, but it also does not cause the classifier to over-fit or over-generalize as much as the other over-sampling techniques. We tested our technique over 20 multi-class UCI [21] and KEEL [2] benchmark data sets with three different classifiers. Also, our technique was compared with four popular and benchmark over-sampling methods in literature. Our empirical analyses and conducted statistical tests reveal that MDO is superior to other comparative data-level algorithms in terms of MAUC and minority class precision.

The rest of the paper is organized as follows: Section 2 introduces the popular research progresses in class imbalanced learning problems in the literature. Section 3 explains the proposed method in detail. Our experimental settings and analyses of the results are provided in Section 4. Finally, Section 5 concludes the paper and presents our future work.

2 Related Work

In this section, several approaches in the area of class imbalance problems are presented. The advantages of sampling techniques are that they are more versatile and their use is independent of the classifiers selected [19]. Random under-sampling removes the desired number of majority class instances randomly from the data set. Since this method throws out majority class instances it discards useful information for better learning [31], [53]. Other famous methods of under-sampling include Tomek links [50], condensed nearest neighbour rule (CNN) [28], Wilson's edited nearest neighbour rule (ENN) [54], and one sided selection (OSS) [36]. There are also several proposed methods in literature that combines over-sampling and under-sampling techniques such as SMOTE+ENN [3], SMOTE+Tomek [3], and CNN+Tomek links [3].

On the other hand, in random over-sampling (ROS), 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 the minority class instances and very specific rules that this kind of over-sampling generates can lead the algorithm to over-fit and over-generalize [41], [53].

Synthetic minority over-sampling technique, SMOTE, was proposed by Chawla et al. [7]. 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. SMOTE is reported to have the over-generalization problem [51]. Also, it generates synthetic samples regardless of the majority class; consequently, the over-lapping between different classes increase remarkably [41]. Since SMOTE algorithm generates synthetic minority examples regardless of the majority class regions, it causes the decision boundaries for the minority class to spread further into the majority class space and worsens the problem of overlapping between classes especially in multi-class cases [3]. This problem leads to misclassification of majority class examples into minority class and increases the false positive rate of the learning algorithms. In order to overcome the SMOTE difficulties, many researches are proposed in literature [6], [18], [25], [30], [42]; however, all these techniques are designed for two-class problems and their performances have not been evaluated on multi-class problems.

Safe-level SMOTE [6] is another variant of SMOTE algorithm that tries to overcome the SMOTE difficulties. It carefully generates minority examples along the same line with different weight degree, known as the safe-level. This method assigns each minority class instance its safe level before over-sampling and tries to generate examples in safe regions. The safe level is defined as the number of positive instances in K nearest neighbours. If the safe-level of an example is close to zero, it is considered as noise and if this degree is close to K it is considered as safe. The experiments are conducted on two-class data sets and the results are compared with SMOTE and Borderline SMOTE algorithms.

In [18] research, the authors proposed an over-sampling technique base on using large margin principle and SMOTE algorithm, which is called MSYN. This paper focuses on the margins of nearest neighbour rule and tries to overcome the over-generalization in algorithms. In this case, the hypothesis-margin lower bounds the sample-margin and the hypothesis margin of an instance x with regards to the set of instances A can be computed as follows: $\theta_A(x) = \frac{1}{2}(||x - nearest - miss_A(x)|| - ||x - nearest - hit_A(x)||$ in which $nearest - miss_A(x)$ is the nearest example to x in set A with the same label as x and $nearest - hit_A(x)$ is the nearest sample to x in set A with different label. The MSYN algorithm generates new synthetic examples via SMOTE and then chooses the needed number of them according to criterion. This criterion minimizes the margin loss for the major class and maximizes margin gains for the minority class. The performance of proposed algorithm is evaluated on two-class imbalance problems and the results are compared with SMOTE, Borderline-SMOTE, ROS, and ADASYN algorithms. Although MSYN tries to avoid over-generalization, it uses all features to select a synthetic example [42].

In order to overcome SMOTE and MSYN difficulties, Puntumapon and Waiyamai proposed a method, named TRIM [42], to select precise seed subsets of particular features. TRIM searches for a precise minority class regions and iteratively filters out irrelevant majority data. The algorithm outputs multiple subset of seed minority class examples and generates synthetic data samples via SMOTE. The performance of TRIM is compared with SMOTE and MSYN [18] on several two-class data sets.

Borderline SMOTE [25] 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. Although Borderline-SMOTE performs better than SMOTE algorithm in two-class data sets, it has difficulties in multi-class imbalanced problems. Since in these data sets overlapping between class regions may be very high, the border line examples should be recognized accurately. Borderline-SMOTE also suffers from over-generalization problem [42].

Adaptive Synthetic Sampling (ADASYN) [30] adaptively creates different amount of synthetic samples according to their distribution. It first calculates the number of synthetic instances that need to be generated for the minority class: $G = (|S_{major}| - |S_{minor}|) \times \beta$, where $\beta \in [0, 1]$ is a parameter to control the desired level of balance, $|S_{major}|$ is the number of majority class instances, and $|S_{major}|$ is the number of minority class instances under consideration (do not limited to the class with minimum number of examples). For each minority class sample $x_i \in S_{minor}$, its K-nearest neighbours are calculated and the ratio of Γ_i is defined as: $\frac{\Delta_i/K}{Z}, \quad i = 1, \ldots, |S_{minor}|$. 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 need to be synthesized for each $x_i \in S_{minor}$ is computed as follows: $g_i = \Gamma_i \times G$. Finally, for each $x_i \in |S_{minor}|$ ADASYN generates g_i synthetic data with SMOTE algorithm.

Fernández et al. provides thorough experimental analyses on the behaviour of several pair-wise and ad hoc procedures in multi-class imbalanced problems [19]. They used class decomposition schemes, one versus one (OAO) [29] and one versus all (OAA) [44], in order to apply the two-class methods on multi-class imbalanced problems. The average accuracy rate (mean of accuracy over all the classes) is employed in all experiments to evaluate the performance of methods. In OAO, if there are c classes of data, each of them is considered against every one of the other classes. As a result, $c(c-1)/2$ two-class problems should be learned. 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 positive class and all other classes are considered as negative. In this case, if the original training data is imbalanced already, the problem may get even worse. The authors indicated the good performance of these pair wise techniques in multi-class problems. In experimental part of the paper, we compare the results of our MDO technique with random over-sampling, SMOTE, ADASYN, and Borderline-SMOTE algorithms. Random over-sampling, in spite of its simplicity, is considered as an effective method in comparison with other complex techniques in literature [19], [38]. Due to the nature of MDO and the way it over-samples and preserves the original training examples variance structure, we over-sampled different minority class examples in SMOTE algorithm separately to provide a fair comparison. Moreover, in ADASYN and Borderline-SMOTE algorithms, OAA approach is employed because of its lower computational complexity (in comparison with OAO) in large multi-class data sets. Besides, the potential drawback of OAA in over-sampling is negligible, since we over-sample each of the minority classes separately to reach the number of majority class examples and finally acquire a balanced training data. However, investigating the performance of different two-class over-sampling techniques accompanying with OAA and OAO methods comparing with MDO technique should be regarded in future research work.

Fernández-Navarro et al. [20] proposed a dynamic over-sampling procedure for improving the classification of imbalanced data sets with more than two classes. This procedure is incorporated into a memetic algorithm (MA) that optimizes radial basis functions neural networks (RBFNNs). To handle class imbalance, the training data are re-sampled in two stages. In the first stage, an over-sampling procedure is applied to the minority class to balance in part the size of the classes. Then, the MA is run and the data re-over-sampled in different generations of the evolution, generating new patterns of the minimum sensitivity class (the class with the worst accuracy for the best RBFNN of the population).

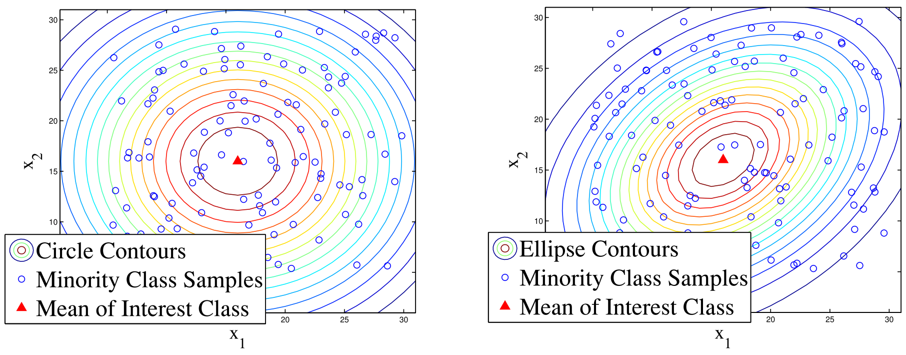


Fig. 1. Simulated data for two variables x_1, x_2 with the circle (left) and ellipse contours (right), representing equal Euclidean and Mahalanobis distance from the centre mean, respectively.

Lin et al. [38] proposed a dynamic sampling method, called DyS, for multilayer perceptrons (MLP). In DyS, for each epoch of the training process, every example is fed to the current MLP and then the probability of it being selected for training the MLP is estimated, using a heuristic. DyS dynamically selects informative data to train the MLP.

Imbalanced data stream mining problems are received great attentions recently by many researches [10], [11], [12]. Among different class imbalance methods in literature, there are some researches which used Mahalanobis distance in their algorithms [10], [12], [46], [49]. In paper [10], the authors proposed a framework, called SERA, which is a two-class technique to handle non-stationary imbalanced data streams. In SERA algorithm the minority class examples from the previous data chunks are absorbed into the current data chunk. Instead of adding the entire minority examples of previous chunk into current one, SERA applies a post balance ratio to accommodate minority examples of previous chunks into the current one by measuring similarity between the minority examples of these chunks. The Mahalanobis distance is applied as a measure of similarity.

Inspired by their previous SERA algorithm, Chen et al. in [12] proposed an ensemble learning algorithm, named MuSeRA, to handle the problem of imbalanced data streams. MuSeRA unlike SERA generates multiple hypotheses build on previous data chunks and all of them are weighted to classify the label of the testing data. At a certain time, a defined amount of minority examples that are received so far are evaluated and added to the current data chunk. The evaluation is based on the Mahalanobis distance between the current minority examples and the previously received minority examples. Then a hypothesis is built over the current balanced data chunk and added into the set of trained hypotheses so far. Finally, the learned hypotheses are weighted to make prediction on the unseen testing data.

3 MDO: Mahalanobis Distance-Based Over-Sampling Technique

The proposed over-sampling technique is based on Mahalanobis distance [39]. The idea is that "we can generate new synthetic instances which have the same Mahalanobis distance from their corresponding class mean". Fig. 1 indicates a simulated two dimensional data. The circle and ellipse contours represent equal Euclidean and Mahalanobis distance to the centre point of the considered class, respectively. In calculation of Mahalanobis distance, the correlations of the data are also taken into account and in this way, Mahalanobis distance differs from Euclidean distance.

The Mahalanobis distance of a multivariate vector $x = (x_1, x_2, x_3, \dots, x_d)^T$ from a group of values with mean $\mu = (\mu_1, \mu_2, \mu_3, \dots, \mu_d)^T$ and covariance matrix S is defined as:

$$D_M(x) = \sqrt{(x-\mu)^T S^{-1} (x-\mu)} \ . \tag{1}$$

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 instance 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 preserve the covariance structure of the original minority 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 equation:

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

In general, for d dimensional ellipse with origin of the Cartesian coordinates as its centre point, we can generalize equation (2) as follows:

$$\frac{(x_1^2)}{a_1^2} + \frac{(x_2^2)}{a_2^2} + \cdots + \frac{(x_i^2)}{a_i^2} + \cdots + \frac{(x_d^2)}{a_d^2} = 1 \ . \tag{3}$$

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 equation (3) as follows:

$$\frac{(x_1^2)}{\alpha.V_1} + \frac{(x_2^2)}{\alpha.V_2} + \cdots + \frac{(x_i^2)}{\alpha.V_i} + \cdots + \frac{(x_d^2)}{\alpha.V_d} = 1 \ . \tag{4}$$

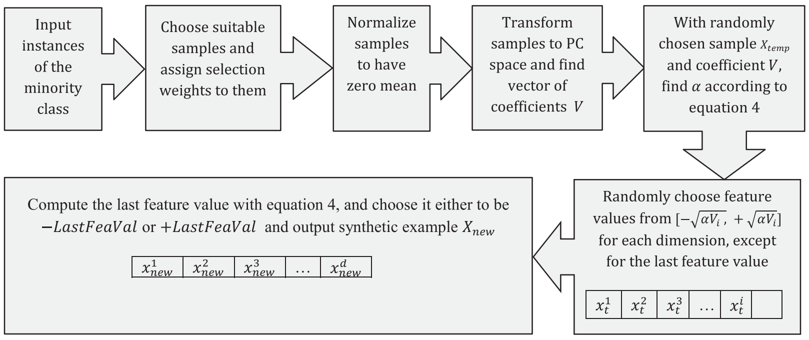

Input instances of the minority class
Choose suitable samples and assign selection weights to them
Normalize samples to have zero mean
Transform samples to PC space and find vector of coefficients V
With randomly chosen sample X_{temp} and coefficient V, find α according to equation 4
Randomly choose 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.

We will use equation (4) to generate new synthetic examples for the minority classes. The block diagram of the algorithm is provided in Fig. 2.

The pseudo code for MDO over-sampling technique is presented in Algorithm 1. Consider a minority class of i with class label of L. In steps 3 to 6 of the Algorithm 1, the number of minority classes t and a minority class set S_{minor} are considered. In step 7, MDO identifies the suitable minority class samples from the original minority set S_{minor}, and creates a new set S_{Cminor}. The detailed explanation of the *chooseSample* function is provided in Algorithm 2.

Algorithm 1. MDO over-sampling technique

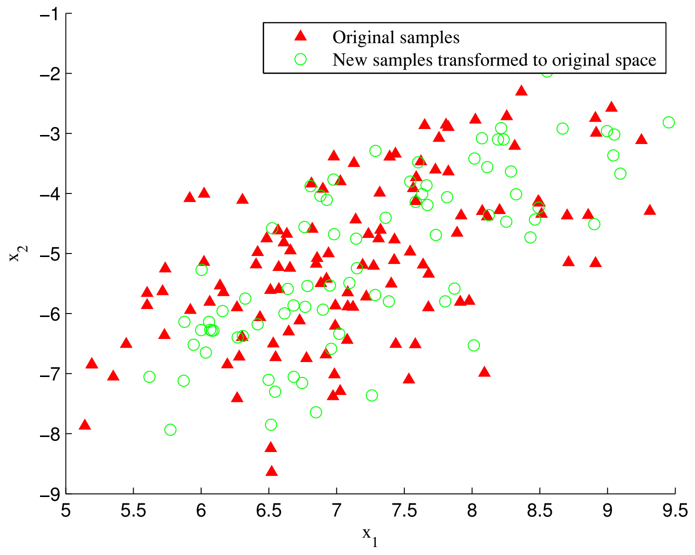
```
1: Input: Data set $S = \{(x_1, y_1), \ldots, (x_m, y_m)\}$ $x_i \in X; y_i \in Y = \{1, \ldots, c\}$; Number of majority class instances $n_{maj}$.
2: Output: A balanced set of data.
3: $t$ = Number of minority classes;
4: $S_{New} = \emptyset$ ;
5: for $i = 1$ to $t$ do
6:     $S_{minor_i}$ = instances of class $i$;
7:     $[S_{Cminor_i}, weights_i]$ = chooseSamples($S, S_{minor_i}$);
       /* chooseSamples function takes the whole sample set $S$ and considered class samples $S_{minor_i}$ as its inputs and returns two outputs, selected samples $S_{Cminor_i}$ and their corresponding selection weights $weights_i$.*/
8:     $n_i$ = number of instances in $S_{Cminor_i}$;
9:     $\mu_i = \frac{1}{n_i}\sum_{j=1}^{n_i} x_i^j$; /* class mean of samples in $S_{Cminor_i}$ */
10:    $Z_i$ = replace each $x_i^j$ with $x_i^j - \mu_i$; /* make samples to have zero mean */
11:    Find the principal components of $Z_i$ using eigenvalue decomposition;
12:    $T_i$ = transform $Z_i$ to corresponding PC space and make them uncorrelated;
13:    $V$ = diagonal(covariance($T_i$)); */A vector of coefficients. */
14:    $Orate_i = n_{maj} - n_i$;
15:    $S_{temp}$ = MDO-oversampling($T_i, V, Orate_i$);
16:    Transform the newly generated samples $S_{temp_i}$ to the original space;
17:    Add the class mean $\mu_i$ to the new samples;
18:    Add new samples to $S_{New}$;
19: end for
20: Return balanced set of data;
```

In Algorithm 2, the data points which are located in the dense areas of the feature space and have more nearest neighbours with the class label of L are identified. For each of the samples in S_{minor}, MDO finds its $K2$ nearest neighbours from the whole sample set S. If the data point has more than $K1$ nearest neighbours with class label of L, MDO considers it as a suitable data point for over-sampling and gives it a selection weight.

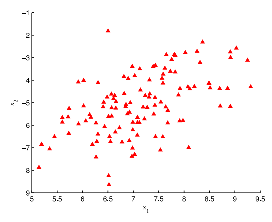
Algorithm 2. chooseSamples function

```
1: Input: Whole data set $S = \{(x_1, y_1), \ldots, (x_m, y_m)\}$ and considered class set, $S_{minor_i}$.
2: Output: Selected samples of class $i$, $S_{Cminor_i}$; and their corresponding weights, $weights_i$.
3: $m_i$ = number of instances in $S_{minor_i}$;
4: for $j = 1$ to $m_i$ do
5:     $X_j = temp_i(j)$; /* $jth$ sample of the set $S_{minor_i}$*/
6:     for $k = 1$ to $m$ do
7:         $X_k = S(k)$; /* $kth$ sample of the set $S$ */
8:         $Dist(k) = EuclideanDistance(X_j, X_k)$;
9:     end for
10:    $temp$ = select $K2$ nearest neighbours of $X_j$;
11:    $num$ = number of nearest neighbours of $X_j$ in $temp$ which has the same label as $X_j$;
12:    if $num \geq K1$ then
13:        Add $X_j$ to $S_{Cminor_i}$; /* if the number of nearest neighbours of $X_j$ in $temp$ is greater than $K1$, $X_j$ is selected as a suitable sample for over-sampling */
14:        $weights(j) = \frac{num}{K2}$;
15:    end if
16: end for
17: Normalized the weights $weights$, so that $weights_i$ is a density distribution;
18: Return weights $weights_i$ and chosen samples $S_{Cminor_i}$;
```

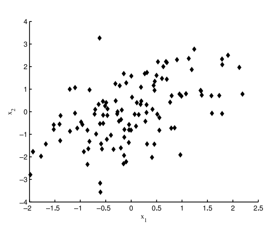
In steps 8 to 10 of the Algorithm 1, each new sample set of the class i, i.e., S_{Cminor_i}, is normalized to have zero mean. The principal components of the class i instances are computed via eigenvalue decomposition in step 11. In step 12, the instances of the considered class are transformed to PC space. The corresponding coefficients, which are the variances of the data in each dimension, are computed in step 13. These coefficients together with the α parameter, form the denominators of the ellipse equation (4). Over-sampling rate parameter $Orate_i$ which is the size difference between the number of majority and minority class instances, is computed in step 14. In step 15, *MDO-oversampling* function is called; the pseudo code of this function is presented in Algorithm 3.



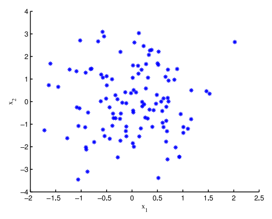
(a) Orginal and synthetic samples in one figure.



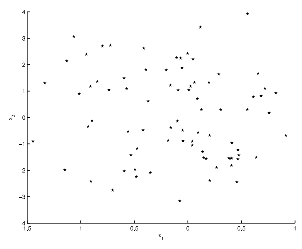
(b) Original samples



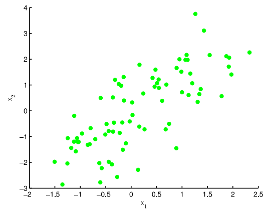
(c) Zero mean samples



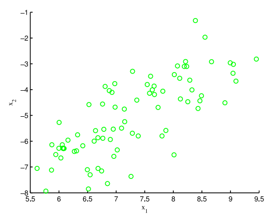
(d) Transformed samples to PC space



(e) Generated samples via MDO



(f) New samples transformed to original space



(g) Added by mean of the class

Fig. 3. Illustrates step-by-step MDO over-sampling technique just for the considered minority class.

In Algorithm 3, each of the minority class samples in S_{Cminor_i} has a selection weight according to the number of its nearest neighbours having the same class label (which is calculated in *chooseSamples* function). MDO selects a random minority class sample X_{temp} according to weight distribution $weights_i$ and then through steps 4 to 15 of the Algorithm 2, it generates a new synthetic sample X_{new} using ellipse equation (4). This newly generated sample is then added to the set S_{temp}.

In steps 16 to 18 of the Algorithm 1, synthetic samples which are generated in PC space, are transformed to the original space and they are added by the calculated class mean.

3.1 An Illustrative Example of MDO Technique

Fig. 3 shows a step-by-step example of calculating synthetic samples, using MDO over-sampling technique. In this example, a two-dimensional data set with five classes is considered. After choosing suitable examples for over-sampling, they are normalized and mapped to PC space. The new synthetic samples are generated according to equation (4) and transformed to the original space. Finally, they are added by their class mean. In Fig. 4, the resulting samples together with majority and minority class instances are illustrated.

3.2 Computational Complexity of the Proposed Method

In MDO over-sampling technique, first the considered samples are normalized to have zero mean. Then, they are mapped to PC space via eigenvalue decomposition; next, the 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 instances. 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 literature which can reduce eigenvalue computation to matrix multiplication or other less expensive methods. For example, Demmel et al. [15]

indicate that all standard linear algebra operations, such as linear equation solving, matrix inversion, (generalised) 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 equation (4) two times, which is reduced to solving a linear equation. The first equation, which is the calculation of α, is a linear equation that 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.

Algorithm 3. MDO over-sampling function

```
1:  Input: Transformed samples $T_i$, vector of coefficients $V$,
    over-sampling rate $Orate_i$.
2:  Output: Synthetic samples of class $i$, $S_{temp}$.
3:  $Num_{attrs}$ = number of attributes or the data dimension;
4:  while $Orate_i \neq 0$ do
5:      $X_{temp}$ = choose a random sample from $T_i$ according to
        $weights_i$;
6:      $X$ = square $X_{temp}$;
7:      Compute $\alpha$ from ellipse equation (4), with $V$ and $X$;
8:      $AlphaV = \alpha \times V$; /* A vector results from $\alpha \times V$, which
        forms the denominators of equation (4) */
9:      Set $s = 0$;
10:     for $j = 1$ to $(Num_{attrs} - 1)$ do
11:         $r$ = choose a random number from
              $\left[-\sqrt{AlphaV(j)} \quad , \quad \sqrt{AlphaV(j)}\right]$
            for the $j$th dimension of the new sample;
12:         $s = s + (r^2/AlphaV(j))$;
13:     end for
14:     Solve ellipse equation (4), with $s$ and $AlphaV(last)$ and
        find the last feature value, $LastFeaVal$;
15:     Set last feature value randomly from
              $\{+LastFeaVal, -LastFeaVal\}$;
16:     Add synthetic sample $X_{new}$ to $S_{temp}$;
17: end while
18: Return synthetic samples $S_{temp}$;
```

3.3 In-Depth Overview of the Proposed Method

In this section we try to review the formulation of our proposed method and indicate that why we used equation (4) to over-sample minority class instances.

- *Euclidean distance*. The Euclidean distance between two points $x = (x_1, \ldots, x_d)^T$ and $y = (y_1, \ldots, y_d)^T$ in the *d-dimensional* space $\Re^d$ is defined as:

$$D_E(x, y) = \sqrt{(x_1 - y_1)^2 + \cdots + (x_d - y_d)^2}$$
$$= \sqrt{(x - y)^T (x - y)}.$$

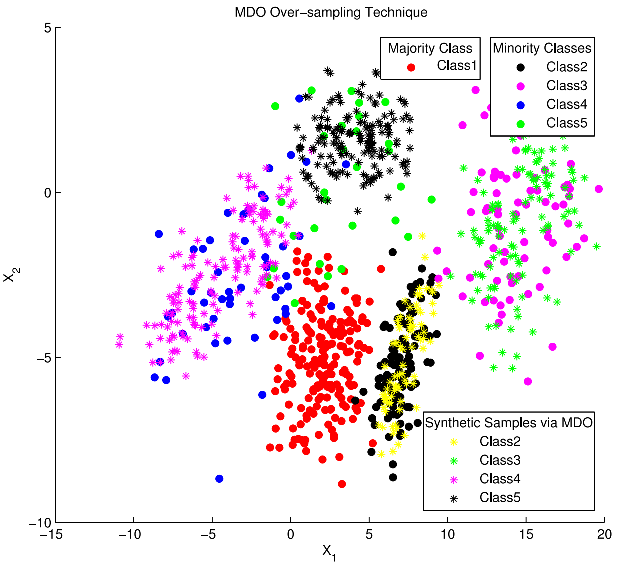


Fig. 4. Illustrates five-class artificial data set. Original minority/majority class instances, together with the synthetic minority class samples. The filled circles are original data samples and stars are synthetic samples via MDO technique.

It indicates that all observations with the same Euclidean distance from the origin $\|x\|_2 = c$ satisfy $x_1^2 + \cdots + x_d^2 = c^2$ which represents the equation of a spheroid. In other words, all components of a considered point x contribute equally to the Euclidean distance of x. But, in statistics when we want to determine the distance of a variable from its origin, we prefer 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 achieved by rescaling the components. Denote $u = (\frac{x_1}{s_1}, \ldots, \frac{x_d}{s_d})$ and $v = (\frac{y_1}{s_1}, \ldots, \frac{y_d}{s_d})$ then define the distance between x and y as:

$$D(x, y) = D_E(u, v) = \sqrt{\frac{(x_1 - y_1)^2}{s_1} + \cdots + \frac{(x_d - y_d)^2}{s_d}}$$
$$= \sqrt{(x - y)^T S^{-1} (x - y)},$$

where

$$S = diag(s_1^2, \ldots, s_d^2).$$

Now the norm of x equals:

$$\|x\| = D(x, 0) = D_E(u, 0) = \|u\|_2$$
$$= \sqrt{\frac{(x_1)^2}{s_1} + \cdots + \frac{(x_d)^2}{s_d}} = \sqrt{x^T S^{-1} x}$$

and all observations with the same distance from the centre $\|x\| = c$ satisfy: $(\frac{x_1}{s_1})^2 + \cdots + (\frac{x_d}{s_d})^2 = c^2$, which indicates the equation of an ellipsoid centred at the origin with principal axes equal to the coordinate axes. We also want to take the *correlation* between different variables into account when computing distances which means associations between variables. As a result, we want the axes of ellipsoid to reflect this correlation. This can be obtained by allowing the axes of the ellipsoid at constant distance to rotate. This results the following general form for the statistical distance of two considered points:

- *Mahalanobis distance*. The Mahalanobis distance between two data points $x = (x_1, \ldots, x_d)^T$ and $y = (y_1, \ldots, y_d)^T$ in the *d-dimensional* space $\Re^d$ is defined as:

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

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 points may differ from the origin and the distance of a point from its centre μ given by:

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

In equation (4), we substitute all "V" coefficients with variance of each dimension in order to parametrize the ellipse equation and express each radius of the ellipse with the corresponding variance. In order 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 equation (4). We used the concept of Mahalanobis distance instead of Euclidean distance to over-sample the minority class instances, because we want to preserve the covariance structure of the data in minority classes; and in this way, we will be able to generate suitable synthetic samples for learning algorithms.

4 Experimental Settings

Our proposed method is compared with other state-of-the-art methods, namely, SMOTE, random over-sampling, Borderline-SMOTE, and ADASYN. With respect to the parameter settings, the nearest neighbour parameter K is set to 5 in all methods; the most accepted value in the literature. In ADASYN over-sampling method, β parameter is set to 1, as it was suggested in [30]. In MDO $K1$ and $K2$ parameters are set to 5 and 10, respectively. These values are selected after some preliminary runs. The rate of over-sampling for a class c is the size difference between the majority class and class c. Each minority class is over-sampled as equally as the size of the majority class in all methods.

4.1 Data Sets and Base Learners

The performance of our over-sampling method for multi-class imbalanced problems is evaluated using 20 multi-class benchmark data sets from the UCI [21] and KEEL [2] 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. The smallest data set is *Breast-tissue* with 106 instances and the largest one is *Letter* with $20,000$ samples. The imbalanced ratio (IR in short) of these data sets varies between 1.09 and 439.6. Imbalance ratio or IR is computed as the proportion of the number of majority class examples to the number of minority class examples [40]. The class with maximum number of instances is the majority class and the class with minimum number of examples is the minority class. Also, the percentage of minority classes are between 0.102 and 29.96 percent. The number of classes in these benchmarks varies between 3 and 26.

We use three different base classifiers for our experiments which are implemented in Weka [55]. The following classifiers are used: a) C4.5 decision tree [43] which uses normalized information gain splitting criterion from information theory [1]; b) RIPPER [14], 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 [34]; c) KNN classifier, which is an instanced-based or lazy learning algorithm, is also used in our experiments. Parameter K in KNN classifier is set to 5. For other classifiers the default Weka parameters are used. As some data sets have very small classes of data, we perform stratified five-fold cross-validation with 10 independent runs to be sure that each fold of test or train contains at least one example of each class.

4.2 Assessment Metrics

Traditionally, the most widely used metrics of learning algorithms are accuracy and error rate. However, they can be deceiving in certain situations and are highly sensitive to changes in data [31]. Precision, recall, and F-measure [45] are the most prevalently used single-class measures for two-class problems. We computed these single-class metrics for the minority class, solely. Also, MAUC and G-mean metrics are used for multi-class evaluation. Precision is a measure of exactness, i.e., from among the examples labelled as positive, how many are actually labelled correctly [31]. On the other hand, recall is a measure of completeness, i.e., how many examples of the positive class are labelled correctly. F-measure incorporates both measures, precision and recall, to express the trade-off between them. Parameter β which is set to one in most cases and in our case is a coefficient to adjust the relative importance of these two metrics [31],

$$Precision = \frac{TP}{TP + FP} \quad . \tag{5}$$

$$Recall = \frac{TP}{TP + FN} \quad . \tag{6}$$

$$F\text{-}measure = \frac{(1 + \beta^2) Recall.Precision}{\beta^2.Recall + Precision} \quad . \tag{7}$$

Extended G-mean [35] for multi-class cases which was adapted by Sun et al. [47] is provided in equation (8). It is defined as the geometric mean of the recall over all classes. Given a *c-class* problem:

$$G\text{-}mean = \left(\prod_{i=1}^{c} Recall_i \right)^{\frac{1}{c}} \quad . \tag{8}$$

TABLE 1
Description of 20 UCI and KEEL Benchmark Data Sets with Their Characteristics, Including Number of Classes, Data Set Size, Class Distribution, Number of Features, Imbalance Ratio (IR), Percentage of Minority Classes (%Min), and Minority/Majority Class Names

Data set	Class	Size	Class Distribution	Features	IR	%Min	Minority Class	Majority Class
Breast-tissue	6	106	22/21/18/16/15/14	9	1.57	13.2%	Class of 'con','gla', 'mas','fad','car'	Class of 'adi'
Hayes-roth	3	132	51/51/30	4	1.7	22.72%	Class of '3'	All other classes
Wine	3	178	71/59/48	13	1.48	29.96%	Class of '1','3'	Class of '2'
Autos	5	202	67/54/32/27/22	15	3.04	10.89%	Class of '1','2','3','-1'	Class of '0'
Glass	6	214	76/70/29/17/13/9	10	8.44	4.2%	Class of '3','4','5'	Class of '1','2'
New-thyroid	3	215	150/35/30	5	5	13.95%	Class of 'hypo','hyper'	Class of 'normal'
Voice	3	238	168/52/18	9	9.33	7.56%	Class of '1','2'	Class of '3'
User Knowledge-Modelling (UKM)	4	258	88/83/63/24	5	3.66	9.3	Class of '1','2','4'	Class of '3'
Cleveland	5	303	164/55/36/35/13	13	12.62	4.29%	Class of '1','2','3','4'	Class of '5'
Vertebral-column	3	310	150/100/60	6	2.5	19.35%	Class of 'Hernia', 'Normal'	Class of 'Spondylolisthesis'
Ecoli	5	327	143/77/52/35/20	7	7.15	6.11%	Class of 'im', 'imU', 'om','pp'	Class of 'cp'
Pageblocks	4	545	492/33/12/8	10	61.5	1.46%	Class of '2','3','4'	Class of '1'
Balance	3	625	288/288/49	4	5.88	7.84%	Class of 'B'	All other classes
Vehicle	4	846	218/217/212/199	18	1.09	23.52%	Class of 'saab','van', 'opel'	Class of 'bus'
Contraceptive	3	1,473	629/511/333	9	1.89	22.61	Class of '2','3'	Class of '1'
Yeast	10	1,484	463/429/244/163/ 51/44/35/30/20/5	8	92.6	0.34%	Class of 'MIT','NUC', 'ME1','ME2', 'ME3', 'EXC','VAC', 'POX','ERL'	Class of 'CYT'
winequality-red	6	1,599	981/638/199/53/18/10	11	68.1	0.63%	Class of '1','2','3', '4','6'	'5'
Abalone	18	4,139	689/634/568/487/ 391/267/259/203/ 126/115/103/67/ 58/57/42/32/26/15	7	45.93	0.36%	Class of '1','2','3', '4','5','6','7', '8','10', '11','12','13,'14','15', '16','17','18'	Class of '9'
winequality-white	7	4898	2198/1457/880/ 175/163/20/5	11	439.6	0.102%	Class of '1','2', '3','4','5'	Class of '6'
Letter	26	20000	813/805/803/796/ 792/789/787/786/ 783/783/775/773/ 768/766/764/761/ 758/755/753/752/748/ 747/739/736/734/734	16	1.11	3.67%	All other classes	Class of '21'

MAUC or M-measure [26], [48] is the average of AUC [4], [27] over all pairs of classes. It is defined as follows:

$$MAUC = \frac{2}{c(c-1)} \sum_{i<j} \frac{[A(i,j) + A(j,i)]}{2}. \qquad (9)$$

Where $A(i,j)$ is the AUC between class i and class 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 $A(i,j)$ and $A(j,i)$ may 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

In order to evaluate the statistically significant differences of the results from the comparative methods, we conduct the Friedman test [23] with the corresponding post-hoc test which was recommended by Demšar [16]. The Friedman test is a non-parametric statistical method for ranking all the algorithms over all data sets separately. It is worth mentioning that, in general, due to some drawbacks of parametric tests such as paired t-test, non-parametric tests are more suitable for statistically comparing the performance of the desirable methods [16].

Ranking the comparative methods over multiple data sets is an appropriate way of evaluation [16]. Friedman test compares the mean ranks of the considered techniques. The best performing algorithm getting the rank of 1, the second best rank of 2, and etc. In tie cases the average ranks are assigned to the considered methods. Then, these ranks are averaged to compute the mean rank of performance for each algorithm. Therefore, smaller value indicates a higher rank of performance. If the null hypothesis, i.e., all the algorithms are equivalent, is rejected then we proceed with the post-hoc test to find out which algorithms actually differ. We used an improved Friedman test which was proposed by Iman and Davenport [32]. The Bonferroni-Dunn test [17] is used as the post-hoc test method. Tables 2, 3, 4, 5, 6, and 7

TABLE 2
Mean Ranks of Five Comparative Methods over All Data Sets with C4.5 as the Base Classifier

Rank	*ROS*	*MDO*	*SMOTE*	*B-S*	*ADASYN*
MAUC	3.6250	1.1750	2.2250	4.4500	3.5250
G-mean	2.9250	1.9500	2.3750	4.0000	3.7500
Precision	3.1750	1.7250	3.2750	3.4750	3.3500
Recall	3.0000	2.0250	2.7250	4.1750	3.0750
F-measure	2.9750	1.9000	2.7250	4.1000	3.3000

TABLE 3
Friedman Test with the Corresponding Post-Hoc Test, Bonferroni-Dunn for Five Comparative Methods over All Data Sets with C4.5 as the Base Classifier

	Friedman	*ROS*	*SMOTE*	*B-S*	*ADASYN*
MAUC	Reject	**2.4500**	1.0500	**3.2750**	**2.3500**
G-mean	Reject	0.9750	0.4250	**2.0500**	**1.8000**
Precision	Reject	**1.4500**	**1.5500**	**1.7500**	**1.6250**
Recall	Reject	0.9750	0.7000	**2.1500**	1.0500
F-measure	Reject	1.0750	0.8250	**2.2000**	**1.4000**

A value greater than the CD (CD = 1.2490*) indicate statistically significant differences between the methods, which are highlighted in boldface.*

TABLE 4
Mean Ranks of Five Comparative Methods over All Data Sets with KNN as the Base Classifier

Rank	*ROS*	*MDO*	*SMOTE*	*B-S*	*ADASYN*
MAUC	3.6500	1.2000	2.6500	4.4500	3.0500
G-mean	2.7500	2.1000	2.4750	3.8250	3.8500
Precision	3.8250	1.4000	2.9250	3.8250	3.0250
Recall	2.3250	2.7500	2.4500	3.8250	3.6500
F-measure	3.2750	1.9000	2.4250	4.0250	3.3750

present the results of Friedman and post-hoc tests on MAUC, G-mean, precision, recall, and F-measure for comparative methods over 20 data sets. In our experiments the MDO over-sampling technique is selected as the "control" method to be compared with other algorithms. Each value in post-hoc tables (i.e., Tables 3, 5, and 7) indicates the difference of the mean ranks between the "control" method and the other technique in the corresponding column. The CD value, which stands for critical difference, is determined by the number of competing algorithms and data sets; also, critical value $q_{\alpha=0.05}$ is equal to 2.4980 in our experiments. If the difference of mean ranks of two comparative method is greater than the CD, the performance of these two algorithms is statistically significant. These significant differences are highlighted in boldface in post-hoc tables. In all cases the value of the CD is 1.2490.

- *The comparative methods with C4.5 classifier*: In Tables 2 and 3, the results of Friedman test and the corresponding post-hoc test with C4.5 classifier are reported. As it is shown in Table 3, the Friedman test compares average ranks over null hypothesis. We used F_F test that is based on Friedman's χ^2_F statistic. In all cases there are critical differences, so we reject the null hypothesis and proceed with the post-hoc test.

TABLE 5
Friedman Test with the Corresponding Post-Hoc Test, Bonferroni-Dunn for Five Comparative Methods over All Data Sets with KNN as the Base Classifier

	Friedman	*ROS*	*SMOTE*	*B-S*	*ADASYN*
MAUC	Reject	**2.4500**	**1.4500**	**3.2500**	**1.8500**
G-mean	Reject	0.6500	0.3750	**1.7250**	**1.7500**
Precision	Reject	**2.4250**	**1.5250**	**2.4250**	**1.6250**
Recall	Reject	0.4250	0.3000	1.0750	0.9000
F-measure	Reject	**1.3750**	0.5250	**2.1250**	**1.4750**

A value greater than the CD (CD = 1.2490*) indicate statistically significant differences between the methods, which are highlighted in boldface.*

TABLE 6
Mean Ranks of Five Comparative Methods over All Data Sets with Ripper as the Base Classifier

Rank	*ROS*	*MDO*	*SMOTE*	*B-S*	*ADASYN*
MAUC	3.6500	1.2000	2.5500	4.2000	3.4000
G-mean	2.6750	2.1000	2.7750	3.9250	3.5250
Precision	3.2000	1.4500	3.3500	3.6500	3.3500
Recall	3.0250	2.1000	2.4500	4.2250	3.2000
F-measure	3.1000	1.6000	2.7000	4.0000	3.6000

TABLE 7
Friedman Test with the Corresponding Post-Hoc Test, Bonferroni-Dunn for Five Comparative Methods over All Data Sets with Ripper as the Base Classifier

	Friedman	*ROS*	*SMOTE*	*B-S*	*ADASYN*
MAUC	Reject	**2.4500**	**1.3500**	**3.0000**	**2.2000**
G-mean	Reject	0.5750	0.6750	**1.8250**	**1.4250**
Precision	Reject	**1.7500**	**1.9000**	**2.2000**	**1.9000**
Recall	Reject	0.9250	0.3500	**2.1250**	1.1000
F-measure	Reject	**1.5000**	1.1000	**2.4000**	**2.0000**

A value greater than the CD (CD = 1.2490*) indicate statistically significant differences between the methods, which are highlighted in boldface.*

  We noticed that the best performing algorithm in terms of precision with average ranks of 1.7250 is MDO, which significantly outperforms all other over-sampling techniques. In terms of G-mean and F-measure, also, the best performing algorithm with average rank of 1.9500 and 1.9000 is MDO which is significantly better than Borderline-SMOTE, and ADASYN. In terms of MAUC, MDO ranked first with average rank of 1.1750 outperforming random over-sampling, Borderline-SMOTE, and ADASYN significantly. For recall metric, MDO outperforms Borderline-SMOTE significantly.
- *The comparative methods with KNN classifier*: Tables 4 and 5 present the results of the conducted tests with KNN classifier. In this case, the best performing algorithm in terms of MAUC and precision, with average ranks of 1.200 and 1.400, is MDO. MDO significantly outperforms all other over-sampling techniques in terms of MAUC and precision metrics. In terms of G-mean, the best performing algorithm with average rank of 2.1000 is MDO which is significantly better than Borderline-SMOTE and ADASYN.

In terms of F-measure, MDO with average rank of 1.9000 significantly outperforms random over-sampling, Borderline-SMOTE, and ADASYN.

- *The comparative methods with Ripper classifier*: Tables 6 and 7 indicate the results of the Friedman and post-hoc tests with Ripper classifier. According to the mean ranks, MDO ranked first in terms of all evaluation measures. It outperforms random over-sampling, ADASYN, and Borderline-SMOTE in terms of F-measure significantly. In terms of MAUC and precision with average rank of 1.2000 and 1.4500, it outperforms all comparative techniques. For recall metric, MDO preforms significantly better than Borderline-SMOTE. In G-mean with average rank of 2.1000, MDO outperforms Borderline-SMOTE and ADASYN significantly.

Considering all the results, MDO significantly outperforms all other over-sampling techniques in terms of precision with all classifiers. In terms of MAUC, MDO is significantly better than other methods with KNN and Ripper classifiers. In terms of the other evaluation metrics, also, MDO performs better than other techniques generally, which indicates remarkably lower average rank. The best performing classifiers with MDO technique are KNN and Ripper, which indicate statistically significant difference of performance in MAUC and minority class precision.

MAUC evaluates the average ability of separating any pair of classes [52]. A high MAUC indicates that a learner is good at separating most class pairs; however, it is still possible that some classes are difficult to be distinguished from the other classes. MDO causes the classifier to build more specific decision regions which contain nearby minority class points. MDO makes the learner to recognize class pairs better; this results in significant improvements in MAUC, against all other over-sampling methods. G-mean, on the other hand, indicates how well a classifier can balance the recognition among different classes [52]. G-mean is the geometric mean of recall over all classes. If the recall over a class is zero, this metric becomes zero. A high G-mean guarantees that no class is ignored [52]. MDO, which is tuned to decrease the overlapping between class regions, can recognize different class instances better; this, in turn, increases the G-mean metric.

It is reported that over-sampling methods like SMOTE cause degradation in precision due to increased number of false positive samples [9]. These over-samplings cause the algorithm to over-generalize and this, in turn, causes the degradation of precision. In terms of precision, MDO performs significantly better than other techniques. It is probably related to the algorithm; because the training samples are modelled better in MDO, the probability of over-fitting or over-generalization of the classifier with over-sampled instances decreases considerably; and this, in turn, indicates significantly better precision.

We can infer that over-sampling the minority class instances properly increases the overall performance of a classifiers. In all three cases of our experiments, MDO indicates great results in terms of different evaluation metrics; especially MAUC and precision. We can conclude that balancing the instances of multi-class imbalanced data sets with a simple and effective over-sampling technique yields significantly better results and imposes less complexity on learning algorithms.

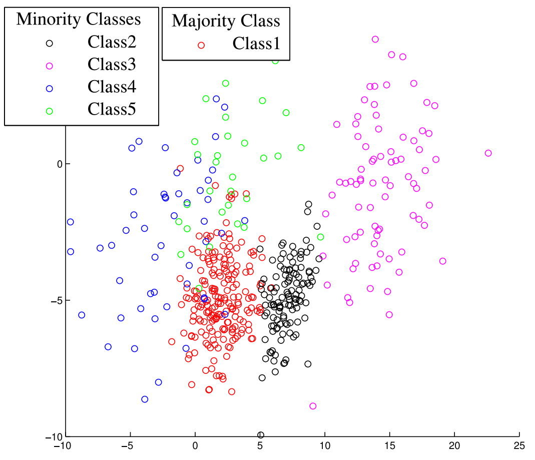
4.4 An Empirical Elaboration on the Performance of MDO and Other Over-Sampling Techniques

This section investigates the performance of our proposed over-sampling method and the other comparative over-sampling techniques to provide a better understanding of these algorithms. Furthermore, this comparison may help us support our claims about MDO and be an answer to the question that why we expect MDO to produce promising results. Due to the way MDO over-samples, new synthetic minority class samples preserve the original minority class covariance structure. Specifically in multi-class cases, due to the existing several minority/majority classes with different underlying distributions, many existing over-sampling techniques produce synthetic samples. This enlarges the minority class regions wrongly. Consequently, the minority class regions fall inside the majority class areas and worsen the situation for learning algorithms.

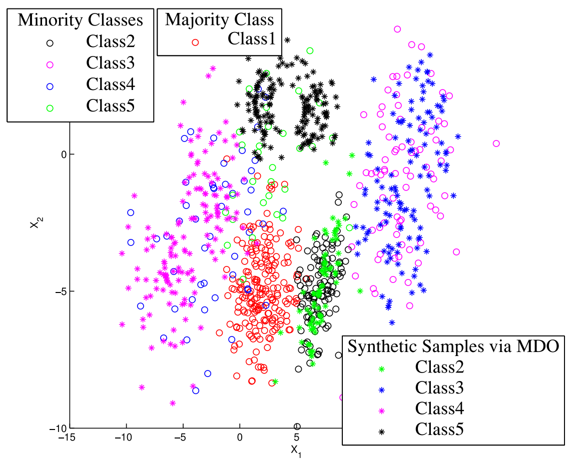
The mean of the selected samples of each candidate class can be a good representative for the corresponding class instances in MDO algorithm. MDO over-samples new examples along the probability contours and in dense areas of the feature space. In other words, MDO considers a minority class sample suitable for over-sampling if it has more nearest neighbours of the same class. In this way MDO helps to reduce the overlapping between different class regions. Simply put, MDO generates a new synthetic sample from instances which settle in dense areas of the considered class regions.

In order to investigate the performances of MDO and other over-sampling techniques, we compare these algorithms using a five-class artificial data set. In Fig. 5, original instances of the data set together with over-sampled examples via different methods are depicted. For example, consider the class 5 of the depicted minority class (green circles) in Fig. 5 part (a): the synthetic samples of SMOTE, Borderline-SMOTE, and ADASYN techniques fall inside the decision regions of the *class 1* (red circles), *class 2* (black circles), and *class 4* (blue circles). The overlapping between these considered classes already exists in original data (part (a) of the Fig. 5; however, SMOTE, ADASYN, and Bordeline-SMOTE increased the overlapping between these three class regions and this, in turn, worsens the situation for learning algorithms.

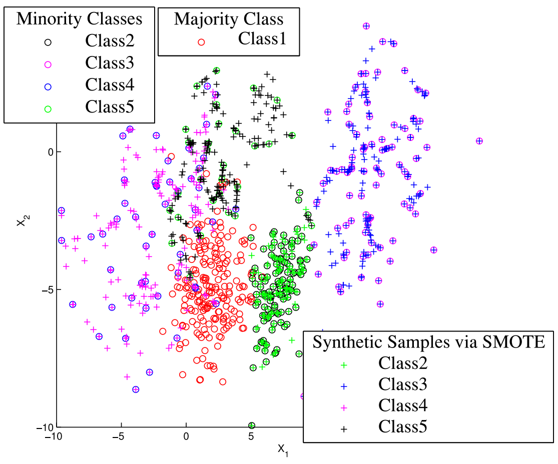
According to the Fig. 5, the MDO over-sampled instances are generated in the broader area of the considered class region; however, this way of generating synthetic samples does not increase the overlapping between different class regions as much as other techniques. Therefore, the decision regions that a learner may estimate in this data space are very generalized in comparison with other techniques. Also, the generated synthetic samples via MDO provide more information for the learning model than others algorithms. SMOTE, Borderline-SMOTE, and ADASYN generate synthetic samples between different class regions and worsen the situation for learning tasks. Moreover, theses algorithms generate more duplicated and overlapped data points which do not provide useful information for learning



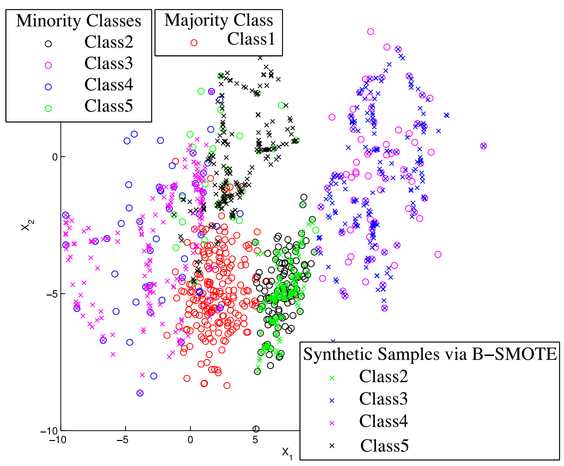
(a) Original data set



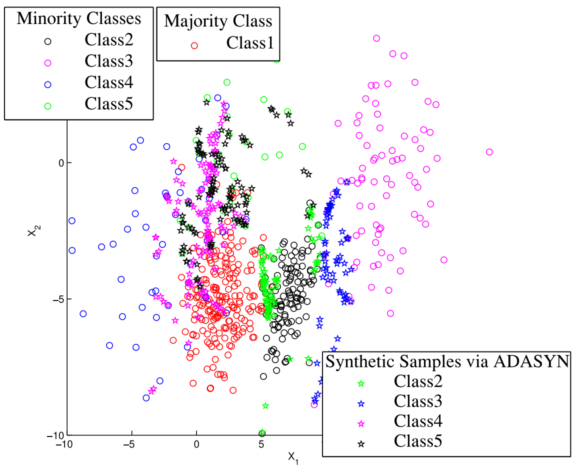
(b) MDO



(c) SMOTE



(d) Boderline-SMOTE



(e) ADASYN

Fig. 5. Scatter plots of a five-class artificial data set together with over-sampled instances with MDO, SMOTE, Borderline-SMOTE, and ADASYN over-sampling techniques.

models. The way that MDO generates synthetic examples makes the classifier increase its generalization ability. That, in turn, makes MDO to recognize the minority class examples better; hence, better minority class recall and precision than other techniques. This way of creating new artificial examples also helps the classifier in recognizing different class pairs better. The result is significantly better MAUC.

5 Conclusion and Future Work

In this paper we proposed a new over-sampling technique, called MDO, for multi-class imbalanced problems. In MDO method, the synthetic samples are generated in such a way that they have the same Mahalanobis distance from their corresponding class mean. These synthetic samples are generated toward the variation of the corresponding class. By considering the mean of each class and generating synthetic samples in dense areas of the minority class regions, MDO can reduce the risk of overlapping between different classes. MDO not only improves the generalization ability of a classifier remarkably, but it also does not cause the classifier to over-fit or over-generalize as much as the other over-sampling techniques. In order to examine the performance of our method, we compared our results with other over-sampling techniques. Twenty UCI and KEEL multi-class imbalanced data sets and three different classifiers were used. MDO outperformed exiting over-sampling techniques in terms of MAUC and precision with KNN and Ripper classifiers, significantly. Based on the analyses, we concluded that over-sampling methods improve the performance of learning from multi-class imbalanced data sets, significantly. Although over-sampling techniques can perform well in most cases, generating synthetic samples properly, i.e., within each class region, can improve the performance much more. In many applications, over-sampling methods should be used with suitable classifiers to yield the best results.

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 we can conduct a pre-process task and get the most informative features and generate new samples from a subset of features. 2) All data sets which we used for our experiments have numeric (i.e., real) attributes. In our future work, we decided to study different possibilities of adapting MDO to handle nominal and mixed-type (numeric/nominal) attributes too. 3) Using another distance metrics like Hellinger distance [13] which indicates advantageous properties in imbalance problems, to over-sample the minority class instances may get superior results. 4) Finally, it is interesting to study the performance of MDO and other two-class over-sampling techniques with OAA and OAO to better investigate the effects of over-sampling methods on multi-class imbalance problems.

Acknowledgments

The authors would like to express the deepest gratitude to Mr. Farhad Abdi for his constructive advice and assistance for editing this paper.

References

[1] J. Aczél and Z. Daróczy, *On Measures of Information and Their Characterizations*. New York, NY, USA: Academic, 1975.

[2] J. Alcalá, A. Fernández, J. Luengo, J. Derrac, S. García, L. Sánchez, and F. Herrera, "KEEL data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework," *J. Multiple-Valued Logic Soft. Comput.*, vol. 17, pp. 255–287, 2010.

[3] G. E. A. P. A. Batista, R. C. Prati, and W. C. Monard, "A study of the behavior of several methods for balancing machine learning training data," *ACM Sigkdd Explorations Newslett.*, vol. 6, no. 1, pp. 20–29, 2004.

[4] A. P. Bradley, "The use of the area under the roc curve in the evaluation of machine learning algorithms," *Pattern Recognit.*, vol. 30, no. 7, pp. 1145–1159, 1997.

[5] L. Breiman, "Bagging predictors," *Mach. Learn.*, vol. 24, no. 2, pp. 123–140, 1996.

[6] C. Bunkhumpornpat, K. Sinapiromsaran, and C. Lursinsap, "Safe-level-smote: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem," in *Proc. 13th Pacific-Asia Conf. Adv. Knowl. Discovery Data Mining*, 2009, pp. 475–482.

[7] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, no. 16, pp. 341–378, 2002.

[8] N. V. Chawla, N. Japkowicz, and A. Kotcz, "Editorial: Special issue on learning from imbalanced data sets," *ACM Sigkdd Explorations Newslett.*, vol. 6, no. 1, pp. 1–6, 2004.

[9] N. V. Chawla, A. Lazarevic, L. O. Hall, and K. W. Bowyer, "SMOTEBoost: Improving prediction of the minority class in boosting," in *Proc. 7th Eur. Conf. Principles Practice Knowl. Discovery Databases*, 2003, pp. 107–119.

[10] S. Chen and H. He, "Sera: Selectively recursive approach towards nonstationary imbalanced stream data mining," in *Proc. Int. Joint Conf. Neural Netw.*, 2009, pp. 522–529.

[11] S. Chen and H. He, "Towards incremental learning of nonstationary imbalanced data stream: A multiple selectively recursive approach," *Evolving Syst.*, vol. 2, no. 1, pp. 35–50, 2011.

[12] S. Chen, H. He, K. Li, and S. Desai, "Musera: Multiple selectively recursive approach towards imbalanced stream data mining," in *Proc. Int. Joint Conf. Neural Netw.*, 2010, pp. 1–8.

[13] D. A. Cieslak, T. R. Hoens, N. V. Chawla, and W. P. Kegelmeyer, "Hellinger distance decision trees are robust and skew-insensitive," *Data Mining Knowl. Discovery*, vol. 24, no. 1, pp. 136–158, 2012.

[14] W. W. Cohen, "Fast effective rule induction in machine learning," in *Proc. 12th Int. Conf. Mach. Learn.*, 1995, pp. 115–123.

[15] J. Demmel, I. Dumitriu, and O. Holtz, "Fast linear algebra is stable," *Numerische Mathematik*, vol. 108, no. 1, pp. 59–91, 2007.

[16] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *The J. Mach. Learn. Res.*, vol. 7, pp. 1–30, 2006.

[17] O. J. Dunn, "Multiple comparisons among means," *J. Am. Statist. Assoc.*, vol. 56, no. 293, pp. 52–64, 1961.

[18] X. Fan, K. Tang, and T. Weise, "Margin-based over-sampling method for learning from imbalanced datasets," in *Proc. 15th Pacific-Asia Conf. Adv. Knowl. Discovery Data Mining*, 2011, pp. 309–320.

[19] A. Fernández, V. López, M. Galar, M. D. Jesus, and F. Herrera, "Analysing the classification of imbalanced data-sets with multiple classes: Binarization techniques and ad-hoc approaches," *Knowl.-Based Syst.*, vol. 42, pp. 97–110, 2013.

[20] F. Fernández-Navarro, C. Hervás-Martínez, and P. Antonio Gutiérrez, "A dynamic over-sampling procedure based on sensitivity for multi-class problems," *Pattern Recognit.*, vol. 44, no. 8, pp. 1821–1833, 2011.

[21] A. Frank and A. Asuncion. (2010). UCI machine learning repository [Online]. Available: http://archive.ics.uci.edu/ml

[22] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," in *Proc. Int. Conf. Mach. Learn.*, 1996, vol. 96, pp. 148–156.

[23] M. Friedman, "The use of ranks to avoid the assumption of normality implicit in the analysis of variance," *J. Am. Statist. Assoc.*, vol. 32, no. 200, pp. 675–701, 1937.

[24] V. García, J. S. Sánchez, and R. A. Mollineda, "On the effectiveness of preprocessing methods when dealing with different levels of class imbalance," *Knowl.-Based Syst.*, vol. 25, no. 1, pp. 13–21, 2012.

[25] H. Han, W.-Y. Wang, and B.-H. Mao, "Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning," in *Proc. Int. Conf. Adv. Intell. Comput.*, 2005, pp. 878–887.

[26] D. J. Hand and R. J. Till, "A simple generalisation of the area under the roc curve for multiple class classification problems," *Mach. Learn.*, vol. 45, no. 2, pp. 171–186, 2001.

[27] J. A. Hanley and B. J. McNeil, "The meaning and use of the area under a receiver operating characteristic (roc) curve," *Radiology*, vol. 143, no. 1, pp. 29–36, 1982.

[28] P. E. Hart, "The condensed nearest neighbor rule (corresp.)," *IEEE Trans. Inf. Theory*, vol. 14, no. 3, pp. 515–516, May 1968.

[29] T. Hastie, R. Tibshirani, et al., "Classification by pairwise coupling," *The Ann. Statist.*, vol. 26, no. 2, pp. 451–471, 1998.

[30] H. He, Y. Bai, E. A. Garcia, and S. Li, "ADASYN: Adaptive synthetic sampling approach for imbalanced learning," in *Proc. IEEE Int. Joint Conf. Neural Netw (IEEE World Congress Comput. Intell).*, 2008, pp. 1322–1328.

[31] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 9, pp. 1263–1284, Sep. 2009.
[32] R. L. Iman and J. M. Davenport, "Approximations of the critical region of the fbietkan statistic," *Commun. Statist.-Theory Methods*, vol. 9, no. 6, pp. 571–595, 1980.
[33] M. V. Joshi, V. Kumar, and R. C. Agarwal, "Evaluating boosting algorithms to classify rare classes: Comparison and improvements," in *Proc. IEEE Int. Conf. Data Mining*, 2001, pp. 257–264.
[34] S. B. Kotsiantis, I. Zaharakis, and P. Pintelas, "Supervised machine learning: A review of classification techniques," pp. 3–24, 2007.
[35] M. Kubat, R. Holte, and S. Matwin, "Learning when negative examples abound," in *Proc. 9th Eur. Conf. Mach. Learn.*, 1997, pp. 146–153.
[36] M. Kubat, S. Matwin, et al., "Addressing the curse of imbalanced training sets: One-sided selection," in *Proc. 14th Int. Conf. Mach. Learn.*, 1997, vol. 97, pp. 179–186.
[37] T. W. Liao, "Classification of weld flaws with imbalanced class data," *Expert Syst. Appl.*, vol. 35, no. 3, pp. 1041–1052, 2008.
[38] M. Lin, K. Tang, and X. Yao, "Dynamic sampling approach to training neural networks for multiclass imbalance classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 4, pp. 647–660, Apr. 2013.
[39] P. C. Mahalanobis, "On the generalized distance in statistics," in *Proc. Nat. Instit. Sci.*, 1936, vol. 2, pp. 49–55.
[40] A. Orriols-Puig and E. Bernadó-Mansilla, "Evolutionary rule-based systems for imbalanced data sets," *Soft Comput.*, vol. 13, no. 3, pp. 213–225, 2009.
[41] R. C. Prati, G. E. A. P. A. Batista, and M. C. Monard, "Class imbalances versus class overlapping: An analysis of a learning system behavior," in *Proc. Adv. Artif. Intell.*, 2004, pp. 312–321.
[42] K. Puntumapon and K. Waiyamai, "A pruning-based approach for searching precise and generalized region for synthetic minority over-sampling," in *Proc. 16th Pacific-Asia Conf. Adv. Knowl. Discovery Data Mining*, 2012, pp. 371–382.
[43] J. R. Quinlan, *C4.5: Programs for Machine Learning*, vol. 1. San Mateo, CA, USA: Morgan Kaufmann, 1993.
[44] R. Rifkin and A. Klautau, "In defense of one-vs-all classification," *The J. Mach. Learn. Res.*, no. 5, pp. 101–141, 2004.
[45] C. V. Rijsbergen, *Information Retrieval*. London, U.K.: Butterworth, 1979.
[46] W. A Rivera, A. Goel, and J. P. Kincaid, "Oups: A combined approach using smote and propensity score matching," in *Proc. 13th Int. Conf. Mach. Learn. Appl.*, 2014, pp. 424–427.
[47] Y. Sun, M. S. Kamel, and Y. Wang, "Boosting for learning multiple classes with imbalanced class distribution," in *Proc. 6th Int. Conf. Data Mining*, 2006, pp. 592–602.
[48] K. Tang, R. Wang, and T. Chen, "Towards maximizing the area under the ROC Curve for multi-class classification problems," in *Proc. 25th AAAI Conf. Artif. Intell.*, 2011, pp. 483–488.
[49] J. Tian, H. Gu, and W. Liu, "Imbalanced classification using support vector machine ensemble," *Neural Comput. Appl.*, vol. 20, no. 2, pp. 203–209, 2011.
[50] I. Tomek, "Two modifications of CNN," *IEEE Trans. Syst., Man Cybern.*, vol. 6, no. 11, pp. 769–772, Nov. 1976.
[51] B. X. Wang and N. Japkowicz, "Imbalanced data set learning with synthetic samples," in *Proc. IRIS Mach. Learn. Workshop*, Ottawa, Canada, Jun. 2004.
[52] S. Wang and X. Yao, "Multiclass imbalance problems: Analysis and potential solutions," *IEEE Trans. Syst., Man Cybern.*, vol. 42, no. 4, pp. 1119–1130, Aug. 2012.
[53] G. M. Weiss and F. Provost, "The Effect of Class Distribution on Classifier Learning: An Empirical Study," Department of Computer Science, Rutgers University, New Jersey, Tech. Rep. ML-TR-44, 2001.
[54] D. Wilson, "Asymptotic properties of nearest neighbor rules using edited data," *IEEE Trans. Syst., Man Cybern.*, no. 3, pp. 2:408–421, 1972.
[55] I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*. San Mateo, CA, USA: Morgan Kaufmann, 2005.
[56] X.-M. Zhao, X. Li, L. Chen, and K. Aihara, "Protein classification with imbalanced data," *Proteins: Structure, Function Bioinf.*, vol. 70, no. 4, pp. 1125–1132, 2008.
[57] Z.-H. Zhou and X.-Y. Liu, "Training cost-sensitive neural networks with methods addressing the class imbalance problem," *IEEE Trans. Knowl. Data Eng.*, vol. 18, no. 1, pp. 63–77, Jan. 2006.

Lida Abdi received the BSc degree in computer engineering from Shiraz Payamnoor University in 2010, and the MSc degree in artificial intelligence from Shiraz University, Iran, in 2013. Her research interests include machine learning, data mining, and class imbalance learning.

Sattar Hashemi received the PhD degree in computer science from the Iran University of Science and Technology in conjunction with Monash University, Australia, in 2008. Following academic appointments at Shiraz University, he is currently an associate professor in the Electrical and Computer Engineering School, Shiraz University, Shiraz, Iran. He is recognized for contributions in the fields of machine learning and data mining. He has published many refereed papers and book chapters on data stream classification, game theory, social networks, database intrusion detection, and computer security.