Expert Systems With Applications 46 (2016) 405–416

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Adaptive semi-unsupervised weighted oversampling (A-SUWO) for imbalanced datasets

Iman Nekooeimehr, Susana K. Lai-Yuen*

Industrial and Management Systems Engineering, University of South Florida, 4202 East Fowler Avenue, ENB 118, Tampa, Florida 33620, USA

ARTICLE INFO

Keywords:
Imbalanced dataset
Classification
Clustering
Oversampling

ABSTRACT

In many applications, the dataset for classification may be highly imbalanced where most of the instances in the training set may belong to one of the classes (majority class), while only a few instances are from the other class (minority class). Conventional classifiers will strongly favor the majority class and ignore the minority instances. In this paper, we present a new oversampling method called Adaptive Semi-Unsupervised Weighted Oversampling (A-SUWO) for imbalanced binary dataset classification. The proposed method clusters the minority instances using a semi-unsupervised hierarchical clustering approach and adaptively determines the size to oversample each sub-cluster using its classification complexity and cross validation. Then, the minority instances are oversampled depending on their Euclidean distance to the majority class. A-SUWO aims to identify hard-to-learn instances by considering minority instances from each sub-cluster that are closer to the borderline. It also avoids generating synthetic minority instances that overlap with the majority class by considering the majority class in the clustering and oversampling stages. Results demonstrate that the proposed method achieves significantly better results in most datasets compared with other sampling methods.

1. Introduction

Many datasets in various applications are imbalanced where some classes contain many more instances than others. Some examples where imbalanced datasets need to be classified include detection of fraudulent bank account transactions or telephone calls (Akbani, Kwek, & Japkowicz, 2004; Wei, Li, Cao, Ou, & Chen, 2013), biomedical diagnosis (He & Garcia, 2009; Li, Chan, Fu, & Krishnan, 2014), text classification (Zheng, Wu, & Srihari, 2004), information retrieval and filtering (Piras & Giacinto, 2012) and college student retention (Thammasiri, Delen, Meesad, & Kasap, 2014). In two-class problems, the class that contains many instances is the majority class whereas the class that contains fewer instances is the minority class. When the dataset is imbalanced, conventional classifiers typically favor the majority class thus failing to classify the minority observations correctly and resulting in performance loss (Prati, Batista, & Silva, 2014). When the training data is highly imbalanced, the minority class may not even be detected. This kind of imbalance that exists between two different classes is called *between-class* imbalance. Another kind of imbalance that results in performance loss is *within-class* imbalance, which happens when the minority or majority instances have more than one concept (sub-cluster of data) and some of these concepts have less number of instances than others. In addition, the presence of high overlapping among the concepts is another factor that leads to classifiers' performance loss on minority instances (Alshomrani, Bawakid, Shim, Fernández, & Herrera, 2015). Current methods developed for imbalanced dataset problems do not address both *within-class* imbalance and *between-class* imbalance at the same time. Most of these methods also exacerbate the overlapping among the concepts after trying to address the imbalance problem.

Traditionally, the objective of supervised learning is to optimize the accuracy for the whole dataset, which may cause the classifier to ignore the performance on each individual class. In particular, in an imbalanced dataset, if a random classifier predicts all instances as the majority class, a very high accuracy can be achieved despite incorrectly classifying all minority instances. Therefore, it is strongly suggested to use measurements that are suitable for imbalanced dataset classification.

In this paper, a new oversampling method called Adaptive Semi-Unsupervised Weighted Oversampling (A-SUWO) is presented for imbalanced binary dataset classification. A-SUWO finds hard-to-learn instances by first clustering the minority instances and then assigning higher weights to those instances from each sub-cluster that are closer to the majority class. This approach enables the identification of all instances that are close to the decision boundary and

* Corresponding author. Tel.: +18139745547.
E-mail addresses: nekooeimehr@mail.usf.edu (I. Nekooeimehr), laiyuen@usf.edu (S.K. Lai-Yuen).

http://dx.doi.org/10.1016/j.eswa.2015.10.031
0957-4174/© 2015 Elsevier Ltd. All rights reserved.

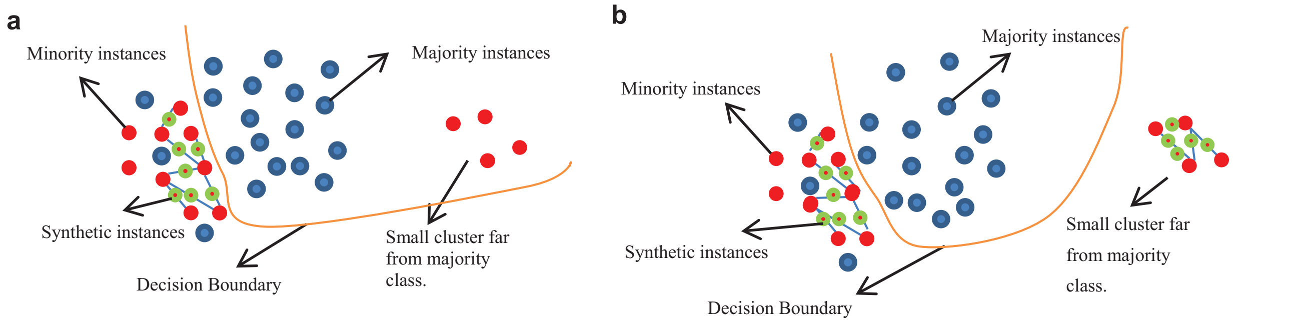

Fig. 1. (a) Minority cluster that is far from the majority class is ignored and not oversampled. (b) All minority clusters are considered for oversampling based on their misclassification complexity.

also considers all sub-clusters, even small ones, for oversampling as shown in Fig. 1(b). A-SUWO avoids over-generalization using two strategies. First, it clusters the minority instances by considering the majority class to reduce overlapping between the generated minority instances and majority instances. A semi-unsupervised hierarchical clustering approach is proposed that iteratively forms minority sub-clusters while avoiding majority sub-clusters in between. Second, it oversamples minority instances based on their average Euclidean distance to majority instances to further decrease the chance of generating overlapping instances between classes. In addition, A-SUWO determines sub-cluster sizes adaptively based on their misclassification error. In our method, misclassification error is an indication of sub-cluster complexity and is determined using a new measurement based on the standardized average error rate and cross validation. Sub-clusters with higher misclassification error will be assigned a larger size while the ones with lower misclassification error will be assigned a smaller size.

In order to validate A-SUWO, an extensive experimental design is performed. The proposed A-SUWO method is evaluated on 16 publicly available datasets, classified using 4 classifiers and compared with eight other oversampling techniques. F-measure, G-mean and AUC are used as the performance measures. The performance measures are determined using 4-fold stratified cross validation and repeated three times.

The remainder of this paper is organized as follows. In the next section, a review of related previous works is presented. In Section 3, the A-SUWO methodology is described. Section 4 presents the experimental design and results, while conclusions are provided in Section 5.

2. Previous work

There is an increasing interest in addressing imbalanced dataset classification. These works can be categorized into four main types of techniques: data preprocessing, algorithmic modification, cost-sensitive learning, and ensemble of classifier sampling methods (Díez-Pastor, Rodríguez, García-Osorio, & Kuncheva, 2015; Galar, Fernandez, Barrenechea, Bustince, & Herrera, 2012). Although there is no one single method that works well for all imbalanced dataset problems, sampling methods have shown great potential as they attempt to improve the dataset itself rather than the classifier (Barua, Islam, Yao, & Murase, 2014), (Chawla, Bowyer, Hall, & Kegelmeyer, 2011), (Han, Wang, & Mao, 2005), (Yen & Lee, 2009). Sampling methods change the distribution of each class observation by either oversampling the minority samples or undersampling the majority samples. In the case of oversampling, sampling methods generate new minority instances to balance the dataset and in the case of undersampling, they remove some majority instances from the dataset. Undersampling methods have shown to be less efficient than oversampling methods because the removal of majority instances may eliminate important information from the dataset, especially in cases where the dataset is small (He, Bai, Garcia, & Li, 2008; Japkowicz & Stephen, 2002; Zhou, 2013).

The simplest oversampling method is random sampling. It randomly selects a minority instance and duplicates it until the minority class reaches a desired size. Random oversampling generates new instances that are very similar to the original instances resulting in over-fitting. To overcome this problem, Chawla et al. developed Synthetic Minority Oversampling Technique (SMOTE) (Chawla et al., 2011) where new synthetic instances are generated between randomly selected minority instances and their *NN*-nearest neighbors, where *NN* is a user-defined variable. However, this may cause over-generalization as the new instances are generated without considering the majority instances thus increasing the overlap between minority and majority classes (He & Garcia, 2009), (Yen & Lee, 2009), (López, Fernández, García, Palade, & Herrera, 2013). Over-generalization can be exacerbated when the dataset has higher imbalance ratio as the instances of the minority class are very sparse and can become contained within the majority class after oversampling. This can further deteriorate subsequent classification performance (Kotsiantis, Kanellopoulos, & Pintelas, 2006).

Various approaches have been proposed to address over-generalization. Safe-level SMOTE (Bunkhumpornpat, Sinapiromsaran, & Lursinsap, 2009) presents a method that calculates a “safe-level” value for each minority instance, then generates synthetic instances closer to the largest safe level. The safe-level value is defined as the number of other minority instances among its *NN*-nearest neighbors. Safe-level SMOTE can cause overfitting because synthetic instances are forced to be generated farther from the decision boundary. Borderline-SMOTE (Han et al., 2005) presents a method to identify the borderline between the two classes, and oversamples only the minority samples on the borderline. ADASYN (He et al., 2008) assigns weights to minority instances so that those that have more majority instances in their neighborhood have higher chance to be oversampled. However, Borderline-SMOTE and ADASYN do not find all the minority instances close to the decision boundary (Barua et al., 2014). MWMOTE (Barua et al., 2014) approaches this problem by presenting a two-step procedure to find candidate majority border instances and then candidate minority border instances. Then, weights are assigned to candidate minority instances based on their Euclidean distances to the candidate majority border instances so that those with higher weights have a higher chance to be oversampled. However, small concepts of minority instances that are far from the majority class are not detected even if they may contain important information as shown in Fig. 1(a). In general, it is necessary to find hard-to-learn instances to be used for oversampling because they contain important information for the classifier. These instances are usually near the decision boundary or belong to small concepts (He & Garcia, 2009; Japkowicz & Stephen, 2002). The presence of small concepts in the dataset is referred to as *within-class* imbalance.

Recently, clustering-based methods (Bunkhumpornpat, Sinapiromsaran, & Lursinsap, 2012; Cieslak & Chawla, 2008; Cieslak, Chawla, & Striegel, 2006; Jo & Japkowicz, 2004; Yen & Lee, 2009) have been presented to address *within-class* imbalance. Generally, these methods decompose the dataset into several smaller sub-clusters and use sampling methods to increase or decrease their size. In (Jo & Japkowicz, 2004), each class is clustered separately so oversampling is performed on all the sub-clusters of the same class to make their size equal. Cluster-SMOTE (Cieslak et al., 2006) first clusters the minority class into *m* sub-clusters using k-means algorithm and then applies SMOTE to each of them. Under-sampling based on Clustering (SBC) (Yen & Lee, 2009) method first clusters the whole dataset into *m* sub-clusters, then for each of them, it computes the ratio of the number of majority instances to the number of minority instances. Finally, their method removes majority instances based on the ratio, i.e., they remove more majority instances from sub-clusters with lower ratio while they keep more majority instances from the ones with higher ratio. However, removing instances from the dataset may remove important information. In (Cieslak & Chawla, 2008), the dataset is partitioned using the Hellinger distance and for each partition, the majority instances are undersampled while the minority instances are oversampled to reach a desired imbalance ratio. In (Bunkhumpornpat et al., 2012), the minority class is clustered into several arbitrary shaped sub-clusters, and the synthetic instances are generated between the minority instances and their corresponding sub-cluster's pseudo-centroids. However, these methods do not identify instances that are close to the decision boundary and do not consider the classification complexity of the sub-clusters when determining the level to which each sub-cluster should be oversampled.

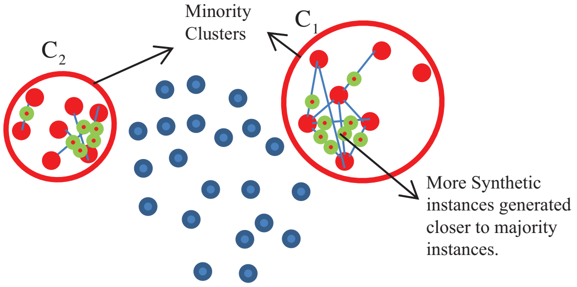
Minority
Clusters
C_2
C_1
More Synthetic instances generated closer to majority instances.

Fig. 4. Synthetic instances (red dots with green outline) are generated between original minority instances (red dots) where the generated instances are closer to majority instances (blue dots). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article).

3. Adaptive semi-unsupervised weighted oversampling (A-SUWO)

A-SUWO consists of three main steps: (1) Semi-Unsupervised Clustering, (2) Adaptive Sub-cluster Sizing, and (3) Synthetic Instance Generation. In the first step, the minority instances are clustered using a semi-unsupervised hierarchical clustering approach that iteratively groups minority instances while avoiding majority groups in between. In the Adaptive Sub-cluster Sizing step, the size to which each minority sub-cluster will be oversampled is determined based on its complexity in being classified (misclassification error). A new measurement based on the standardized average error rate is

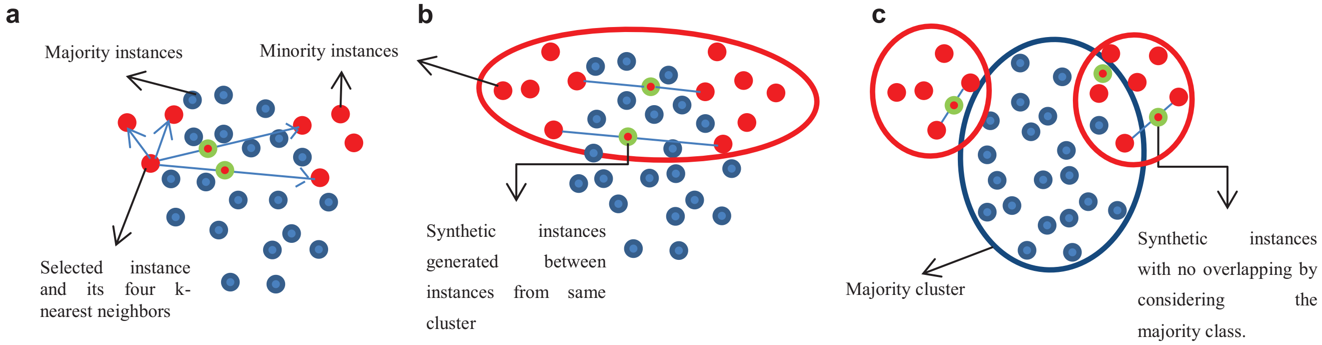
a
Majority instances
Minority instances
Selected instance and its four k-nearest neighbors
b
Synthetic instances generated between instances from same cluster
c
Majority cluster
Synthetic instances with no overlapping by considering the majority class.

Fig. 2. Approaches for new instance (red dots with green outline) generation based on minority instances (red dots) and majority instances (blue dots): (a) between selected instances and two of its 4-nearest neighbors; (b) between instances of the same cluster; and (c) between selected instances and its 4-nearest neighbors provided that they belong to the same cluster. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article).

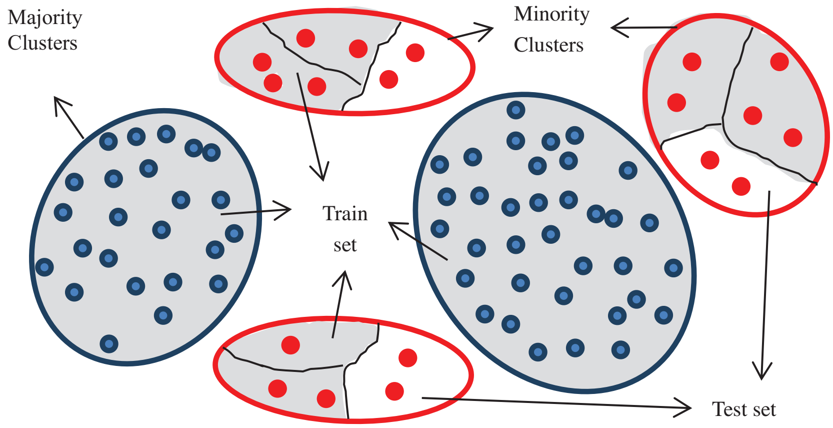

Fig. 3. Adaptive minority cluster size identification for oversampling based on misclassification error and cross validation. Majority samples (blue dots) and minority samples (red dots). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article).

proposed to determine the sub-cluster's complexity and is calculated using cross validation. Finally, in the Synthetic Instance Generation step, a new weighting system is proposed to assign weights to minority instances based on their average Euclidean distance to their *NN*-nearest majority class neighbors so that synthetic instances are generated based on these weights.

3.1. Semi-unsupervised clustering

In general, there are two approaches for generating synthetic instances. The first one is to generate a new instance between a candidate instance and one of its *NN*-nearest neighbors (Chawla et al., 2011; Han et al., 2005; He et al., 2008). The second approach is to

Table 1
Description of the datasets.

#	Dataset	Minority class	Majority class	# of features	# of instances	# of minority instances	# of majority instances	Imbalanced ratio
1	Vehicle	Class "van"	All other	17	846	199	647	1:3.25
2	Ecoli	Class "pp"	All other	7	336	52	259	1:4.98
3	Pima	Class "1"	Class "0"	8	768	268	500	1:1.87
4	Balance	Class "2"	All other	4	625	49	576	1:11.76
5	Liver disorders	Class "1"	Class "2"	6	345	145	200	1:1.38
6	Wine	Class "2"	All other	13	178	71	130	1:1.83
7	Breast tissue	Class "car" and "fad	All other	9	106	36	70	1:1.94
8	Libra	Class "1", "2", "3"	All other	90	360	72	288	1:4:00
9	LEV	Class "1"	All other	4	1000	93	907	1:9.75
10	Iris	Class "2"	All other	4	150	50	100	1:2.00
11	Heart	Class "1"	Class "-1"	13	270	120	150	1:1.25
12	Glass	Class "1"	All other	9	214	70	138	1:1.97
13	Haberman	Class "2"	Class "1"	3	306	81	225	1:2.78
14	Eucalyptus	Class "5"	All other	91	736	105	631	1:6.01
15	Heating	Class "6", "7", "8"	All other	8	768	201	567	1:2.82
16	Segment	Class of "WINDOW"	All other	18	2310	330	1980	1:6.00

Table 2
Results for the sampling methods on the 16 datasets classified using SVM.

Dataset	Meas.	Random	SMOTE	Borderline SMOTE	Safe-level SMOTE	SBC	Cluster SMOTE	CBOS	MWMOTE	A-SUWO
Vehicle	F_M	0.955 ± 0.014	0.953 ± 0.019	0.959 ± 0.010	0.954 ± 0.014	0.917 ± 0.026	0.858 ± 0.099	0.955 ± 0.015	0.948 ± 0.010	**0.961 ± 0.012**
	G-M	0.969 ± 0.009	0.969 ± 0.013	0.972 ± 0.009	0.970 ± 0.009	0.962 ± 0.014	0.931 ± 0.064	0.973 ± 0.007	0.970 ± 0.006	**0.976 ± 0.007**
	AUC	0.995 ± 0.002	0.994 ± 0.003	0.995 ± 0.003	**0.996 ± 0.001**	0.993 ± 0.003	0.989 ± 0.013	**0.996 ± 0.002**	0.995 ± 0.002	**0.996 ± 0.002**
Ecoli	F_M	0.844 ± 0.055	0.860 ± 0.023	0.756 ± 0.069	**0.867 ± 0.033**	0.586 ± 0.158	0.671 ± 0.230	0.796 ± 0.081	0.851 ± 0.025	0.860 ± 0.032
	G-M	0.933 ± 0.039	0.940 ± 0.032	0.905 ± 0.035	0.938 ± 0.030	0.818 ± 0.114	0.835 ± 0.144	0.913 ± 0.030	0.934 ± 0.032	**0.940 ± 0.034**
	AUC	0.954 ± 0.036	0.958 ± 0.028	0.947 ± 0.028	0.960 ± 0.034	0.946 ± 0.038	0.929 ± 0.040	**0.961 ± 0.031**	0.950 ± 0.035	0.959 ± 0.031
Pima	F_M	0.593 ± 0.086	0.589 ± 0.080	0.595 ± 0.088	0.607 ± 0.065	0.652 ± 0.020	0.660 ± 0.037	0.649 ± 0.028	**0.669 ± 0.022**	0.658 ± 0.018
	G-M	0.682 ± 0.071	0.678 ± 0.066	0.681 ± 0.071	0.692 ± 0.053	0.726 ± 0.016	0.736 ± 0.031	0.726 ± 0.022	**0.743 ± 0.018**	0.734 ± 0.015
	AUC	0.769 ± 0.080	0.757 ± 0.069	0.754 ± 0.069	0.767 ± 0.063	0.809 ± 0.022	0.822 ± 0.036	0.811 ± 0.016	0.813 ± 0.021	**0.825 ± 0.022**
Balance	F_M	NaN	0.113 ± 0.066	0.129 ± 0.051	NaN	0.183 ± 0.060	0.213 ± 0.032	**0.250 ± 0.054**	0.221 ± 0.030	0.212 ± 0.036
	G-M	0.115 ± 0.126	0.358 ± 0.116	0.398 ± 0.086	0.079 ± 0.122	0.574 ± 0.113	0.634 ± 0.048	**0.654 ± 0.089**	0.596 ± 0.050	0.548 ± 0.086
	AUC	0.666 ± 0.012	0.703 ± 0.030	0.715 ± 0.022	0.679 ± 0.077	0.648 ± 0.096	0.695 ± 0.021	**0.767 ± 0.045**	0.727 ± 0.026	0.758 ± 0.026
Liver	F_M	0.623 ± 0.032	0.607 ± 0.055	0.628 ± 0.042	**0.637 ± 0.044**	0.590 ± 0.033	0.592 ± 0.030	0.596 ± 0.057	0.595 ± 0.053	0.604 ± 0.044
	G-M	0.668 ± 0.026	0.655 ± 0.041	0.670 ± 0.033	**0.683 ± 0.036**	0.565 ± 0.067	0.546 ± 0.051	0.640 ± 0.037	0.643 ± 0.040	0.659 ± 0.033
	AUC	0.726 ± 0.027	0.727 ± 0.034	0.734 ± 0.038	**0.735 ± 0.034**	0.672 ± 0.062	0.671 ± 0.049	0.697 ± 0.046	0.712 ± 0.047	0.719 ± 0.032
Wine	F_M	0.976 ± 0.020	0.976 ± 0.020	0.976 ± 0.020	0.976 ± 0.020	0.874 ± 0.113	0.974 ± 0.023	0.981 ± 0.017	**0.983 ± 0.021**	0.979 ± 0.018
	G-M	0.978 ± 0.019	0.978 ± 0.019	0.978 ± 0.019	0.978 ± 0.019	0.874 ± 0.137	0.978 ± 0.020	0.983 ± 0.015	**0.985 ± 0.020**	0.981 ± 0.017
	AUC	**0.999 ± 0.001**	**0.999 ± 0.001**	**0.999 ± 0.001**	**0.999 ± 0.001**	0.990 ± 0.011	**0.999 ± 0.002**	**0.999 ± 0.002**	**0.999 ± 0.001**	**0.999 ± 0.001**
Breast	F_M	0.634 ± 0.085	0.654 ± 0.085	0.677 ± 0.082	0.663 ± 0.060	**0.695 ± 0.074**	0.679 ± 0.063	0.664 ± 0.087	0.672 ± 0.087	0.685 ± 0.088
	G-M	0.704 ± 0.077	0.722 ± 0.078	0.741 ± 0.081	0.729 ± 0.057	**0.749 ± 0.085**	0.718 ± 0.082	0.734 ± 0.077	0.739 ± 0.086	0.748 ± 0.088
	AUC	0.815 ± 0.063	0.829 ± 0.052	0.834 ± 0.071	0.833 ± 0.054	0.806 ± 0.076	0.834 ± 0.077	**0.860 ± 0.059**	0.849 ± 0.065	**0.860 ± 0.066**
Libra	F_M	0.509 ± 0.085	0.625 ± 0.140	0.540 ± 0.098	0.610 ± 0.117	0.731 ± 0.142	**0.803 ± 0.099**	0.719 ± 0.101	0.670 ± 0.080	0.684 ± 0.098
	G-M	0.725 ± 0.072	0.756 ± 0.115	0.750 ± 0.082	0.750 ± 0.093	0.762 ± 0.115	**0.828 ± 0.092**	0.751 ± 0.082	0.711 ± 0.065	0.723 ± 0.079
	AUC	0.995 ± 0.007	0.994 ± 0.007	0.995 ± 0.007	0.995 ± 0.007	0.989 ± 0.013	0.988 ± 0.012	**0.996 ± 0.006**	**0.996 ± 0.006**	**0.996 ± 0.006**
LEV	F_M	0.477 ± 0.048	0.510 ± 0.043	0.471 ± 0.039	**0.568 ± 0.049**	0.358 ± 0.056	0.459 ± 0.103	0.415 ± 0.053	0.513 ± 0.045	0.553 ± 0.064
	G-M	0.747 ± 0.047	0.746 ± 0.049	0.746 ± 0.041	0.762 ± 0.048	0.750 ± 0.055	0.773 ± 0.066	0.792 ± 0.040	0.797 ± 0.057	**0.804 ± 0.076**
	AUC	0.748 ± 0.055	0.750 ± 0.053	0.755 ± 0.053	0.787 ± 0.062	0.864 ± 0.034	0.884 ± 0.041	0.869 ± 0.038	**0.889 ± 0.029**	0.870 ± 0.073
Iris	F_M	0.947 ± 0.035	**0.956 ± 0.025**	0.938 ± 0.042	**0.956 ± 0.025**	0.803 ± 0.141	0.828 ± 0.120	0.951 ± 0.031	0.947 ± 0.034	**0.956 ± 0.025**
	G-M	0.967 ± 0.024	**0.972 ± 0.018**	0.962 ± 0.028	**0.972 ± 0.018**	0.832 ± 0.136	0.868 ± 0.100	0.967 ± 0.022	0.970 ± 0.021	**0.972 ± 0.018**
	AUC	0.993 ± 0.004	**0.994 ± 0.004**	0.986 ± 0.014	0.993 ± 0.004	0.990 ± 0.012	0.983 ± 0.016	0.993 ± 0.005	0.993 ± 0.004	**0.994 ± 0.004**
Heart	F_M	0.797 ± 0.028	0.796 ± 0.034	0.794 ± 0.022	0.793 ± 0.026	0.790 ± 0.059	**0.812 ± 0.027**	0.794 ± 0.030	0.801 ± 0.037	0.810 ± 0.036
	G-M	0.817 ± 0.023	0.816 ± 0.029	0.813 ± 0.019	0.814 ± 0.022	0.776 ± 0.130	0.828 ± 0.021	0.814 ± 0.025	0.821 ± 0.031	**0.829 ± 0.030**
	AUC	0.865 ± 0.027	0.867 ± 0.027	0.858 ± 0.020	0.864 ± 0.026	0.864 ± 0.043	0.870 ± 0.031	0.860 ± 0.022	0.872 ± 0.028	**0.873 ± 0.027**
Glass	F_M	**0.755 ± 0.028**	0.740 ± 0.042	0.746 ± 0.047	0.745 ± 0.035	0.650 ± 0.040	0.662 ± 0.042	0.741 ± 0.031	0.750 ± 0.033	**0.755 ± 0.027**
	G-M	**0.828 ± 0.028**	0.813 ± 0.036	0.819 ± 0.041	0.818 ± 0.031	0.699 ± 0.050	0.709 ± 0.067	0.812 ± 0.032	0.821 ± 0.027	**0.828 ± 0.025**
	AUC	0.873 ± 0.020	0.873 ± 0.022	0.867 ± 0.030	**0.880 ± 0.023**	0.821 ± 0.063	0.856 ± 0.037	0.862 ± 0.030	0.861 ± 0.035	0.870 ± 0.028
Haber	F_M	0.435 ± 0.036	0.410 ± 0.042	**0.445 ± 0.067**	0.401 ± 0.035	0.442 ± 0.049	0.395 ± 0.059	0.389 ± 0.034	0.395 ± 0.069	0.412 ± 0.050
	G-M	0.591 ± 0.031	0.571 ± 0.037	**0.604 ± 0.059**	0.566 ± 0.031	0.593 ± 0.042	0.553 ± 0.045	0.552 ± 0.028	0.559 ± 0.062	0.571 ± 0.046
	AUC	0.651 ± 0.031	0.645 ± 0.040	0.649 ± 0.052	0.632 ± 0.027	0.628 ± 0.032	0.636 ± 0.049	0.621 ± 0.025	0.633 ± 0.042	**0.659 ± 0.018**
Eucal.	F_M	NaN	0.189 ± 0.118	0.412 ± 0.082	0.162 ± 0.097	0.327 ± 0.130	0.379 ± 0.124	0.410 ± 0.127	**0.421 ± 0.059**	0.417 ± 0.061
	G-M	0.097 ± 0.112	0.335 ± 0.145	0.567 ± 0.063	0.303 ± 0.116	0.594 ± 0.192	**0.648 ± 0.182**	0.615 ± 0.132	0.604 ± 0.077	0.617 ± 0.061
	AUC	0.707 ± 0.053	0.733 ± 0.036	0.778 ± 0.031	0.724 ± 0.045	0.746 ± 0.072	0.752 ± 0.073	0.753 ± 0.049	**0.797 ± 0.053**	0.785 ± 0.045
Heating	F_M	NaN	0.591 ± 0.021	0.572 ± 0.022	0.614 ± 0.027	0.741 ± 0.071	0.698 ± 0.031	0.746 ± 0.022	0.756 ± 0.039	**0.759 ± 0.052**
	G-M	0.431 ± 0.374	0.690 ± 0.007	0.674 ± 0.012	0.706 ± 0.025	0.844 ± 0.052	0.823 ± 0.013	0.846 ± 0.025	0.859 ± 0.025	**0.865 ± 0.034**
	AUC	0.844 ± 0.038	0.875 ± 0.027	0.853 ± 0.011	0.887 ± 0.015	0.891 ± 0.044	0.880 ± 0.037	0.919 ± 0.020	0.919 ± 0.014	**0.923 ± 0.021**
Seg.	F_M	0.874 ± 0.017	**0.886 ± 0.031**	0.852 ± 0.024	0.883 ± 0.033	0.650 ± 0.158	0.696 ± 0.062	0.855 ± 0.015	0.826 ± 0.056	0.868 ± 0.006
	G-M	0.956 ± 0.009	0.956 ± 0.014	0.940 ± 0.027	0.950 ± 0.012	0.884 ± 0.081	0.915 ± 0.021	**0.957 ± 0.011**	0.946 ± 0.011	0.948 ± 0.008
	AUC	0.986 ± 0.006	0.986 ± 0.006	0.986 ± 0.004	0.983 ± 0.002	0.969 ± 0.017	0.976 ± 0.006	**0.987 ± 0.004**	0.983 ± 0.008	0.984 ± 0.005

generate a new instance between a candidate instance and one of its neighbors from the same sub-cluster (Barua et al., 2014). As can be seen in Fig. 2(a) and (b), both approaches can lead to the generation of synthetic instances that overlap with the instances of the other class. In the first approach, some of the *NN*-nearest neighbors may be far from the candidate instance whereas in the second approach, sub-clusters from different classes may overlap. Overlapping synthetic instances can deteriorate the performance of the classifiers significantly (Barua et al., 2014; Beyan & Fisher, 2014).

Our proposed semi-unsupervised hierarchical clustering algorithm is designed to reduce the generation of overlapping synthetic instances. The algorithm is based on the Agglomerative Complete-Linkage Hierarchical Clustering (Voorhees, 1986) in which overlapping is checked in each iteration for the two minority sub-clusters that are nominated to be merged. If a majority sub-cluster exists between them, the algorithm will not merge the minority sub-clusters. Otherwise, the two nominated sub-clusters are merged if their distance is less than a pre-defined threshold. In contrast with the algorithm presented in (Voorhees, 1986), our hierarchical clustering method uses information about the majority instances to merge the nominated minority sub-clusters and avoid generating overlapping synthetic instances as shown in Fig. 2(c). Given that information about the majority instances is used in our clustering approach, the algorithm is not fully unsupervised as in conventional clustering approaches but semi-unsupervised.

Before clustering, noisy instances are identified for both classes using the method suggested by (Han et al., 2005) and removed from the dataset. For each instance, *NS*-nearest neighbors are found. If all the *NS*-nearest neighbors belong to the other class, then the instance is considered as noise and removed from the dataset because it indicates that it is surrounded by instances of the other class. The Semi-Unsupervised clustering algorithm starts by first clustering the majority class using hierarchical clustering, which results in *m* majority sub-clusters $Cmaj_{i=1,\ \ldots,\ m}$. For the minority class, a modification of the hierarchical clustering approach was used because hierarchical clustering enables the detection and avoidance of majority sub-clusters between the generated minority ones. The steps of our proposed semi-unsupervised hierarchical clustering algorithm are as follows:

Assume we have a dataset with *N* instances as input.

(1) Assign each minority instance to a separate sub-cluster. This will result in *N* sub-clusters of size one $B = \{Cmin_{\tau=1,\ldots,N}\}$.
(2) Identify the two sub-clusters say $Cmin_a$ and $Cmin_b$ with the lowest Euclidean distance between them. Let their distance be represented by π.

Table 3
Results for the sampling methods on the 16 datasets classified using KNN.

Dataset	Meas.	Random	SMOTE	Borderline SMOTE	Safe-level SMOTE	SBC	Cluster SMOTE	CBOS	MWMOTE	A-SUWO
Vehicle	F_M	0.840 ± 0.025	0.835 ± 0.018	0.864 ± 0.027	0.840 ± 0.022	0.743 ± 0.102	0.833 ± 0.022	0.869 ± 0.032	0.842 ± 0.023	**0.903 ± 0.020**
	G-M	0.930 ± 0.014	0.928 ± 0.009	0.940 ± 0.018	0.929 ± 0.010	0.854 ± 0.074	0.926 ± 0.011	0.942 ± 0.014	0.930 ± 0.012	**0.954 ± 0.010**
	AUC	0.981 ± 0.006	0.984 ± 0.007	0.982 ± 0.004	**0.985 ± 0.005**	0.962 ± 0.014	0.985 ± 0.004	0.983 ± 0.004	0.982 ± 0.005	0.983 ± 0.008
Ecoli	F_M	0.735 ± 0.042	0.731 ± 0.072	0.692 ± 0.055	0.828 ± 0.045	0.601 ± 0.141	0.766 ± 0.025	0.780 ± 0.068	0.795 ± 0.057	**0.840 ± 0.049**
	G-M	0.906 ± 0.027	0.902 ± 0.031	0.890 ± 0.029	0.932 ± 0.035	0.822 ± 0.107	0.915 ± 0.026	0.913 ± 0.029	0.924 ± 0.035	**0.935 ± 0.034**
	AUC	0.939 ± 0.035	0.939 ± 0.035	0.924 ± 0.026	**0.947 ± 0.040**	0.932 ± 0.033	0.944 ± 0.037	0.935 ± 0.034	0.944 ± 0.039	0.945 ± 0.037
Pima	F_M	0.617 ± 0.029	0.616 ± 0.044	0.609 ± 0.026	0.627 ± 0.034	0.590 ± 0.050	**0.639 ± 0.022**	0.579 ± 0.034	0.618 ± 0.035	0.612 ± 0.023
	G-M	0.690 ± 0.023	0.687 ± 0.039	0.673 ± 0.024	0.701 ± 0.028	0.663 ± 0.036	**0.707 ± 0.020**	0.663 ± 0.029	0.687 ± 0.031	0.683 ± 0.020
	AUC	0.742 ± 0.042	0.741 ± 0.045	0.733 ± 0.036	0.754 ± 0.036	0.736 ± 0.032	**0.768 ± 0.026**	0.715 ± 0.037	0.747 ± 0.033	0.741 ± 0.026
Balance	F_M	0.000 ± 0.000	0.094 ± 0.038	0.093 ± 0.052	0.000 ± 0.000	0.208 ± 0.021	**0.243 ± 0.045**	0.124 ± 0.051	0.134 ± 0.061	0.126 ± 0.075
	G-M	0.076 ± 0.118	0.362 ± 0.085	0.357 ± 0.107	0.170 ± 0.137	0.612 ± 0.041	**0.649 ± 0.081**	0.417 ± 0.109	0.433 ± 0.125	0.417 ± 0.146
	AUC	0.445 ± 0.020	0.525 ± 0.042	0.528 ± 0.048	0.454 ± 0.040	0.729 ± 0.039	**0.717 ± 0.064**	0.564 ± 0.049	0.557 ± 0.064	0.592 ± 0.056
Liver	F_M	0.592 ± 0.029	0.576 ± 0.039	**0.596 ± 0.031**	0.574 ± 0.049	0.557 ± 0.054	0.581 ± 0.037	0.587 ± 0.031	0.584 ± 0.036	0.553 ± 0.064
	G-M	0.567 ± 0.025	0.551 ± 0.044	0.569 ± 0.019	0.554 ± 0.054	0.481 ± 0.049	0.550 ± 0.040	**0.577 ± 0.039**	0.562 ± 0.021	0.567 ± 0.033
	AUC	0.608 ± 0.044	0.602 ± 0.035	**0.629 ± 0.044**	0.607 ± 0.054	0.575 ± 0.058	0.611 ± 0.037	0.612 ± 0.036	0.616 ± 0.048	0.603 ± 0.030
Wine	F_M	0.950 ± 0.030	0.953 ± 0.027	0.958 ± 0.031	0.964 ± 0.028	0.910 ± 0.118	0.957 ± 0.024	0.960 ± 0.022	0.964 ± 0.030	**0.969 ± 0.031**
	G-M	0.956 ± 0.026	0.960 ± 0.023	0.965 ± 0.026	0.968 ± 0.025	0.908 ± 0.132	0.964 ± 0.022	0.964 ± 0.020	0.968 ± 0.027	**0.971 ± 0.029**
	AUC	0.990 ± 0.012	0.990 ± 0.012	**0.992 ± 0.012**	0.991 ± 0.012	0.979 ± 0.035	0.990 ± 0.013	0.981 ± 0.020	0.991 ± 0.012	0.991 ± 0.013
Breast	F_M	0.682 ± 0.076	0.697 ± 0.080	0.700 ± 0.083	0.706 ± 0.066	0.689 ± 0.075	**0.738 ± 0.095**	0.698 ± 0.075	0.700 ± 0.067	0.710 ± 0.064
	G-M	0.752 ± 0.065	0.763 ± 0.075	0.763 ± 0.080	0.771 ± 0.064	0.734 ± 0.093	**0.795 ± 0.090**	0.760 ± 0.072	0.766 ± 0.069	0.779 ± 0.053
	AUC	0.845 ± 0.035	0.849 ± 0.039	0.851 ± 0.046	0.846 ± 0.039	0.825 ± 0.052	**0.859 ± 0.054**	0.844 ± 0.056	0.856 ± 0.038	0.845 ± 0.045
Libra	F_M	0.974 ± 0.018	0.979 ± 0.021	0.966 ± 0.025	0.974 ± 0.026	0.811 ± 0.140	0.949 ± 0.039	0.979 ± 0.018	0.977 ± 0.019	**0.981 ± 0.020**
	G-M	0.983 ± 0.014	0.984 ± 0.015	0.979 ± 0.015	0.978 ± 0.024	0.918 ± 0.073	0.978 ± 0.022	0.984 ± 0.015	0.984 ± 0.015	**0.985 ± 0.016**
	AUC	0.986 ± 0.015	0.988 ± 0.015	0.988 ± 0.015	0.986 ± 0.015	0.986 ± 0.014	**0.992 ± 0.012**	0.986 ± 0.015	0.986 ± 0.015	0.988 ± 0.015
LEV	F_M	0.446 ± 0.044	0.451 ± 0.033	0.436 ± 0.038	0.568 ± 0.056	0.324 ± 0.036	0.474 ± 0.072	0.465 ± 0.042	0.477 ± 0.042	**0.581 ± 0.066**
	G-M	0.760 ± 0.044	0.755 ± 0.034	0.759 ± 0.038	0.768 ± 0.038	0.735 ± 0.046	0.755 ± 0.064	0.741 ± 0.036	**0.771 ± 0.043**	0.759 ± 0.038
	AUC	0.791 ± 0.046	0.799 ± 0.042	0.787 ± 0.039	0.810 ± 0.050	**0.833 ± 0.049**	0.814 ± 0.056	0.795 ± 0.038	0.801 ± 0.046	0.807 ± 0.043
Iris	F_M	0.937 ± 0.035	0.942 ± 0.040	0.916 ± 0.041	0.937 ± 0.043	0.826 ± 0.114	0.937 ± 0.039	0.929 ± 0.047	0.933 ± 0.034	**0.942 ± 0.025**
	G-M	0.959 ± 0.025	0.962 ± 0.026	0.946 ± 0.030	0.957 ± 0.031	0.873 ± 0.099	0.959 ± 0.029	0.954 ± 0.033	0.957 ± 0.024	**0.965 ± 0.016**
	AUC	0.975 ± 0.023	**0.980 ± 0.019**	0.973 ± 0.026	0.975 ± 0.022	0.966 ± 0.024	0.979 ± 0.020	0.979 ± 0.019	0.972 ± 0.019	0.977 ± 0.020
Heart	F_M	0.825 ± 0.024	0.826 ± 0.021	0.809 ± 0.014	0.824 ± 0.024	0.823 ± 0.017	0.823 ± 0.027	0.787 ± 0.026	0.820 ± 0.024	**0.832 ± 0.021**
	G-M	0.836 ± 0.021	0.838 ± 0.019	0.812 ± 0.019	0.836 ± 0.021	0.833 ± 0.016	0.835 ± 0.023	0.801 ± 0.025	0.832 ± 0.022	**0.845 ± 0.018**
	AUC	0.893 ± 0.019	0.890 ± 0.020	0.876 ± 0.015	0.892 ± 0.020	0.888 ± 0.024	**0.894 ± 0.019**	0.861 ± 0.021	0.891 ± 0.020	0.886 ± 0.027
Glass	F_M	0.688 ± 0.027	0.707 ± 0.041	0.696 ± 0.029	0.696 ± 0.034	0.629 ± 0.038	0.702 ± 0.031	0.709 ± 0.034	0.700 ± 0.034	**0.720 ± 0.023**
	G-M	0.766 ± 0.024	0.783 ± 0.037	0.773 ± 0.024	0.773 ± 0.029	0.669 ± 0.062	0.778 ± 0.027	0.786 ± 0.030	0.779 ± 0.029	**0.796 ± 0.022**
	AUC	0.839 ± 0.039	0.855 ± 0.036	0.841 ± 0.048	0.850 ± 0.030	0.819 ± 0.043	0.853 ± 0.026	**0.859 ± 0.034**	0.846 ± 0.033	0.856 ± 0.030
Haber	F_M	0.401 ± 0.043	0.393 ± 0.068	0.403 ± 0.071	0.367 ± 0.057	0.391 ± 0.052	**0.447 ± 0.047**	0.383 ± 0.052	0.401 ± 0.087	0.392 ± 0.048
	G-M	0.560 ± 0.038	0.552 ± 0.062	0.559 ± 0.068	0.535 ± 0.051	0.520 ± 0.036	**0.593 ± 0.046**	0.548 ± 0.045	0.560 ± 0.078	0.558 ± 0.042
	AUC	0.574 ± 0.035	0.566 ± 0.054	0.569 ± 0.051	0.571 ± 0.039	0.562 ± 0.062	**0.609 ± 0.043**	0.578 ± 0.040	0.576 ± 0.062	0.576 ± 0.044
Eucal.	F_M	**0.395 ± 0.027**	0.343 ± 0.025	0.342 ± 0.018	0.354 ± 0.024	0.340 ± 0.020	0.388 ± 0.061	0.361 ± 0.021	0.329 ± 0.031	0.368 ± 0.020
	G-M	0.679 ± 0.028	0.647 ± 0.032	0.644 ± 0.021	0.650 ± 0.029	0.620 ± 0.036	**0.687 ± 0.064**	0.666 ± 0.020	0.632 ± 0.035	0.674 ± 0.020
	AUC	0.724 ± 0.049	0.730 ± 0.026	0.719 ± 0.028	0.728 ± 0.018	0.730 ± 0.041	**0.734 ± 0.069**	0.708 ± 0.032	0.701 ± 0.040	0.720 ± 0.025
Heating	F_M	0.697 ± 0.022	0.711 ± 0.030	0.707 ± 0.044	0.724 ± 0.019	0.697 ± 0.014	0.716 ± 0.021	0.665 ± 0.098	0.704 ± 0.014	**0.752 ± 0.048**
	G-M	0.821 ± 0.011	0.835 ± 0.021	0.834 ± 0.035	0.845 ± 0.015	0.827 ± 0.011	0.841 ± 0.013	0.781 ± 0.094	0.829 ± 0.006	**0.860 ± 0.027**
	AUC	0.874 ± 0.014	0.875 ± 0.016	0.866 ± 0.023	0.886 ± 0.019	0.896 ± 0.024	**0.897 ± 0.013**	0.865 ± 0.033	0.885 ± 0.021	0.886 ± 0.016
Seg.	F_M	0.832 ± 0.028	0.827 ± 0.033	0.829 ± 0.024	0.848 ± 0.027	0.593 ± 0.048	0.833 ± 0.041	0.824 ± 0.027	0.838 ± 0.031	**0.855 ± 0.025**
	G-M	0.948 ± 0.012	0.946 ± 0.013	0.945 ± 0.015	0.950 ± 0.009	0.876 ± 0.027	**0.953 ± 0.013**	0.947 ± 0.010	0.950 ± 0.011	0.947 ± 0.013
	AUC	0.967 ± 0.012	0.973 ± 0.012	0.972 ± 0.010	0.971 ± 0.012	0.954 ± 0.017	**0.978 ± 0.009**	0.969 ± 0.009	0.969 ± 0.011	0.964 ± 0.006

(3) Find majority sub-clusters, say $Cmaj_{i \in A}$ with the Euclidean distance to $Cmin_a$ and $Cmin_b$ less than π. A is the set of majority class indices with such property.

(4) If $A \neq \emptyset$, then, there exists a majority sub-cluster between $Cmin_a$ and $Cmin_b$ and hence they should not be merged. The distance between $Cmin_a$ and $Cmin_b$ will be set to a large number to avoid being considered for merging again.

(5) Else, $Cmin_a$ and $Cmin_b$ are merged into one sub-cluster $Cmin_c$. This will result in one less member in B.

(6) Finally, the Euclidean distance between the newly formed $Cmin_c$ and existing sub-clusters is recalculated. Steps 2 to 6 are repeated until the Euclidean distance between the closest sub-clusters is less than a threshold T. This will result in n minority sub-clusters.

In contrast with the clustering algorithm from (Voorhees, 1986), our proposed semi-unsupervised hierarchical clustering algorithm checks whether the two sub-clusters $Cmin_a$ and $Cmin_b$ contain part of a majority sub-cluster (steps 3 through 5). If so, they will not be merged.

In order to obtain a better estimate of T for both minority and majority classes, the median Euclidean distance $d_{med,\,h}$ of each minority (majority) instance h to all other minority (majority) instances is determined. The median Euclidean distance is used rather than the average distance because the former is more robust to noisy minority instances. Then, we define d_{avg} as the average $d_{med,\,h}$ over all minority (majority) instances. Therefore, T can be estimated as follows:

$$T = d_{avg} * c_{thres} \tag{1}$$

where c_{thres} is a user-defined constant parameter and its optimum value depends on the dataset. Further suggestion regarding the selection of reasonable values for this parameter can be found in the "Results and Discussion" section

3.2. Adaptive sub-cluster sizing

In current cluster-based oversampling techniques, all sub-clusters have similar sizes after oversampling. However, there might be some sub-clusters with higher misclassification error rate that need more oversampling. Similarly, there might be some with lower misclassification error rate that do not need much oversampling. In the proposed A-SUWO method, the size of the sub-cluster depends on the misclassification rate of its instances. The misclassification error for each sub-cluster is calculated using cross validation. This has two main goals. The first goal is to balance the dataset with a 1:1 ratio so that both classes are of the same size. The second goal is to assign a larger size to sub-clusters with higher misclassification error to provide more importance to the ones whose instances are harder to classify.

Table 4
Results for the sampling methods on the 16 datasets classified using logistic regression.

Dataset	Meas.	Random	SMOTE	Borderline SMOTE	Safe-level SMOTE	SBC	Cluster SMOTE	CBOS	MWMOTE	A-SUWO
Vehicle	F_M	0.932 ± 0.019	**0.934 ± 0.020**	0.923 ± 0.028	0.932 ± 0.019	0.903 ± 0.025	0.931 ± 0.021	0.930 ± 0.021	0.934 ± 0.020	0.921 ± 0.028
	G-M	**0.963 ± 0.012**	0.961 ± 0.014	0.951 ± 0.025	0.960 ± 0.016	0.950 ± 0.009	0.960 ± 0.016	0.958 ± 0.016	0.961 ± 0.014	0.951 ± 0.024
	AUC	0.991 ± 0.006	0.991 ± 0.005	0.989 ± 0.007	0.991 ± 0.006	0.987 ± 0.004	0.991 ± 0.005	0.991 ± 0.006	0.990 ± 0.005	**0.991 ± 0.006**
Ecoli	F_M	0.703 ± 0.039	0.696 ± 0.034	0.601 ± 0.046	0.700 ± 0.020	0.546 ± 0.156	0.692 ± 0.035	0.685 ± 0.085	0.698 ± 0.051	**0.716 ± 0.033**
	G-M	**0.878 ± 0.021**	0.871 ± 0.029	0.839 ± 0.015	0.857 ± 0.039	0.780 ± 0.118	0.863 ± 0.024	0.860 ± 0.031	0.873 ± 0.030	0.863 ± 0.015
	AUC	**0.933 ± 0.026**	0.933 ± 0.027	0.913 ± 0.023	0.933 ± 0.026	0.875 ± 0.088	0.927 ± 0.025	0.924 ± 0.025	0.931 ± 0.026	0.933 ± 0.027
Pima	F_M	0.593 ± 0.086	0.589 ± 0.080	0.595 ± 0.088	0.607 ± 0.065	0.652 ± 0.020	0.660 ± 0.037	0.649 ± 0.028	0.669 ± 0.022	**0.658 ± 0.018**
	G-M	0.682 ± 0.071	0.678 ± 0.066	0.681 ± 0.071	0.692 ± 0.053	0.726 ± 0.016	0.736 ± 0.031	0.726 ± 0.022	0.743 ± 0.018	**0.734 ± 0.015**
	AUC	0.769 ± 0.080	0.757 ± 0.069	0.754 ± 0.069	0.767 ± 0.063	0.809 ± 0.022	0.822 ± 0.036	0.811 ± 0.016	0.813 ± 0.021	**0.825 ± 0.022**
Balance	F_M	0.110 ± 0.032	0.111 ± 0.025	0.116 ± 0.030	0.115 ± 0.060	0.089 ± 0.028	0.133 ± 0.039	0.102 ± 0.023	0.111 ± 0.019	**0.149 ± 0.027**
	G-M	0.442 ± 0.068	0.447 ± 0.053	0.456 ± 0.061	0.425 ± 0.093	0.393 ± 0.067	0.489 ± 0.083	0.427 ± 0.052	0.448 ± 0.036	**0.517 ± 0.048**
	AUC	0.419 ± 0.042	0.432 ± 0.033	0.428 ± 0.037	0.449 ± 0.089	0.448 ± 0.052	0.483 ± 0.070	0.436 ± 0.050	0.417 ± 0.034	**0.530 ± 0.065**
Liver	F_M	0.606 ± 0.052	0.627 ± 0.043	0.611 ± 0.068	0.625 ± 0.037	0.596 ± 0.015	0.617 ± 0.033	0.579 ± 0.064	0.628 ± 0.040	**0.637 ± 0.044**
	G-M	0.641 ± 0.048	0.658 ± 0.034	0.646 ± 0.054	0.661 ± 0.029	0.570 ± 0.063	0.651 ± 0.030	0.600 ± 0.044	0.662 ± 0.031	**0.676 ± 0.034**
	AUC	0.714 ± 0.031	0.718 ± 0.029	0.715 ± 0.033	**0.720 ± 0.024**	0.694 ± 0.026	0.718 ± 0.025	0.690 ± 0.045	0.720 ± 0.023	0.720 ± 0.032
Wine	F_M	0.947 ± 0.034	0.945 ± 0.030	0.945 ± 0.030	0.942 ± 0.030	0.942 ± 0.025	0.945 ± 0.030	0.947 ± 0.034	0.945 ± 0.030	**0.952 ± 0.036**
	G-M	0.954 ± 0.032	0.952 ± 0.030	0.952 ± 0.030	0.950 ± 0.029	0.954 ± 0.022	0.952 ± 0.030	0.954 ± 0.032	0.952 ± 0.030	**0.959 ± 0.031**
	AUC	0.995 ± 0.005	0.994 ± 0.007	0.994 ± 0.008	0.995 ± 0.006	0.992 ± 0.010	0.995 ± 0.006	0.995 ± 0.005	0.994 ± 0.008	**0.996 ± 0.004**
Breast	F_M	0.724 ± 0.085	0.733 ± 0.096	0.696 ± 0.078	0.738 ± 0.093	0.697 ± 0.064	0.759 ± 0.066	0.739 ± 0.093	0.725 ± 0.074	**0.764 ± 0.091**
	G-M	0.792 ± 0.070	0.796 ± 0.076	0.770 ± 0.065	0.803 ± 0.075	0.763 ± 0.062	0.821 ± 0.054	0.806 ± 0.080	0.794 ± 0.062	**0.824 ± 0.073**
	AUC	0.880 ± 0.079	0.882 ± 0.080	**0.896 ± 0.061**	0.883 ± 0.080	0.854 ± 0.073	0.880 ± 0.083	0.892 ± 0.072	0.885 ± 0.069	0.890 ± 0.073
Libra	F_M	0.485 ± 0.109	0.504 ± 0.101	0.505 ± 0.092	0.503 ± 0.098	0.320 ± 0.077	0.498 ± 0.115	0.529 ± 0.128	0.508 ± 0.112	**0.541 ± 0.099**
	G-M	0.647 ± 0.090	0.662 ± 0.088	0.659 ± 0.084	0.658 ± 0.082	0.536 ± 0.079	0.656 ± 0.104	0.678 ± 0.114	0.665 ± 0.097	**0.683 ± 0.088**
	AUC	0.696 ± 0.091	**0.710 ± 0.106**	0.705 ± 0.099	0.707 ± 0.101	0.549 ± 0.095	0.708 ± 0.101	0.708 ± 0.104	0.703 ± 0.102	0.707 ± 0.096
LEV	F_M	0.445 ± 0.024	0.469 ± 0.030	0.387 ± 0.025	0.565 ± 0.058	0.428 ± 0.033	0.448 ± 0.045	0.428 ± 0.055	0.512 ± 0.032	**0.586 ± 0.082**
	G-M	0.813 ± 0.032	0.822 ± 0.033	0.795 ± 0.027	0.815 ± 0.063	0.810 ± 0.026	0.811 ± 0.042	0.815 ± 0.037	**0.824 ± 0.042**	0.813 ± 0.071
	AUC	0.893 ± 0.034	0.894 ± 0.034	0.888 ± 0.034	0.896 ± 0.034	0.892 ± 0.037	0.893 ± 0.034	0.883 ± 0.050	0.893 ± 0.034	**0.897 ± 0.039**
Iris	F_M	0.936 ± 0.029	0.931 ± 0.031	**0.941 ± 0.018**	**0.941 ± 0.018**	0.893 ± 0.110	0.941 ± 0.019	0.931 ± 0.031	0.936 ± 0.029	**0.941 ± 0.018**
	G-M	0.955 ± 0.023	0.950 ± 0.026	**0.960 ± 0.012**	**0.960 ± 0.012**	0.918 ± 0.101	0.957 ± 0.017	0.950 ± 0.026	0.955 ± 0.023	**0.960 ± 0.012**
	AUC	0.991 ± 0.006	0.992 ± 0.006	0.991 ± 0.006	0.992 ± 0.006	0.970 ± 0.057	**0.993 ± 0.005**	0.992 ± 0.006	0.992 ± 0.006	**0.992 ± 0.005**
Heart	F_M	**0.853 ± 0.027**	0.853 ± 0.028	0.852 ± 0.027	0.849 ± 0.029	0.845 ± 0.028	0.852 ± 0.028	0.829 ± 0.025	0.851 ± 0.027	0.853 ± 0.028
	G-M	0.866 ± 0.025	0.865 ± 0.026	0.864 ± 0.026	0.862 ± 0.027	0.858 ± 0.026	0.865 ± 0.026	0.844 ± 0.023	0.864 ± 0.025	**0.866 ± 0.026**
	AUC	0.930 ± 0.016	**0.930 ± 0.016**	0.923 ± 0.015	0.930 ± 0.015	0.923 ± 0.015	0.930 ± 0.015	0.915 ± 0.015	0.929 ± 0.015	0.929 ± 0.017
Glass	F_M	0.649 ± 0.038	0.637 ± 0.040	0.635 ± 0.050	0.629 ± 0.062	0.642 ± 0.034	0.640 ± 0.058	**0.670 ± 0.058**	0.641 ± 0.046	0.663 ± 0.040
	G-M	0.735 ± 0.030	0.725 ± 0.033	0.721 ± 0.044	0.718 ± 0.049	0.699 ± 0.049	0.727 ± 0.049	**0.753 ± 0.049**	0.728 ± 0.037	0.749 ± 0.035
	AUC	0.827 ± 0.037	**0.830 ± 0.035**	0.820 ± 0.038	0.827 ± 0.034	0.803 ± 0.044	0.824 ± 0.036	0.822 ± 0.037	0.825 ± 0.034	0.818 ± 0.033
Haber	F_M	0.477 ± 0.049	0.465 ± 0.032	0.467 ± 0.041	0.462 ± 0.022	0.458 ± 0.075	0.459 ± 0.067	0.469 ± 0.080	0.453 ± 0.056	**0.508 ± 0.074**
	G-M	0.626 ± 0.041	0.617 ± 0.022	0.622 ± 0.034	0.612 ± 0.022	0.601 ± 0.074	0.605 ± 0.052	0.614 ± 0.079	0.606 ± 0.046	**0.649 ± 0.059**
	AUC	0.673 ± 0.049	0.648 ± 0.029	0.654 ± 0.039	0.653 ± 0.036	0.629 ± 0.082	0.645 ± 0.062	0.634 ± 0.104	0.638 ± 0.038	**0.695 ± 0.074**
Eucal.	F_M	0.499 ± 0.037	0.498 ± 0.041	0.512 ± 0.048	0.496 ± 0.015	0.366 ± 0.024	0.502 ± 0.077	**0.515 ± 0.023**	0.498 ± 0.044	0.511 ± 0.029
	G-M	0.731 ± 0.038	0.730 ± 0.039	0.746 ± 0.046	0.720 ± 0.014	0.672 ± 0.036	0.727 ± 0.071	**0.747 ± 0.018**	0.724 ± 0.034	0.728 ± 0.013
	AUC	0.846 ± 0.016	0.845 ± 0.017	0.843 ± 0.015	0.846 ± 0.018	0.738 ± 0.016	0.842 ± 0.012	0.845 ± 0.023	**0.848 ± 0.017**	0.834 ± 0.029
Heating	F_M	0.720 ± 0.059	0.726 ± 0.041	0.720 ± 0.053	0.723 ± 0.058	0.726 ± 0.055	**0.732 ± 0.042**	0.720 ± 0.059	0.730 ± 0.048	0.728 ± 0.059
	G-M	0.839 ± 0.045	0.842 ± 0.029	**0.845 ± 0.043**	0.837 ± 0.044	0.840 ± 0.042	0.844 ± 0.029	0.835 ± 0.043	0.845 ± 0.033	0.841 ± 0.047
	AUC	0.916 ± 0.027	0.919 ± 0.028	0.907 ± 0.030	0.918 ± 0.028	0.915 ± 0.028	0.919 ± 0.026	0.915 ± 0.030	0.917 ± 0.028	**0.921 ± 0.026**
Seg.	F_M	0.641 ± 0.039	0.647 ± 0.030	0.602 ± 0.026	0.657 ± 0.025	0.591 ± 0.075	**0.667 ± 0.008**	0.630 ± 0.023	0.657 ± 0.034	0.661 ± 0.018
	G-M	0.878 ± 0.012	0.877 ± 0.017	0.854 ± 0.017	0.879 ± 0.016	0.867 ± 0.042	0.872 ± 0.002	**0.881 ± 0.019**	0.875 ± 0.004	0.879 ± 0.011
	AUC	0.942 ± 0.008	0.942 ± 0.008	0.908 ± 0.021	0.942 ± 0.008	0.901 ± 0.051	**0.944 ± 0.010**	0.937 ± 0.006	0.944 ± 0.010	0.942 ± 0.008

As shown in Fig. 3, our method first randomly splits each of the n minority sub-clusters into K similar size partitions ($K = 3$ in the figure). Then, the classification method (Linear Discriminant Analysis) runs K times and in each fold, $K - 1$ partitions from each minority sub-cluster and all majority instances (in gray background) are used as the training set. Linear Discriminant Analysis was used as our classifier because it is simple and does not require any parameters to tune. Moreover, it was selected over other methods because the purpose is not to get high measures, but rather an estimate of the complexity of the sub-clusters. The remaining one partition from each minority sub-cluster (in white background) is used as the testing set. The misclassification error $\varepsilon_{j\kappa}$ for each minority sub-cluster j in fold κ is determined as the number of minority instances in the testing set incorrectly classified as majority. The error rate $\varepsilon^*_{j\kappa}$ is obtained by dividing $\varepsilon_{j\kappa}$ by the number of instances in each sub-cluster R_j. The average error rate $\bar{\varepsilon}^*_j$ is then obtained by averaging the error rate over all folds.

The next step is to standardize $\bar{\varepsilon}^*_j$ to obtain standardized average error rate $\hat{\varepsilon}^*_j$ using the following equation.

$$\hat{\varepsilon}^*_j = \frac{\bar{\varepsilon}^*_j}{\sum_{j=1}^{n} \bar{\varepsilon}^*_j} \tag{2}$$

Following our second goal, the final sizes of any two minority sub-clusters, say $L1$ and $L2$ should have similar ratio to their average error rates $\hat{\varepsilon}^*_{L1}$ and $\hat{\varepsilon}^*_{L2}$. That means,

$$\frac{S_{L1}}{S_{L2}} = \frac{\hat{\varepsilon}^*_{L1}}{\hat{\varepsilon}^*_{L2}} \;\forall\, L1, L2 \in \{1, \ldots, n\} \tag{3}$$

where S_{L1} and S_{L2} are the final sizes of $L1$ and $L2$ after oversampling, respectively. $\hat{\varepsilon}^*_{L1}$ and $\hat{\varepsilon}^*_{L2}$ are the standardized average error rate for $L1$ and $L2$, respectively.

The proposed method does not undersample any sub-cluster even if the size calculated using cross-validation is less than its initial size to avoid losing any information. After determining the required number of instances for each minority sub-cluster ($S_{j=1,\ldots,n}$), they should be oversampled to have the corresponding sizes.

3.3. Synthetic instance generation

In A-SUWO, we propose to generate synthetic instances between the original instances and their NN-nearest neighbors provided that they belong to the same sub-cluster (Fig. 2(c)). This is to avoid selecting a NN-nearest neighbor that is far from the instance and that belongs to another sub-cluster thus reducing the generation of overlapping synthetic instances. At the same time, A-SUWO assigns weights to the instances of all sub-clusters separately, which will guarantee that all sub-clusters are oversampled and no isolated small ones are ignored. This is in contrast with the work in (Barua et al., 2014), where there might be some sub-clusters that are not oversampled at all. It is

Table 5
Results for the Sampling methods on the 16 datasets classified using LDA.

Dataset	Meas.	Random	SMOTE	Borderline SMOTE	Safe-level SMOTE	SBC	Cluster SMOTE	CBOS	MWMOTE	A-SUWO
Vehicle	F_M	0.923 ± 0.014	0.923 ± 0.020	0.921 ± 0.025	0.925 ± 0.016	0.909 ± 0.020	0.926 ± 0.014	0.932 ± 0.012	0.917 ± 0.019	**0.935 ± 0.013**
	G-M	0.963 ± 0.005	0.963 ± 0.007	0.950 ± 0.021	0.963 ± 0.005	0.959 ± 0.010	0.963 ± 0.006	**0.965 ± 0.007**	0.961 ± 0.009	0.964 ± 0.010
	AUC	0.990 ± 0.004	0.991 ± 0.004	0.990 ± 0.007	0.990 ± 0.004	0.986 ± 0.011	**0.991 ± 0.004**	0.989 ± 0.006	0.991 ± 0.004	0.990 ± 0.005
Ecoli	F_M	0.729 ± 0.029	0.724 ± 0.046	0.618 ± 0.039	**0.743 ± 0.038**	0.583 ± 0.142	0.718 ± 0.034	0.713 ± 0.089	0.722 ± 0.038	0.735 ± 0.021
	G-M	0.907 ± 0.017	0.901 ± 0.008	0.847 ± 0.014	**0.911 ± 0.017**	0.818 ± 0.116	0.890 ± 0.018	0.899 ± 0.041	0.901 ± 0.010	0.901 ± 0.018
	AUC	0.937 ± 0.027	0.936 ± 0.027	0.919 ± 0.017	0.937 ± 0.028	0.920 ± 0.059	0.939 ± 0.027	0.938 ± 0.027	0.938 ± 0.026	**0.939 ± 0.028**
Pima	F_M	0.663 ± 0.039	0.669 ± 0.038	**0.675 ± 0.027**	0.670 ± 0.038	0.666 ± 0.034	0.661 ± 0.028	0.652 ± 0.024	0.666 ± 0.025	0.671 ± 0.038
	G-M	0.739 ± 0.033	0.743 ± 0.031	**0.747 ± 0.023**	0.744 ± 0.031	0.740 ± 0.028	0.737 ± 0.023	0.728 ± 0.020	0.741 ± 0.021	0.745 ± 0.032
	AUC	0.828 ± 0.030	0.829 ± 0.029	0.828 ± 0.031	**0.831 ± 0.027**	0.825 ± 0.028	0.830 ± 0.025	0.814 ± 0.023	0.828 ± 0.026	0.826 ± 0.028
Balance	F_M	0.110 ± 0.031	0.111 ± 0.025	0.116 ± 0.030	0.119 ± 0.066	0.115 ± 0.017	0.126 ± 0.028	0.107 ± 0.035	0.113 ± 0.021	**0.149 ± 0.029**
	G-M	0.442 ± 0.067	0.447 ± 0.053	0.456 ± 0.061	0.435 ± 0.105	0.454 ± 0.030	0.476 ± 0.059	0.437 ± 0.079	0.451 ± 0.045	**0.517 ± 0.049**
	AUC	0.419 ± 0.042	0.432 ± 0.033	0.428 ± 0.037	0.449 ± 0.089	0.451 ± 0.032	0.472 ± 0.066	0.454 ± 0.074	0.429 ± 0.043	**0.533 ± 0.051**
Liver	F_M	0.604 ± 0.051	0.601 ± 0.063	0.600 ± 0.064	0.610 ± 0.057	0.599 ± 0.013	0.602 ± 0.062	0.592 ± 0.029	0.603 ± 0.057	**0.613 ± 0.063**
	G-M	0.636 ± 0.048	0.632 ± 0.051	0.631 ± 0.054	0.640 ± 0.048	0.555 ± 0.057	0.632 ± 0.051	0.621 ± 0.033	0.633 ± 0.050	**0.654 ± 0.048**
	AUC	0.708 ± 0.040	0.711 ± 0.040	0.710 ± 0.039	**0.713 ± 0.039**	0.676 ± 0.031	0.710 ± 0.040	0.678 ± 0.038	0.710 ± 0.037	0.708 ± 0.047
Wine	F_M	0.965 ± 0.032	0.959 ± 0.029	0.976 ± 0.021	**0.961 ± 0.031**	0.929 ± 0.068	0.973 ± 0.018	0.966 ± 0.019	0.966 ± 0.022	0.968 ± 0.019
	G-M	0.967 ± 0.031	0.964 ± 0.026	0.977 ± 0.019	**0.964 ± 0.030**	0.936 ± 0.069	0.974 ± 0.018	0.970 ± 0.017	0.970 ± 0.021	0.970 ± 0.018
	AUC	0.999 ± 0.001	**0.999 ± 0.002**	0.999 ± 0.002	0.998 ± 0.002	0.990 ± 0.012	0.999 ± 0.001	0.999 ± 0.002	0.999 ± 0.001	0.999 ± 0.001
Breast	F_M	0.707 ± 0.066	0.696 ± 0.060	0.698 ± 0.083	0.706 ± 0.076	0.677 ± 0.057	0.704 ± 0.080	0.703 ± 0.078	0.719 ± 0.080	**0.719 ± 0.093**
	G-M	0.762 ± 0.087	0.754 ± 0.078	0.752 ± 0.094	0.765 ± 0.091	0.720 ± 0.079	0.760 ± 0.092	0.763 ± 0.091	0.769 ± 0.097	**0.773 ± 0.102**
	AUC	**0.899 ± 0.031**	0.897 ± 0.042	0.882 ± 0.026	0.891 ± 0.028	0.873 ± 0.044	0.887 ± 0.028	0.887 ± 0.033	0.892 ± 0.034	0.897 ± 0.028
Libra	F_M	0.511 ± 0.092	0.521 ± 0.111	0.517 ± 0.106	0.502 ± 0.149	0.300 ± 0.079	0.511 ± 0.100	0.500 ± 0.110	0.506 ± 0.128	**0.541 ± 0.098**
	G-M	0.676 ± 0.079	0.684 ± 0.096	0.675 ± 0.091	0.667 ± 0.124	0.519 ± 0.083	0.674 ± 0.091	0.662 ± 0.095	0.670 ± 0.112	**0.691 ± 0.081**
	AUC	0.696 ± 0.089	0.702 ± 0.096	0.705 ± 0.097	0.693 ± 0.094	0.546 ± 0.082	0.701 ± 0.090	0.695 ± 0.081	0.686 ± 0.093	**0.715 ± 0.101**
LEV	F_M	0.498 ± 0.050	0.518 ± 0.055	0.459 ± 0.042	0.533 ± 0.064	0.386 ± 0.028	0.493 ± 0.051	0.489 ± 0.056	0.541 ± 0.071	**0.564 ± 0.075**
	G-M	0.675 ± 0.034	0.745 ± 0.052	0.712 ± 0.042	0.698 ± 0.045	**0.779 ± 0.030**	0.736 ± 0.041	0.718 ± 0.052	0.721 ± 0.046	0.712 ± 0.053
	AUC	0.861 ± 0.037	0.854 ± 0.036	0.837 ± 0.034	**0.873 ± 0.045**	0.869 ± 0.039	0.858 ± 0.017	0.847 ± 0.033	0.853 ± 0.040	0.847 ± 0.058
Iris	F_M	0.854 ± 0.027	0.858 ± 0.050	0.820 ± 0.032	0.858 ± 0.038	0.772 ± 0.083	**0.879 ± 0.047**	0.844 ± 0.056	0.856 ± 0.024	0.855 ± 0.033
	G-M	0.906 ± 0.022	0.910 ± 0.037	0.878 ± 0.028	0.910 ± 0.029	0.829 ± 0.080	**0.926 ± 0.032**	0.899 ± 0.041	0.909 ± 0.017	0.906 ± 0.024
	AUC	0.984 ± 0.012	0.981 ± 0.014	0.980 ± 0.014	0.982 ± 0.014	**0.986 ± 0.014**	0.981 ± 0.014	0.983 ± 0.014	0.981 ± 0.014	0.980 ± 0.014
Heart	F_M	0.854 ± 0.027	0.854 ± 0.024	**0.857 ± 0.029**	0.854 ± 0.025	0.849 ± 0.027	0.854 ± 0.025	0.838 ± 0.036	0.854 ± 0.026	0.854 ± 0.026
	G-M	0.864 ± 0.026	0.864 ± 0.023	**0.867 ± 0.028**	0.864 ± 0.024	0.858 ± 0.026	0.864 ± 0.025	0.849 ± 0.035	0.864 ± 0.024	0.864 ± 0.025
	AUC	0.928 ± 0.017	**0.929 ± 0.017**	0.923 ± 0.016	0.929 ± 0.016	0.920 ± 0.018	0.928 ± 0.016	0.916 ± 0.027	0.928 ± 0.017	0.927 ± 0.017
Glass	F_M	0.640 ± 0.048	0.636 ± 0.034	0.633 ± 0.052	0.634 ± 0.044	0.611 ± 0.037	0.634 ± 0.040	0.644 ± 0.038	0.631 ± 0.059	**0.658 ± 0.045**
	G-M	0.715 ± 0.050	0.708 ± 0.039	0.703 ± 0.062	0.706 ± 0.053	0.647 ± 0.072	0.710 ± 0.041	0.709 ± 0.045	0.707 ± 0.060	**0.739 ± 0.045**
	AUC	0.821 ± 0.035	0.821 ± 0.032	0.814 ± 0.035	**0.824 ± 0.035**	0.803 ± 0.039	0.821 ± 0.034	0.809 ± 0.040	0.818 ± 0.036	0.804 ± 0.047
Haber	F_M	0.470 ± 0.060	0.460 ± 0.035	0.462 ± 0.048	0.453 ± 0.025	0.428 ± 0.062	0.419 ± 0.070	0.418 ± 0.104	0.441 ± 0.040	**0.483 ± 0.060**
	G-M	0.618 ± 0.049	0.611 ± 0.025	0.617 ± 0.039	0.604 ± 0.020	0.581 ± 0.048	0.569 ± 0.055	0.573 ± 0.095	0.597 ± 0.032	**0.631 ± 0.054**
	AUC	0.664 ± 0.053	0.637 ± 0.036	0.647 ± 0.039	0.643 ± 0.038	0.625 ± 0.061	0.622 ± 0.079	0.643 ± 0.104	0.627 ± 0.043	**0.684 ± 0.069**
Eucal.	F_M	0.268 ± 0.107	0.385 ± 0.069	0.293 ± 0.151	0.406 ± 0.110	0.399 ± 0.122	0.394 ± 0.133	**0.432 ± 0.070**	0.314 ± 0.041	0.318 ± 0.073
	G-M	0.411 ± 0.105	0.537 ± 0.056	0.436 ± 0.126	0.550 ± 0.096	**0.628 ± 0.224**	0.534 ± 0.101	0.578 ± 0.053	0.468 ± 0.033	0.460 ± 0.055
	AUC	0.838 ± 0.012	0.860 ± 0.015	0.847 ± 0.029	0.849 ± 0.025	0.782 ± 0.090	0.849 ± 0.029	**0.880 ± 0.008**	0.865 ± 0.019	0.841 ± 0.018
Heating	F_M	0.754 ± 0.060	0.758 ± 0.046	0.726 ± 0.036	**0.761 ± 0.030**	0.746 ± 0.063	0.741 ± 0.029	0.719 ± 0.062	0.744 ± 0.025	0.751 ± 0.023
	G-M	0.845 ± 0.035	0.851 ± 0.027	0.833 ± 0.018	0.850 ± 0.017	**0.852 ± 0.034**	0.843 ± 0.015	0.807 ± 0.048	0.838 ± 0.008	0.844 ± 0.009
	AUC	0.920 ± 0.022	0.921 ± 0.017	0.907 ± 0.029	0.925 ± 0.014	0.913 ± 0.043	0.919 ± 0.021	0.917 ± 0.017	0.917 ± 0.018	**0.931 ± 0.015**
Seg.	F_M	0.614 ± 0.016	0.618 ± 0.014	0.613 ± 0.031	0.621 ± 0.014	0.583 ± 0.031	**0.625 ± 0.019**	0.617 ± 0.012	0.613 ± 0.021	0.621 ± 0.013
	G-M	0.881 ± 0.009	0.881 ± 0.011	0.877 ± 0.030	0.883 ± 0.010	0.868 ± 0.015	0.874 ± 0.024	**0.885 ± 0.008**	0.878 ± 0.014	0.884 ± 0.010
	AUC	0.931 ± 0.013	0.932 ± 0.012	0.877 ± 0.027	0.932 ± 0.012	0.908 ± 0.033	**0.937 ± 0.013**	0.924 ± 0.017	0.932 ± 0.014	0.927 ± 0.017

important to oversample all sub-clusters in order to overcome *within-class* imbalance because ignoring some of them will bias the classifier toward oversampled ones.

Following is the description of the A-SUWO oversampling approach. The first step in oversampling each minority sub-cluster is to assign weights to each of its minority instances based on its average Euclidean distance to *NN*-nearest majority class neighbors. The reason for giving weights to minority instances lies on the fact that the minority instances closer to the majority instances are more prone to be misclassified and thus more important for classification. This is in contrast with (Barua et al., 2014), where the weights are assigned based on their average Euclidean distance to all candidate majority border instances even if they are far to some of the minority instances.

For the *h*th minority instance x_{jh} in minority sub-cluster $Cmin_j$, we find its k nearest neighbors according to its Euclidean distance among majority instances $y_{jh(v)}$ and record the distance $d(x_{jh}, y_{jh(v)})$, where $v = 1, \ldots, k$ represents the indices of the *NN*-nearest neighbors. We normalize the distance $d(x_{jh}, y_{jh(v)})$ by dividing by the number of features D to make it robust to datasets with different number of features. Therefore, we have:

$$\hat{d}(x_{jh}, y_{jh(v)}) = \frac{d(x_{jh}, y_{jh(v)})}{D} \tag{4}$$

Now, let's define $\Gamma(x_{jh}, y_{jh(v)})$ as the closeness factor between x_{jh} and $y_{jh(v)}$.

$$\Gamma(x_{jh}, y_{jh(v)}) = f_i\left(\frac{1}{\hat{d}(x_{jh}, y_{jh(v)})}\right) \tag{5}$$

where f_j is a cutoff function for sub-cluster C_j that prevents $\frac{1}{\hat{d}(x_{jh}, y_{jh(v)})}$ from becoming extremely large in the case when the two instances x_{jh} and $y_{jh(v)}$ become too close to each other. Therefore, f_j is defined as:

$$f_j\ (x) = \begin{cases} x\ if\ x \le TH_i \\ TH_j\ otherwise \end{cases} \tag{6}$$

TH_j is the largest value $f_j(x)$ can reach. In our method, TH_j is determined for each C_j automatically. This is achieved by finding the Euclidean distance of all minority instances x_{jh} in each sub-cluster to their closest majority instance $y_{jh(1)}$ and then determining $f(\frac{1}{\hat{d}(x_{jh}, y_{jh(1)})})$. TH_j is then set as the average of $f(\frac{1}{\hat{d}(x_{jh}, y_{jh(1)})})$.

$$TH_j = \sum_{j=1}^{R_j} f\left(\frac{1}{\hat{d}\left(x_{jh}, y_{jh(1)}\right)}\right) \tag{7}$$

where R_j is the number of instances in C_j. Determining TH_j automatically is a critical step in our method as our weighting algorithm runs for each minority sub-cluster separately and each sub-cluster requires a specific threshold.

In Eq. 5, we have taken the reciprocal of $\delta(x_{jh}, y_{jh(v)})$ because the minority instances closer to the majority instances should have higher weights, while the ones further from majority instances should have lower weights. Finally, the weights $W(x_{jh})$ are determined based on the Euclidean distance of x_{ij} from all *NN*-nearest neighbors as follows:

$$W\left(x_{jh}\right) = \sum_{v=1}^{k} \Gamma\left(x_{jh}, y_{jh(v)}\right) \tag{8}$$

The weights are converted into a probability distribution $P(x_i)$ by dividing each weight by the summation of all weights as follows:

$$P\left(x_{jh}\right) = \frac{W\left(x_{jh}\right)}{\sum_{h=1}^{R_j} W\left(x_{jh}\right)} \tag{9}$$

In the last step, each C_j, $j = 1, \ldots, n$ will be oversampled so that they will have size S_j. In order to oversample them, we first select an instance a in the sub-cluster by sampling from the probability distribution $P(x_{jh})$. Then, one of its *NN*-nearest neighbors b is randomly selected provided that they belong to the same sub-cluster and a new instance c is generated between a and b as follows:

$$c = \beta a + (1 - \beta) b \tag{10}$$

where β is a random number between 0 and 1. At the end, as can be seen in Fig. 4, each minority sub-cluster will have some synthetic instances that are generated between original minority instances and are closer to the majority instances. The proposed Adaptive Semi-Unsupervised Weighted Oversampling (A-SUWO) procedure is described as follows:

```
Algorithm 1. Adaptive Semi-Unsupervised Weighted Oversampling (A-SUWO)
Inputs:
  - $I$ : Original dataset to be oversampled.
  - $c_{thres}$: Coefficient to tune the threshold for clustering.
  - $NN$: Number of nearest neighbors to be found for each minority instance to determine the weights.
  - $NS$: Number of nearest neighbors used to identify noisy instances.
  - $K$: Number of folds in the $K$-fold Cross Validation.
Outputs:
  - $O$: Oversampled dataset.
Procedure:
  i. Semi-unsupervised clustering
    1. Remove noisy instances from the dataset.
    2. Determine $T$ using Eq. 1.
    3. Cluster majority class, which will result in $m$ sub-clusters $Cmaj_{i=1,\ldots,m}$.
    4. Assign each minority instance to a separate sub-cluster.
    5. Find the two closest sub-clusters $Cmin_a$ and $Cmin_b$.
    6. Check if there is any overlapping majority sub-cluster between $Cmin_a$ and $Cmin_b$.
    7. If yes, set their distance to infinity and return back to step 5. Else, merge $Cmin_a$ and $Cmin_b$ into one sub-cluster $Cmin_c$.
    8. Repeat steps 5 to 7 until the Euclidean distance between the closest sub-clusters is less than a threshold $T$.
  ii. Adaptive sub-cluster sizing
    1. Randomly split each minority sub-cluster into $K$ folds.
    2. Build a model using $K - 1$ folds from each minority sub-cluster in addition to all majority instances as the training set.
    3. Test the model using the remaining one fold from each minority sub-cluster.
    4. Determine Standardized Average Minority Error Rate $\hat{\varepsilon}_j^*$.
    5. Repeat steps 2 to 4 $K$ times.
    6. Determine final sizes $S_j$ for all sub-clusters $Cmin_{j=1,\ldots,n}$ using Eqs. 2 and 3.
  iii. Synthetic instance generation
    Determine the probability distribution for instances within each minority sub-cluster:
  - For each sub-cluster $j = 1, 2, \ldots, n$
    1. For all minority instances $x_{jh}$ in sub-cluster $Cmin_j$, find $NN$-nearest neighbors among majority instances.
    2. Determine $W(x_{jh})$ for each minority instance in $Cmin_j$ using Eq. 4 - 8 and by estimating $TH_j$.
    3. Transform the weights to a probability distribution $P(x_{jh})$ using Eq. 9.
  Oversample minority instances:
  - Initialize $O = I$.
  - For each sub-cluster $j = 1, 2, \ldots, n$
    1. Select a minority instance $a$ in sub-cluster $j$ by sampling from the probability distribution $P(x_{jh})$.
    2. Select randomly one of its $NN$-nearest neighbors $b$ that belongs to the same sub-cluster.
    3. Generate a new synthetic instance $c$ between $a$ and $b$ using Eq. 10.
    4. Add $c$ to set $O$.
    5. Repeat steps 1 to 4 until the sub-cluster size reaches $S_j$.
```

4. Results and discussion

The proposed A-SUWO method was evaluated on 16 publicly available datasets and compared with eight other oversampling techniques: (1) Random Oversampling, (2) SMOTE (Chawla et al., 2011), (3) Borderline SMOTE (Han et al., 2005), (4) Safe-Level SMOTE (Bunkhumpornpat et al., 2009), (5) Cluster SMOTE (Cieslak et al., 2006), (6) SBC (Jo & Japkowicz, 2004), (7) Clustering-Based Oversampling (CBOS) (Jo & Japkowicz, 2004), and (8) MWMOTE (Barua et al., 2014). The techniques were implemented using Matlab on a workstation with 64-bit Operating System, 4.00 GB RAM, and 2.67 GHz CPU. In this study, the performance measures used to compare the different methods are: F-measure, G-mean, and Area under Receiving Operator Characteristics Graph (AUC). F-measure can be calculated using Eqs. 11–13 in which minority instances are referred to as positive (*P*) and majority instances are referred to as negative (*N*) in the confusion matrix.

$$Precision = \frac{TP}{TP + FP} \tag{11}$$

$$Recall = \frac{TP}{TP + FN} \tag{12}$$

$$F_{measure} = \frac{\left(1 + \beta^2\right) * Recall * Precision}{\beta^2 * Recall + Precision} \tag{13}$$

Precision measures the exactness of the classifier that is, the number of instances labeled as positive (minority) that are actually positive. Recall measures the completeness of the classifier as the number of positive examples that were classified correctly as positive. The parameter β for $F_{measure}$ adjusts the relative importance between precision and recall.

The G-mean is determined as follows:

$$G_{mean} = \sqrt{\frac{TP}{TP + FN} \times \frac{TN}{TN + FP}} \tag{14}$$

G-mean considers the accuracy for both classes. As a result, if the minority class is ignored by the classifier and the majority class is favored, the classifier will obtain a low G-mean.

AUC is the area under ROC graph and is not sensitive to the distribution of the two classes which makes it suitable as a performance measure for the imbalanced problem. The ROC graph is obtained by plotting the True Positive Rate (TPR) over the False Positive Rate (FPR) as defined as follows:

$$TPR = \frac{TP}{N_p} \quad FPR = \frac{FP}{N_n} \tag{15}$$

where N_p is the number of positive (minority) instances and N_n is the number of negative (majority) instances.

More detailed information about the 16 datasets is shown in Table 1. For those datasets with more than two classes, they were converted into two-class datasets. In order to determine the mean and standard deviation of the performance measures for the oversampling methods, 4-fold stratified cross validation was used. Each experiment was repeated 3 times to report the average in order to alleviate the randomness effects on the results.

Four classifiers were used to evaluate the oversampling methods: Support Vector Machine with radial basis function (SVM) (Chang & Lin, 2011), Logistic Regression (McCullagh, 1984), Nearest Neighbors (KNN) (Cover & Hart, 1967), and Linear Discriminant Analysis (LDA) (Guo, Hastie, & Tibshirani, 2007). The parameters of the four classifiers and the nine sampling methods are optimized over a small set of values using stratified cross-validation and considering only the training set. The cross-validation criterion is G-mean because it is the only criteria that accounts for all values in the confusion matrix and provides the more reliable measure. In particular, for SVM, the parameters for both cost *C* and gamma γ were selected among the values $(2^{-1},\ 2^{0},\ 2^{1})$. For KNN, the number of nearest neighbors was selected among the values (4, 5, 6). Logistic Regression and LDA do not require any parameters to be tuned. For ASUWO, c_{thres} was selected among (1, 2), *NN* was selected among (3, 5). *NS* was selected among (4, 6) and *k* was set to 3.

Tables 2–5 show the results of the mean and standard deviation for our proposed A-SUWO method and the other eight sampling methods on the 16 datasets classified using the four classifiers. The best measures are shown in bold. A-SUWO obtains the best results according to at least one of the measures in 13 out of the 16 datasets when SVM and Logistic Regression were used and in 10 out of the 16 datasets when KNN and LDA were used. Additionally, the performance variability for A-SUWO does not vary significantly over the four-fold cross validation and 3 iterations.

The results are further summarized in Table 6, which shows each method's mean rankings in terms of F-measure, G-mean and AUC for all the tested datasets. For each dataset, the best performing method receives a ranking of 1 while the method with the worst performance receives a ranking of 9. Friedman's test followed by Holm's test were performed to verify the statistical significance of our method

Table 6
Results for mean ranking of the 9 methods averaged over the 16 datasets.

Measure	Random	SMOTE	Borderline SMOTE	Safe-level SMOTE	SBC	Cluster SMOTE	CBOS	MWMOTE	A-SUWO
Classification method: SVM									
F-measure	5.500	4.969	5.406	4.781	7.438	6.375	4.125	3.906	**2.625**
G-mean	5.688	5.406	5.219	4.938	6.500	5.750	4.875	3.938	**2.688**
AUC	5.906	5.781	5.844	5.344	6.125	5.438	4.313	3.750	**2.500**
Classification method: KNN									
F-measure	5.594	5.125	5.750	4.156	8.000	4.250	5.500	4.438	**2.188**
G-mean	5.500	5.125	5.750	4.375	8.188	4.250	5.313	4.125	**2.375**
AUC	5.906	4.250	5.875	4.094	6.813	**2.500**	6.250	4.938	4.375
Classification method: Logistic Regression									
F-measure	5.000	4.688	5.594	5.000	7.938	4.406	5.750	4.688	**1.938**
G-mean	4.500	4.250	5.281	5.750	8.250	5.031	5.000	4.375	**2.563**
AUC	5.031	3.500	6.594	3.563	8.188	3.938	5.875	4.813	**3.500**
Classification method: Linear Discriminant Analysis									
F-measure	4.750	4.313	5.875	3.250	8.188	5.000	6.563	5.188	**1.875**
G-mean	4.313	4.313	5.875	3.688	8.188	5.625	6.125	4.688	**2.188**
AUC	4.750	**3.094**	6.375	3.188	7.875	4.313	6.500	4.500	4.406

Table 7
Results for Friedman's test.

F-measure		G-mean		AUC	
Classification method	P-Value	Classification method	P-value	Classification method	P-value
SVM	0.005956**	SVM	0.001138**	SVM	3.69E−05**
KNN	1.35E−06**	KNN	1.34E−06**	KNN	0.000148**
Logistic Regression	1.41E−06**	Logistic Regression	3.98E−06**	Logistic Regression	3.31E−07**
LDA	1.72E−09**	LDA	4.84E−08**	LDA	6.55E−07**

compared to the other sampling methods. Friedman test is a non-parametric equivalent of the repeated-measures ANOVA. The null hypothesis in Friedman test is whether all classifiers are performing similarly in mean rankings. The results for the Friedman test are shown in Table 7. As can be observed from the results, for all four classifiers and all three measures, there exists enough evidence at $\alpha = 0.05$ to reject the null hypothesis, which means that classifiers are not performing similarly.

Since the null hypothesis is rejected for all three performance measures, a post-hoc test is applied. The Holm's test was used where our method was considered as the control method. Holm's test is the non-parametric analog of multiple t-test that adjusts α to compensate for multiple comparisons in a step-down procedure. The null hypothesis is whether ASUWO performs better than other methods as the control algorithm. Table 8 shows the adjusted α and the corresponding p-value for each method. As can be seen from the table, the proposed A-SUWO method outperforms all other methods based on all three measures when SVM was used as the classifier. When KNN, Logistic Regression and LDA was used as the classifier, A-SUWO is significantly better than all other methods in terms of G-mean and F-measure. We can also observe that the cluster based undersampling method (Yen & Lee, 2009) was the method that performs the worst based on all three measures for all classifiers. On the other side, SMOTE and Cluster SMOTE perform well according to AUC, while MWMOTE and Safe-Level SMOTE perform satisfactory according to F-measure and G-mean. Moreover, it can be observed that methods that perform well in terms of G-mean also perform well in terms of F-measure while they do not perform well in terms of AUC.

Our results also indicate that compared to other methods, our method works better for datasets with higher imbalance ratio like Balance and LEV datasets. This is because in such datasets, minority instances are highly sparse meaning that there exists small minority clusters in the dataset. In other words, such datasets have high *within-class* imbalance. Therefore, it is very important to identify these small sub-clusters and emphasize them through oversampling as in cluster-based methods. Results also show that our method outperforms other cluster-based methods in most datasets. This is because, unlike the cluster-based methods, we adaptively determine sub-cluster sizes and oversample minority instances based on their distance to the majority class.

4.1. Choosing parameters for A-SUWO

A-SUWO requires four parameters to be defined: c_{thres}, *NN, NS* and *k*. In this section, we briefly explain how to choose appropriate values for these parameters. We also perform sensitivity analysis by running A-SUWO with different set of values for each parameter. The results are shown in Table 9.

1. c_{thres}: This parameter was used to adjust the threshold for agglomerative clustering in Section 2.1. Larger values of c_{thres} will result in smaller clusters with larger sizes while smaller values of c_{thres} will result in larger clusters with smaller sizes. Its optimum value depends on the dataset. Generating large sized clusters as a result of large c_{thres} will increase the chance of over-generalization or generation of overlapping instances. On the other hand, generating small sized clusters will result in over-fitting or generation of less diverse synthetic instances. As can be seen from Table 9, a good range for c_{thres} is between 0.7 and 2. Actually, the G-mean for

Table 8
Holm's test P-value - control algorithm: A-SUWO.

i	$\alpha_{0.10}$	F-measure		G-mean		AUC	
		Method	P-value	Method	P-value	Method	P-value
Classification model: SVM							
1	0.0125	SBC	4.12E−05**	SBC	9.06E−05**	SBC	3.34E−07**
2	0.0143	Cluster SMOTE	0.000781**	Random	0.000217**	Cluster SMOTE	5.38E−05**
3	0.0167	Random	0.000973**	Border SMOTE	0.000277**	Random	0.001492**
4	0.0200	SMOTE	0.002493**	SMOTE	0.000351**	Border SMOTE	0.002036**
5	0.0250	Border SMOTE	0.004471**	Cluster SMOTE	0.001207**	SMOTE	0.007747**
6	0.0333	Safe-Level SMOTE	0.010068*	Safe-Level SMOTE	0.001657**	Safe-Level SMOTE	0.012975*
7	0.0500	CBOS	0.011934*	CBOS	0.030607*	CBOS	0.060668*
8	0.1000	MWMOTE	0.098353*	MWMOTE	0.098353*	MWMOTE	0.092873*
Classification model: KNN							
1	0.0125	SBC	9.68E−10**	SBC	9.68E−10**	SBC	0.005911**
2	0.0143	Border SMOTE	0.000117**	Border SMOTE	0.000245**	CBOS	0.026404
3	0.0167	Random	0.000217**	Random	0.000624**	Random	0.056886
4	0.0200	CBOS	0.000312**	CBOS	0.001207**	Border SMOTE	0.060668
5	0.0250	SMOTE	0.001207**	SMOTE	0.002254**	MWMOTE	0.280638
6	0.0333	MWMOTE	0.010068*	Safe-Level SMOTE	0.019434*	SMOTE	0.551361
7	0.0500	Cluster-SMOTE	0.01658*	Cluster SMOTE	0.026404*	Safe-Level SMOTE	0.614273
8	0.1000	Safe-level SMOTE	0.02101*	MWMOTE	0.035351*	Cluster SMOTE	0.973596
Classification model: Logistic Regression							
1	0.0125	SBC	2.88E−10	SBC	2.13E−09**	SBC	6.45E−07**
2	0.0143	CBOS	4.12E-05	Safe-Level SMOTE	0.000497**	Border SMOTE	0.000699**
3	0.0167	Border SMOTE	7.96E-05	Border SMOTE	0.002493**	CBOS	0.007086**
4	0.0200	Random	0.000781	Cluster SMOTE	0.005391**	Random	0.056886
5	0.0250	Safe-Level SMOTE	0.000781	CBOS	0.005911**	MWMOTE	0.087622
6	0.0333	SMOTE	0.002254	Random	0.022694*	Cluster SMOTE	0.325689
7	0.0500	MWMOTE	0.002254	MWMOTE	0.030607*	Safe-Level SMOTE	0.474266
8	0.1000	Cluster SMOTE	0.005391	SMOTE	0.040681*	SMOTE	0.500000
Classification model: LDA							
1	0.0125	SBC	3.53E−11**	SBC	2.88E−10**	SBC	0.00017**
2	0.0143	CBOS	6.45E−07**	CBOS	2.38E−05**	CBOS	0.015293
3	0.0167	Border SMOTE	1.80E−05**	Border SMOTE	6.99E−05**	Border SMOTE	0.02101
4	0.0200	MWMOTE	0.000312**	Cluster SMOTE	0.000192**	Random	0.361286
5	0.0250	Cluster SMOTE	0.000624**	MWMOTE	0.004912**	MWMOTE	0.461433
6	0.0333	Random	0.001492**	Random	0.014093*	Cluster SMOTE	0.538567
7	0.0500	SMOTE	0.005911*	SMOTE	0.014093*	Safe-Level SMOTE	0.895934
8	0.1000	Safe-Level SMOTE	0.077790*	Safe-Level SMOTE	0.060668*	SMOTE	0.912378

Table 9
Sensitivity analysis on A-SUWO parameters using SVM.

Dataset	G-mean measure for different values of c_{thres}		G-mean measure for different values of *NN*		G-mean measure for different values of *NS*		G-mean measure for different values of *k*	
	c_{thres}	G-mean	*NN*	G-mean	*NS*	G-mean	*k*	G-mean
Haberman	0.3	0.516 ± 0.047	1	0.516 ± 0.047	1	0.550 ± 0.054	1	0.582 ± 0.041
	0.7	0.577 ± 0.017	2	0.577 ± 0.017	2	0.584 ± 0.012	2	**0.594 ± 0.046**
	1.0	0.574 ± 0.039	3	0.574 ± 0.039	3	0.587 ± 0.028	3	0.552 ± 0.066
	1.5	0.541 ± 0.019	4	0.541 ± 0.019	4	0.602 ± 0.018	4	0.559 ± 0.067
	2	**0.611 ± 0.040**	5	**0.611 ± 0.040**	5	**0.604 ± 0.016**	5	0.557 ± 0.085
	2.5	0.584 ± 0.063	7	0.584 ± 0.063	7	0.585 ± 0.015	6	0.550 ± 0.062
	3	0.557 ± 0.031	10	0.557 ± 0.031	10	0.584 ± 0.016	8	0.588 ± 0.069
	8	0.557 ± 0.031	15	0.557 ± 0.031	15	0.585 ± 0.019	10	0.583 ± 0.019
Ecoli	0.3	0.936 ± 0.026	1	0.941 ± 0.018	1	0.939 ± 0.015	1	0.941 ± 0.024
	0.7	0.935 ± 0.025	2	0.940 ± 0.010	2	0.943 ± 0.011	2	**0.942 ± 0.029**
	1.0	**0.937 ± 0.028**	3	0.944 ± 0.012	3	0.941 ± 0.010	3	0.938 ± 0.027
	1.5	0.936 ± 0.030	4	0.944 ± 0.012	4	0.941 ± 0.010	4	0.940 ± 0.027
	2	0.933 ± 0.026	5	**0.946 ± 0.010**	5	0.942 ± 0.011	5	0.936 ± 0.026
	2.5	0.933 ± 0.026	7	0.942 ± 0.011	7	**0.944 ± 0.012**	6	0.940 ± 0.027
	3	0.933 ± 0.026	10	0.939 ± 0.009	10	0.942 ± 0.010	8	0.938 ± 0.026
	8	0.933 ± 0.026	15	0.939 ± 0.009	15	0.942 ± 0.010	10	0.938 ± 0.026
Wine	0.3	0.972 ± 0.021	1	0.967 ± 0.028	1	0.976 ± 0.019	1	**0.985 ± 0.015**
	0.7	0.970 ± 0.021	2	0.986 ± 0.021	2	0.976 ± 0.019	2	**0.985 ± 0.027**
	1.0	**0.975 ± 0.017**	3	0.978 ± 0.018	3	**0.979 ± 0.015**	3	**0.985 ± 0.015**
	1.5	0.967 ± 0.015	4	0.978 ± 0.018	4	**0.979 ± 0.015**	4	0.981 ± 0.012
	2	0.967 ± 0.015	5	0.986 ± 0.011	5	**0.979 ± 0.015**	5	**0.985 ± 0.015**
	2.5	0.969 ± 0.015	7	**0.993 ± 0.011**	7	0.976 ± 0.019	6	0.978 ± 0.026
	3	0.969 ± 0.015	10	**0.993 ± 0.011**	10	0.976 ± 0.019	8	0.981 ± 0.012
	8	0.969 ± 0.015	15	**0.993 ± 0.011**	15	0.976 ± 0.019	10	0.978 ± 0.026
Breast	0.3	0.695 ± 0.034	1	0.690 ± 0.058	1	0.723 ± 0.031	1	0.736 ± 0.114
	0.7	0.705 ± 0.068	2	0.709 ± 0.062	2	0.731 ± 0.074	2	**0.747 ± 0.034**
	1.0	**0.732 ± 0.054**	3	0.722 ± 0.056	3	**0.750 ± 0.060**	3	0.706 ± 0.072
	1.5	0.704 ± 0.029	4	0.730 ± 0.056	4	**0.750 ± 0.060**	4	0.710 ± 0.074
	2	0.692 ± 0.038	5	**0.739 ± 0.067**	5	0.742 ± 0.049	5	0.725 ± 0.065
	2.5	0.692 ± 0.038	7	0.653 ± 0.032	7	0.742 ± 0.049	6	0.716 ± 0.069
	3	0.680 ± 0.048	10	0.654 ± 0.034	10	0.742 ± 0.049	8	0.726 ± 0.062
	8	0.680 ± 0.048	15	0.665 ± 0.021	15	0.727 ± 0.064	10	0.706 ± 0.053
Libra	0.3	0.722 ± 0.045	1	0.726 ± 0.015	1	0.697 ± 0.029	1	0.781 ± 0.036
	0.7	0.751 ± 0.028	2	0.790 ± 0.024	2	0.714 ± 0.063	2	0.782 ± 0.013
	1.0	0.778 ± 0.041	3	**0.799 ± 0.013**	3	**0.763 ± 0.024**	3	0.781 ± 0.036
	1.5	**0.787 ± 0.045**	4	**0.799 ± 0.013**	4	0.754 ± 0.037	4	0.781 ± 0.036
	2	0.772 ± 0.042	5	**0.799 ± 0.013**	5	0.763 ± 0.042	5	**0.799 ± 0.013**
	2.5	0.772 ± 0.042	7	0.781 ± 0.037	7	0.731 ± 0.089	6	0.735 ± 0.049
	3	0.772 ± 0.042	10	0.781 ± 0.037	10	0.742 ± 0.073	8	0.781 ± 0.026
	8	0.772 ± 0.042	15	0.781 ± 0.037	15	0.742 ± 0.073	10	0.790 ± 0.023

all values of c_{thres} larger than 3 is similar because all clusters are merged into one cluster.

2. *NN*: This parameter determines the number of nearest neighbors used to assign weights to each minority instance. The weight for each minority instance depends on the average closeness factor to all *NN*-nearest neighbors from the majority class. If *NN* is selected as a large value, the algorithm assigns almost similar weights to all minority instances even if they are far away from the majority class. This is because the closeness factors are averaged over a large number of nearest neighbors. On the other hand, if *NN* is selected as a small value, then the weights could be very sensitive to noisy majority instances. As can be seen from Table 9, a reasonable value for *NN* could be selected between 3 and 7.
3. *NS*: This parameter is used to find noisy instances. If all *NS* nearest neighbors of an instance are from a different class, then the instance is considered as noise in our method. If *NS* is selected as a large value, then the method is not able to find noisy instances whereas if *NS* is selected as a small value, the method will consider many of the valid instances as noise. As can be seen from Table 9, a reasonable value for *NS* can be between 3 and 7.
4. *k*: This parameter determines the number of folds in our adaptive cluster sizing. The larger this parameter gets the more expensive the computation becomes as the classification method used in A-SUWO to determine the complexity of each cluster is required to run more times. As can be seen from Table 9, *k* can be selected between 2 and 5.

5. Conclusions

In this paper, a new oversampling algorithm called Adaptive Semi-Unsupervised Weighted Oversampling (A-SUWO) has been presented for imbalanced binary dataset classification. The advantages of A-SUWO are that it avoids generating overlapping synthetic instances by considering the majority instances when clustering minority instances; it determines the sub-cluster sizes adaptively using the standardized average error rate and cross-validation; it oversamples the sub-clusters by assigning weights to their instances to avoid over-generalization; and it does not ignore isolated sub-clusters. A-SUWO was tested on 16 publicly available datasets with different imbalance ratios and compared with other sampling techniques using different types of classifiers. Results show that our method performs significantly better compared to other sampling methods in most datasets and in larger datasets with higher imbalance ratio. As future work, the application of A-SUWO to multi-class classification problems will be studied.

Reference

Akbani, R., Kwek, S., & Japkowicz, N. (2004). Applying support vector machines to imbalanced datasets. In *Machine Learning: ECML 2004* (pp. 39–50). Springer.

Alshomrani, S., Bawakid, A., Shim, S.-O., Fernández, A., & Herrera, F. (2015). A proposal for evolutionary fuzzy systems using feature weighting: Dealing with overlapping in imbalanced datasets. *Knowledge-Based Systems, 73*, 1–17.

Barua, S., Islam, M., Yao, X., & Murase, K. (2014). MWMOTE–Majority Weighted Minority Oversampling Technique for Imbalanced Data Set Learning. *IEEE Transactions on Knowledge and Data Engineering, 26*, 405–425.

Beyan, C., & Fisher, R. (2014). Classifying imbalanced data sets using similarity based hierarchical decomposition. *Pattern Recognition, 48*(5), 1653–1672.

Bunkhumpornpat, C., Sinapiromsaran, K., & Lursinsap, C. (2009). Safe-level-smote: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem. In *Advances in Knowledge Discovery and Data Mining* (pp. 475–482). Springer.

Bunkhumpornpat, C., Sinapiromsaran, K., & Lursinsap, C. (2012). DBSMOTE: density-based synthetic minority over-sampling technique. *Applied Intelligence, 36*, 664–684.

Chang, C.-C., & Lin, C.-J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST), 2*, 27.

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: synthetic minority over-sampling technique. *Journal of Artificial Intelligence Resear, 16*, 321–357.

Cieslak, D. A., & Chawla, N. V. (2008). Start globally, optimize locally, predict globally: Improving performance on imbalanced data. In *Eighth IEEE International Conference on Data Mining, 2008. ICDM'08* (pp. 143–152). IEEE.

Cieslak, D. A., Chawla, N. V., & Striegel, A. (2006). Combating imbalance in network intrusion datasets. In *IEEE Int. Conf. Granular Comput.* (pp. 732–737).

Cover, T. M., & Hart, P. E. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory, 13*, 21–27.

Díez-Pastor, J. F., Rodríguez, J. J., García-Osorio, C. I., & Kuncheva, L. I. (2015). Diversity techniques improve the performance of the best imbalance learning ensembles. *Information Sciences, 325*, 98–117.

Galar, M., Fernandez, A., Barrenechea, E., Bustince, H., & Herrera, F. (2012). A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches. *Systems, IEEE Transactions on Man, and Cybernetics, Part C: Applications and Reviews, 42*, 463–484.

Guo, Y., Hastie, T., & Tibshirani, R. (2007). Regularized linear discriminant analysis and its application in microarrays. *Biostatistics, 8*, 86–100.

Han, H., Wang, W. Y., & Mao, B. H. (2005). Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning. *Advances in intelligent computing* (pp. 878–887). Springer.

He, H., Bai, Y., Garcia, E., & Li, S. (2008). ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In *IEEE International Joint Conference on Neural Networks, 2008. IJCNN 2008.(IEEE World Congress on Computational Intelligence)* (pp. 1322–1328). IEEE.

He, H., & Garcia, E. A. (2009). Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering, 21*, 1263–1284.

Japkowicz, N., & Stephen, S. (2002). The class imbalance problem: A systematic study. *Intelligent data analysis, 6*, 429–449.

Jo, T., & Japkowicz, N. (2004). Class imbalances versus small disjuncts. *ACM Sigkdd Explorations Newsletter, 6*, 40–49.

Kotsiantis, S., Kanellopoulos, D., & Pintelas, P. (2006). Handling imbalanced datasets: A review. *GESTS International Transactions on Computer Science and Engineering, 30*, 25–36.

Li, P., Chan, K. L., Fu, S., & Krishnan, S. M. (2014). Kernel Machines for Imbalanced Data Problem in Biomedical Applications. In *Support Vector Machines Applications* (pp. 221–268). Springer.

López, V., Fernández, A., García, S., Palade, V., & Herrera, F. (2013). An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. *Information Sciences, 250*, 113–141.

McCullagh, P. (1984). Generalized linear models. *European Journal of Operational Research, 16*, 285–292.

Piras, L., & Giacinto, G. (2012). Synthetic pattern generation for imbalanced learning in image retrieval. *Pattern Recognition Letters, 33*, 2198–2205.

Prati, R. C., Batista, G. E., & Silva, D. F. (2014). Class imbalance revisited: a new experimental setup to assess the performance of treatment methods. *Knowledge and Information Systems*, 1–24.

Thammasiri, D., Delen, D., Meesad, P., & Kasap, N. (2014). A critical assessment of imbalanced class distribution problem: The case of predicting freshmen student attrition. *Expert Systems with Applications, 41*, 321–330.

Voorhees, E. M. (1986). Implementing agglomerative hierarchic clustering algorithms for use in document retrieval. *Information Processing & Management, 22*, 465–476.

Wei, W., Li, J., Cao, L., Ou, Y., & Chen, J. (2013). Effective detection of sophisticated online banking fraud on extremely imbalanced data. *World Wide Web, 16*, 449–475.

Yen, S.-J., & Lee, Y.-S. (2009). Cluster-based under-sampling approaches for imbalanced data distributions. *Expert Systems with Applications, 36*, 5718–5727.

Zheng, Z., Wu, X., & Srihari, R. (2004). Feature selection for text categorization on imbalanced data. *ACM Sigkdd Explorations Newsletter, 6*, 80–89.

Zhou, L. (2013). Performance of corporate bankruptcy prediction models on imbalanced dataset: The effect of sampling methods. *Knowledge-Based Systems, 41*, 16–25.