SCUT: Multi-Class Imbalanced Data Classification using SMOTE and Cluster-based Undersampling

Astha Agrawal[1], Herna L. Viktor[1] and Eric Paquet[1,2]
[1]*School of Electrical Engineering and Computer Science, University of Ottawa, Ottawa, Ontario, Canada*
[2]*National Research Council of Canada, Ottawa, Ontario, Canada*

Keywords: Multi-Class Imbalance, Undersampling, Oversampling, Classification, Clustering.

Abstract: Class imbalance is a crucial problem in machine learning and occurs in many domains. Specifically, the two-class problem has received interest from researchers in recent years, leading to solutions for oil spill detection, tumour discovery and fraudulent credit card detection, amongst others. However, handling class imbalance in datasets that contains multiple classes, with varying degree of imbalance, has received limited attention. In such a multi-class imbalanced dataset, the classification model tends to favour the majority classes and incorrectly classify instances from the minority classes as belonging to the majority classes, leading to poor predictive accuracies. Further, there is a need to handle both the imbalances between classes as well as address the selection of examples within a class (i.e. the so-called within class imbalance). In this paper, we propose the SCUT hybrid sampling method, which is used to balance the number of training examples in such a multi-class setting. Our SCUT approach oversamples minority class examples through the generation of synthetic examples and employs cluster analysis in order to undersample majority classes. In addition, it handles both within-class and between-class imbalance. Our experimental results against a number of multi-class problems show that, when the SCUT method is used for pre-processing the data before classification, we obtain highly accurate models that compare favourably to the state-of-the-art.

1 INTRODUCTION

In an imbalanced dataset used for classification, the sizes of one or more classes are much greater than the other classes. The classes with the larger number of instances are called majority classes and the classes with the smaller number of instances are referred to as the minority classes. Intuitively, since there are a large number of majority class examples, a classification model tends to favour majority classes while incorrectly classifying the examples from the minority classes. However, in imbalanced datasets, we are often more interested in correctly classifying the minority classes. For instance, in a two class setting within the medical domain, if we are classifying patients' condition, the minority class (e.g. cancer) is of more interest than the majority class (e.g. cancer free). In practice, many problems have more than two classes. For example, in bioinformatics, protein family classification, where a protein may belong to very small families within the large Protein Data Bank repository (Viktor et. al, 2013), as well as protein fold prediction, are examples of multi-class problems. Typically, in such a multi-class imbalanced dataset, there are multiple classes that are underrepresented, that is, there may be multiple majority classes and multiple minority classes, resulting in skewed distributions.

A number of research studies have been realized in order to improve classification performance on imbalanced *binary* class datasets, in which there is one majority class and one minority class. However, improving the performance on imbalanced multi-class datasets has not been researched as extensively. Consequently, most existing techniques for improving classification performance on imbalanced datasets are designed to be applied directly on binary class imbalanced datasets. These methods cannot be applied directly on multi-class datasets (Wang and Yao, 2012). Rather, class decomposition is usually used to convert a multi-class problem into a binary class problem. For instance, the One-versus-one (OVO) approach employs multiple classifiers for each possible pair of classes, discarding the remaining instances that do not belong to the pair under consideration. The One-versus-all (OVA) approach, on the other hand,

Agrawal, A., Viktor, H. and Paquet, E..
SCUT: Multi-Class Imbalanced Data Classification using SMOTE and Cluster-based Undersampling.
In *Proceedings of the 7th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2015) - Volume 1: KDIR*, pages 226-234
ISBN: 978-989-758-158-8
Copyright © 2015 by SCITEPRESS – Science and Technology Publications, Lda. All rights reserved

considers one class as the positive class, and merges the remaining classes to form the negative class. For '*n*' classes, '*n*' classifiers are used, and each class acts as the positive class once (Fernández et al., 2010). Subsequently, the results from different classifiers are combined in order to reach a final decision. Interested readers are referred to (Ramanan et al., 2007) for detailed discussions of the OVO and OVA approaches. However, combining results from classifiers that are trained on different sub-problems may result in classification errors (Wang and Yao, 2012). In addition, in OVO, each classifier is trained only on a subset of the dataset, which may lead to some data regions being left unlearned. In this paper, we propose a different method to improve classification performance on multi-class imbalanced datasets which preserves the structure of the data, without converting the dataset into a binary class problem.

In addition to between-class imbalance (i.e. the imbalance in the number of instances in each classes), within-class imbalance is also commonly observed in datasets. Such a situation occurs when a class is composed of different sub-clusters and these sub-clusters do not contain the same number of examples (Japkowicz, 2001). It follows that between-class and within-class imbalances both affect classification performance. In an attempt to address these two problems, and in order to improve classification performance on imbalanced datasets, sampling methods are often used for pre-processing the data prior to using a classifier to build a classification model.

Sampling methods focus on adapting the class distribution in order to reduce the between-class imbalance. Sampling methods may be divided into two categories, namely undersampling and oversampling. Undersampling reduces the number of majority class instances and oversampling increases the number of minority class instances. Unfortunately, both random oversampling and undersampling techniques present some weaknesses. For instance, random oversampling adds duplicate minority class instances to the minority class. This may result in smaller and more specific decision regions causing the learner to over-fit the data. Also, oversampling may increase the training time. Random undersampling randomly takes away some instances from the majority class. A drawback of this method is that useful information may be taken away (Han et al., 2005). Further, when performing random undersampling, if the dataset has within-class imbalance and some sub-clusters are represented by very few instances, the probability that instances from these sub-clusters be retained is relatively low. Consequently, these instances may remain unlearned.

SMOTE represents an improvement over random oversampling in that the minority class is oversampled by generating "synthetic" examples (Chawla et. al., 2002). However, in highly imbalanced datasets, too much oversampling (i.e. oversampling using a high sampling percentage) may result in overfitting. This is especially important in a multi-class setting where there are a number of minority classes with very few examples. Further, in a multi-class setting, there is a need to find the correct balance, in terms of number of examples, between multiple classes. In order to address this issue, we propose an algorithm called SCUT (SMOTE and Clustered Undersampling Technique) which combines SMOTE and cluster-based undersampling in order to handle between-class and within-class imbalance.

Undersampling is required to balance the dataset without using excessive oversampling. If majority class instances are randomly selected, small disjuncts with less representative data may remain unlearned. Clustering the majority classes helps identify sub-concepts, and if at least one instance is selected from each sub-concept (cluster) while doing undersampling, this issue might be addressed (Sobhani et. al, 2014). This implies that the scenario of having unlearned regions when within-class imbalance exists, is reduced. In this setting, combining clustering and undersampling makes sense as it addresses the disadvantage of random undersampling. To this end, Yen and Lee proposed several cluster-based undersampling approaches to select representative data as training data to improve the classification accuracy for the minority class (Yen and Lee, 2009). The main idea behind their cluster-based undersampling approaches was based on the assumption that each dataset has different clusters and each cluster seems to have distinct characteristics. Subsequently, from each cluster, a suitable number of majority class samples were selected (Yen and Lee, 2009). Rahman and Davis also used a cluster-based undersampling technique for classifying imbalanced cardiovascular data that not only balances the data in a dataset, but further selects good quality training set data for building classification models (Rahman and Davis, 2013).

Chawla et al. combined random undersampling with SMOTE, so that the minority class had a larger presence in the training set. By combining undersampling and oversampling, the initial bias of the learner towards the majority class is reversed in

the favour of the minority class (Chawla et al., 2002). In summary, cluster-based undersampling ensures that all sub-concepts are adequately represented. When used in conjunction with SMOTE, the hybrid sampling method thus aid to ensure that between-class imbalance is reduced without excessive use of oversampling and undersampling.

This paper is organized as follows. Section 2 contains a description of the proposed method. In Section 3, the experimental setup and results are presented while Section 4 concludes the paper and discusses our future plans.

2 SCUT ALGORITHM

Our SCUT algorithm combines both undersampling and oversampling techniques in order to reduce the imbalance between classes in a multi-class setting. The pseudocode for our SCUT method is shown in Figure 1.

For undersampling, we employ a cluster-based undersampling technique, using the Expectation Maximization (EM) algorithm (Dempster et al., 1977). The EM algorithm replaces the hard clusters by a probability distribution formed by a mixture of Gaussians. Instead of being assigned to a particular cluster, each member has a certain probability to belong to a particular Gaussian distribution of the mixture. The parameters of the mixture, including the number of Gaussians, are determined with the Expectation Maximization algorithm. An advantage of using EM is that the number of clusters does not have to be specified beforehand. EM clustering may be used to find both hard and soft clusters. That is, EM assigns a probability distribution to each instance relative to each particular cluster (Dempster et al., 1997).

The SCUT algorithm proceeds as follows. The dataset is split into n parts, namely $D_1, D_2, D_3 \ldots D_n$, where n is the number of classes and D_i represents a single class. Subsequently, the mean (m) of the number of instances of all the classes is calculated.

i) For all classes that have a number of instances less than the mean m, oversampling is performed in order to obtain a number of instances equal to the mean. The sampling percentage used for SMOTE is calculated such that the number of instances in the class after oversampling is equal to m.

ii) For all classes that have a number of instances greater than the mean m, undersampling is conducted to obtain a number of instances equal to the mean. Recall that the EM technique is used to discover the clusters within each class (Dempster et al., 1977). Subsequently, for each cluster within the current class, instances are randomly selected such that the total number of instances from all the clusters is equal to m. Therefore, instead of fixing the number of instances selected from each cluster, we fix the total number of instances. It follows that a different number of instances may be selected from the various clusters. However, we aim to select the instances as uniformly as possible. The selected instances are combined together in order to obtain m instances (for each class).

iii) All classes for which the number of instances is equal to the mean m are left untouched.

```
Input: Dataset D with n classes
Output: Dataset D' with all classes
having m instances, where m is the mean
number of instances of all classes

Split D into D1, D2, D3, ..., Dn where Di
is a single class
Calculate m

Undersampling:
For each Di, i=1,2, ... , n where
number of instances >m
    Cluster Di using EM algorithm
    For each cluster Ci, i = 1,2,
    ... , k
          Randomly select instances
         from Ci
          Add selected instances to
         Ci'
    End For
    C = Ø
    For i=1,2, ... , k
      C = C U Ci'
    End For
    Di' = C
End For

Oversampling:
For each Di, i=1,2, ... , n where
number of instances <m
    Apply SMOTE on Di to get Di'
End For

For each Di, i=1,2, ... , n where
number of instances = m
    Di' = Di

D' = Ø
For i = 1,2, ... , n
   D' = D' U Di'
End For
Return D'
```

Figure 1: SCUT Algorithm.

Finally, all the classes are merged together in order to obtain a dataset *D',* where all the classes have *m* instances. Classification may be performed on *D'* using an appropriate classifier.

For instance, one of the datasets used in our work is the Lymphography dataset, as obtained from the KEEL repository (Alcalá-Fdez et al., 2011). This dataset concerns detecting the presence of a lymphoma, together with its current status, and contains four (4) classes (normal, metastases, malignant-lymphoma and fibrosis), with 2, 81, 61 and 4 examples, respectively. That is, the dataset has a high level of imbalance and contains two majority and two minority classes. The dataset is split into four (4) classes and the mean is 37.

i) For class 1, the number of instances is 2, so SMOTE is applied with a sampling percentage of 1850% in order to obtain 37 instances.

ii) For class 2, the number of examples is 81, so EM is applied and 3 clusters are obtained, with the numbers of instances equal to 29, 17 and 35 respectively. In order to obtain a total of 37 instances, 12, 12 and 13 instances are randomly selected from the clusters.

iii) For class 3, the number of instances is 61. When EM is applied, only one cluster is obtained. Next, 37 instances are randomly selected from this one cluster.

iv) The number of instances of class 4 is equal to 4, so SMOTE is applied with a sampling percentage of 925% in order to obtain 37 instances.

Lastly, the classes are merged together and a new dataset of 148 instances (in which each class has 37 examples) is obtained. The next section discusses our experimental setup and results.

3 EXPERIMENTATION

We implemented our SCUT algorithm by extending WEKA, an open source data mining tool that was developed at the University of Waikato. For classification, the WEKA implementations of four classifiers, namely J48 (decision tree), SMO (support vector machine), Naïve Bayes and IBk (Nearest Neighbour), were used. For IBk, the number of nearest neighbours (*k*) was set to five (5), by inspection. Default values for the other parameters were used.

A ten-fold cross validation approach was used for testing and training. Ten-fold cross validation has been shown to be an effective testing methodology when datasets are not too small, since each fold is a good representation of the entire dataset (Japkowicz, 2001).

3.1 Benchmarking Datasets

Seven multi-class datasets from the KEEL repository (Alcalá-Fdez et al., 2011) and the Wine Quality dataset from the UCI repository (Lichman, 2013) (Cortez et al., 2009) were used in the experiments. The details of these datasets are summarized in Table 1. The table shows that the number of classes in the datasets varies from three (3) to ten (10) and the number of training examples range from 148 to 6497. Here, the levels of imbalance and numbers of classes with majority and minority instances vary considerably.

The WEKA implementation of the EM cluster analysis algorithm was used. Recall that the EM approach employs probabilistic models which imply that the number of clusters does not have to be specified in advance. Therefore, a major strength of EM is that it determines the number of clusters that must be created by cross validation. In order to determine the number of clusters, cross validation is performed as follows:

1. Initially, the number of clusters is set to one (1).
2. The training set is split randomly into ten (10) folds. The number of folds is set to ten, as long as the number of instances in the training set is not smaller ten. If this is the case, the number of folds is set equal to the number of instances.
3. EM is performed ten (10) times using the ten (10) folds.
4. The logarithm of the likelihood is averaged over all ten (10) results. If logarithm of the likelihood increases, the number of clusters is increased by one (1) and the algorithm resumes from step 2.

Table 1: Datasets.

Datasets	***Size***	***# Class***	***Class distribution***
Thyroid	720	3	17, 37, 666
Lymphography	148	4	2, 81, 61, 4
Pageblocks	548	5	492, 33, 8, 12, 3
Dermatology	366	6	112, 61, 72, 49, 52, 20
Autos	159	6	3, 20, 48, 46, 29, 13
Ecoli	336	8	143, 77, 52, 35, 20, 5, 2, 2
Wine Quality	6497	7	30, 216, 2138, 2836, 1079, 193, 5
Yeast	1484	10	244, 429, 463, 44, 51, 163, 35, 30, 20, 5

The G-mean, F-measure and AUC (Area Under the Curve) measures were used for evaluating the results. For imbalanced datasets, the G-mean measure has been found to be highly representative of an algorithm's performance (Sobhani et. al., 2014). We compared our SCUT method with three other techniques, namely the original SMOTE algorithm, Random Undersampling (RU) and an implementation called CUT, which uses SCUT without using SMOTE. We also consider the scenario in which no sampling is performed, which is denoted by Original, in order to determine whether any form of sampling is actually beneficial or not.

3.2 Results

In this section, we discuss the results we have obtained against the eight datasets. Tables 2, 3, 4 and 5 show the G-mean values when the J48, Naïve Bayes, SMO and IBk classifiers are used, respectively. Tables 6, 7, 8 and 9 display the results of the F-measure values, while Tables 10, 11, 12 and 13 depict the AUC values. The tables indicate that the SCUT and SMOTE algorithms consistently produced the best results, in terms of the three measures, when applied to these benchmarking datasets.

Table 2: G-mean values for J48.

Datasets	*SCUT*	*SMOTE*	*CUT*	*RU*	*Orig.*
Thyroid	0.980	**0.989**	0.968	0.986	0.984
Lympho.	**0.936**	0.909	0.817	0.804	0.822
Pageblock	**0.975**	**0.975**	0.873	0.910	0.859
Derma.	0.977	0.971	0.976	**0.980**	0.962
Autos	0.763	**0.893**	0.797	0.818	0.853
Ecoli	0.907	**0.921**	0.862	0.864	0.899
Wine	**0.800**	0.762	0.698	0.728	0.682
Yeast	**0.828**	0.766	0.763	0.749	0.691

Table 3: G-mean values for Naïve Bayes.

Datasets	*SCUT*	*SMOTE*	*CUT*	*RU*	*Orig.*
Thyroid	**0.841**	0.762	0.696	0.734	0.667
Lympho.	**0.946**	0.924	0.859	0.906	0.837
Pageblock	0.920	**0.928**	0.850	0.898	0.848
Derma.	**0.988**	**0.988**	0.982	0.980	0.984
Autos	**0.814**	0.797	0.737	0.687	0.734
Ecoli	0.936	**0.941**	0.860	0.910	0.907
Wine	**0.639**	0.591	0.565	0.583	0.574
Yeast	**0.753**	0.724	0.700	0.711	0.705

Table 4: G-mean values for SMO.

Datasets	*SCUT*	*SMOTE*	*CUT*	*RU*	*Orig.*
Thyroid	**0.849**	0.740	0.404	0.419	0.292
Lympho.	**0.954**	0.926	0.872	0.897	0.904
Pageblocks	**0.952**	0.948	0.737	0.763	0.599
Derma.	**0.982**	**0.982**	0.970	0.970	0.972
Autos	**0.874**	0.862	0.803	0.787	0.811
Ecoli	0.904	**0.919**	0.898	0.909	0.894
Wine	**0.675**	0.620	0.633	0.632	0.601
Yeast	**0.762**	0.727	0.710	0.730	0.692

Table 5: G-mean values for IBk.

Datasets	*SCUT*	*SMOTE*	*CUT*	*RU*	*Orig.*
Thyroid	0.876	**0.883**	0.501	0.512	0.318
Lympho.	**0.946**	0.929	0.802	0.808	0.844
Pageblocks	0.951	**0.969**	0.794	0.809	0.750
Derma	0.978	**0.979**	0.978	0.978	0.974
Autos	0.818	**0.824**	0.718	0.744	0.766
Ecoli	0.920	**0.933**	0.860	0.898	0.910
Wine	**0.792**	0.745	0.657	0.649	0.645
Yeast	**0.835**	0.774	0.738	0.728	0.667

Table 6: F-measure values for J48.

Datasets	*SCUT*	*SMOTE*	*CUT*	*RU*	*Orig.*
Thyroid	0.974	**0.987**	0.970	0.977	0.986
Lympho.	**0.905**	0.875	0.787	0.786	0.804
Pageblocks	0.960	**0.967**	0.873	0.897	0.947
Derma.	0.962	0.953	0.960	**0.967**	0.940
Autos	0.792	**0.828**	0.688	0.722	0.778
Ecoli	0.841	**0.870**	0.777	0.780	0.836
Wine	**0.676**	0.640	0.568	0.605	0.586
Yeast	**0.708**	0.631	0.616	0.595	0.552

Table 7: F-measure values for Naïve Bayes.

Datasets	*SCUT*	*SMOTE*	*CUT*	*RU*	*Orig.*
Thyroid	0.779	0.650	0.765	**0.841**	0.940
Lympho.	**0.918**	0.897	0.832	0.898	0.835
Pageblk	0.871	**0.919**	0.844	0.882	0.930
Derma.	**0.981**	0.979	0.971	0.968	0.973
Autos	**0.692**	0.674	0.588	0.523	0.602
Ecoli	0.887	**0.902**	0.769	0.852	0.854
Wine	0.417	0.383	0.374	0.409	**0.439**
Yeast	**0.595**	0.573	0.521	0.542	0.566

Table 8: F-measure values for SMO.

Datasets	*SCUT*	*SMOTE*	*CUT*	*RU*	*Orig.*
Thyroid	**0.781**	0.693	0.742	0.746	0.892
Lympho.	**0.933**	0.903	0.861	0.903	0.905
Pageblocks	0.923	**0.938**	0.731	0.759	0.897
Derma.	**0.970**	0.969	0.950	0.950	0.954
Autos	**0.801**	0.785	0.695	0.671	0.718
Ecoli	0.834	**0.863**	0.829	0.845	0.823
Wine	**0.480**	0.403	0.448	0.448	0.461
Yeast	**0.606**	0.556	0.537	0.571	0.550

Table 9: F-measure values for IBk.

Datasets	*SCUT*	*SMOTE*	*CUT*	*RU*	*Orig.*
Thyroid	0.831	**0.864**	0.754	0.777	0.893
Lympho.	**0.919**	0.907	0.756	0.817	0.827
Pageblocks	0.943	**0.959**	0.817	0.827	0.931
Derma.	0.964	**0.965**	0.964	0.964	0.957
Autos	0.702	**0.721**	0.578	0.605	0.655
Ecoli	0.859	**0.885**	0.767	0.829	0.853
Wine	**0.658**	0.612	0.509	0.497	0.543
Yeast	**0.713**	0.639	0.576	0.568	0.522

Table 10: AUC values for J48.

Datasets	*SCUT*	*SMOTE*	*CUT*	*RU*	*Orig.*
Thyroid	0.982	**0.993**	0.968	0.981	0.885
Lympho.	**0.943**	0.908	0.809	0.815	0.828
Pageblocks	0.977	**0.981**	0.854	0.923	0.845
Derma	0.984	**0.985**	0.980	**0.985**	0.977
Autos	0.903	**0.932**	0.869	0.872	0.894
Ecoli	0.938	**0.941**	0.874	0.882	0.920
Wine	**0.834**	0.803	0.744	0.769	0.722
Yeast	**0.881**	0.817	0.806	0.779	0.733

Table 11: AUC values for Naïve Bayes.

Datasets	*SCUT*	*SMOTE*	*CUT*	*RU*	*Orig.*
Thyroid	0.916	**0.932**	0.819	0.830	0.872
Lympho.	**0.982**	0.961	0.930	0.947	0.920
Pageblocks	**0.982**	0.981	0.893	0.934	0.916
Derma	0.999	0.999	0.999	0.999	0.999
Autos	**0.910**	0.899	0.832	0.811	0.828
Ecoli	0.974	**0.979**	0.928	0.945	0.960
Wine	**0.801**	0.748	0.684	0.692	0.658
Yeast	**0.912**	0.874	0.858	0.848	0.816

Tables 14, 15 and 16 show the summaries for the G-mean, F-measure and AUC values, respectively, when the results are ranked. For each dataset, the five methods are ranked from 1 to 5. Here, 1 corresponds to the highest rank (for highest value) while 5 denotes the lowest rank. If there is a tie, then

Table 12: AUC values for SMO.

Datasets	*SCUT*	*SMOTE*	*CUT*	*RU*	*Orig.*
Thyroid	**0.892**	0.770	0.522	0.536	0.512
Lympho.	**0.971**	0.937	0.895	0.904	0.907
Pageblocks	**0.973**	0.964	0.792	0.818	0.673
Derma.	**0.989**	**0.989**	0.980	0.980	0.984
Autos	**0.927**	0.925	0.899	0.866	0.896
Ecoli	0.963	**0.970**	0.938	0.949	0.944
Wine	**0.808**	0.759	0.706	0.705	0.678
Yeast	**0.887**	0.846	0.823	0.828	0.781

Table 13: AUC values for IBk.

Datasets	*SCUT*	*SMOTE*	*CUT*	*RU*	*Orig.*
Thyroid	0.939	**0.948**	0.631	0.673	0.591
Lympho.	**0.986**	0.965	0.932	0.957	0.923
Pageblocks	0.981	**0.986**	0.885	0.933	0.925
Derma.	**0.998**	**0.998**	0.997	0.995	0.997
Autos	0.929	**0.933**	0.873	0.876	0.903
Ecoli	0.958	**0.965**	0.913	0.939	0.951
Wine	**0.898**	0.857	0.764	0.769	0.745
Yeast	**0.927**	0.897	0.843	0.840	0.685

while 5 denotes the lowest rank. If there is a tie, then the same rank is assigned to both. For each method, all the ranks are added, and the method with the smallest rank sum is assigned the highest rank (1) while the method with the largest rank sum is attributed the lowest rank (5).

Table 14: Ranks for different methods for G-mean values.

Classifier	*SCUT*	*SMOTE*	*CUT*	*RU*	*Orig.*
J48	2	**1**	5	3	4
NaiveBayes	**1**	2	4	3	4
SMO	**1**	2	4	3	5
IBk	2	**1**	4	3	5

Table 15: Ranks for different methods for F-measure.

Classifier	*SCUT*	*SMOTE*	*CUT*	*RU*	*Orig.*
J48	2	**1**	4	3	3
NaiveBayes	**1**	3	5	4	2
SMO	**1**	2	5	4	3
IBk	2	**1**	5	4	3

Table 16: Ranks for different methods for AUC values.

Classifier	*SCUT*	*SMOTE*	*CUT*	*RU*	*Orig.*
J48	2	**1**	5	3	4
NaiveBayes	**1**	2	4	3	4
SMO	**1**	2	4	3	5
IBk	2	**1**	4	3	4

For the Naïve Bayes and SMO classifiers, our SCUT method obtains the highest overall rank, while SMOTE ranks second or third. On the other hand, when using the J48 and IBk classifiers, SMOTE achieves the highest rank while SCUT comes second. The RU technique scores third, in terms of the G-means and AUC values, while the Original and CUT methods rank either fourth or fifth. For the F-measure, the Original dataset outperforms the undersampling techniques. This result suggests that the undersampling-only techniques do not work well in multi-class settings.

The results from Tables 14, 15 and 16 indicate that our SCUT method is most suitable in scenario where the Naïve Bayes and SMO classifiers are employed. When the J48 and IBk classifiers are used, SMOTE on its own provides the best results against the datasets under consideration. Furthermore, when the datasets are highly imbalanced with some classes having very few instances, CUT and RU consistently perform poorly. This observation is also true for the Original dataset in which sampling is absent. This confirms our initial hypothesis that oversampling is required in order to improve the performance in such a multi-class scenario.

Subsequently, the Friedman statistical test was used in order to assert the statistical significance of the results. The Friedman test is a non-parametric statistical test. Since it ranks the values in each row, it is not affected by factors that equally affect all the values in a row. In addition, unlike other tests such as ANOVA and paired t-test, it does not make any assumptions about the data distribution. The results for each classifier (J48, Naïve Bayes, SMO and IBk) on each evaluation metric (G-mean, F-measure and AUC) are depicted in Tables 2 to 13. The resultant *p*-values are shown in Table 17.

Table 17: Summary of p-values for classifiers and evaluation metrics.

Evaluation metric	**Classifiers**			
	J48	***Naïve Bayes***	***SMO***	***IBk***
G-mean	0.01460	0.00038	0.00060	0.00010
F-measure	0.00221	0.00470	0.00642	0.00029
AUC	0.00011	0.00024	0.00004	0.00005

Assuming that the results are statistically significant if $p < 0.05$, it may be concluded that all our results are valid and statistically significant.

3.3 Discussion

Our experimental results, based on the benchmarking multi-class imbalanced datasets, show that when the SCUT method is used, improved results for the Naïve Bayes and Support Vector Machine classifiers are obtained. This suggests that undersampling makes these two classifiers more sensitive to the minority classes, and aids them to correctly classify minority class instances. On the other hand, the SMOTE technique produces the highest measured values for the J48 and K-Nearest Neighbour classifiers. Thus, undersampling does not improve the results for these two algorithms. Rather, relying solely on oversampling the minority classes is sufficient in this particular case. In general, our results indicate that oversampling the majority instances is crucial in order to address multi-class problems. Undersampling approaches, such as RU and CUT, do not perform well in these conditions. As a matter of fact, using the Original dataset without any form of undersampling sometimes outperforms these two techniques.

The reader should notice that the SCUT algorithm produced the best overall results in the Yeast dataset, which contains the highest number of classes with a wide range or cardinality. It also produced the best results for the Wine Quality dataset (in all cases except Table 7). Recall that this is the largest dataset with a high degree of imbalance. In addition, SCUT produced the most accurate models against the Lymphography dataset, where the levels of imbalance are quite high. This seems to indicate that our SCUT method is particularly suitable in such cases, since undersampling is required. That is, this scenario implies that a classifier may benefit from increased sensitivity to the minority classes. Indeed, consider a dataset in which there are majority classes that contain very large numbers of instances. In this context, solely relying on the oversampling of the minority classes may result in overfitting and noise. Once more, the SCUT method provides a remedy to such a situation. We plan to further investigate this aspect in the near future.

4 CONCLUSIONS

In this paper, we have proposed a hybrid sampling method called SCUT which combines SMOTE and cluster-based undersampling to improve the classification performance on multi-class imbalanced datasets. Cluster-based undersampling

handles within-class imbalance. Further, the combination of cluster-based undersampling and SMOTE aids to reduce between-class imbalance, without excessive use of sampling.

We were not able to establish a clear superiority of one oversampling method over the other. However, we were able to determine that the SCUT method is a promising candidate for further experimentation. Our results suggest that our SCUT algorithm is suitable for domains where the number of classes is high and the levels of examples vary considerably. We intend to further investigate this issue. We also intend extending our approach to very large datasets with extreme levels of imbalances, since our early results indicate that our SCUT approach would potentially outperform undersampling-only techniques in such a setting. In our paper, the number of instances for each class was set to the mean value. Exploring the optimal strategy for fixing the number of instances will be further explored, e.g. by sampling the instances directly from the distribution associated with the mixture of Gaussians as obtained from the EM algorithm.

Cost-sensitive learning is another common approach for dealing with the class-imbalance problem. Most of the existing solutions are applicable to binary-class problems, and cannot be applied directly to multi-class imbalanced datasets (Sun et al., 2006). Rescaling, which is a popular cost-sensitive learning approach for binary class problems can be applied directly on multi-class datasets to obtain good performance only when the costs are consistent (Zhou and Liu, 2010). In addition, rescaling classes based on cost information may not be suitable for highly imbalanced datasets. Designing a multi-class cost-sensitive learning approach for inconsistent costs without transforming the problem into a binary-class problem will be the focus of our future work.

REFERENCES

Chawla, N., Bowyer, K., Hall, L., Kegelmeyer, W., 2002. SMOTE: Synthetic minority over-sampling technique. In *Journal of Artificial Intelligence Research*, Volume 16, pages 321-357.

Alcalá-Fdez, J., Fernandez, A., Luengo, J., Derrac, J., García, S., Sánchez, L., Herrera, F., 2011. KEEL data-mining software tool: data set repository, integration of algorithms and experimental analysis framework. In *Journal of Multiple-Valued Logic and Soft Computing*, pages 255-287.

Cortez, P., Cerdeira, A., Almeida, F., Matos, T., Reis, J, 2009. Modelling wine preferences by data mining from physicochemical properties. In *Decision Support Systems*, Elsevier, Volume 47, number 4, pages 547-553.

Dempster, A. P., Laird, N.M., Rubin, D.B., 1977. Maximum likelihood from incomplete data via the EM algorithm. In *Journal of the Royal Statistical Society*, Volume 39, Number 1, pages 1-38.

Fernández, A., Jesus, M., Herrera, F., 2010. Multi-class imbalanced data-sets with linguistic fuzzy rule based classification systems based on pairwise learning. In *Computational Intelligence for Knowledge-Based System Design*, Volume 6178, Number 20, pages 89-98.

Han, H., Wang, W., Mao, B., 2005. Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning. In *Proceedings of International Conference on Advances in Intelligent Computing*, Springer, Volume Part I, pages 878-887.

Japkowicz, N., 2001. Concept-learning in the presence of between-class and within-class imbalances. In *AI 2001: Lecture Notes in Artificial Intelligence*, Volume 2056, Springer, pages 67-77.

Lichman, M., 2013. UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

Rahman, M. M., Davis, D. N., 2013. Addressing the class imbalance problem in medical datasets, In *International Journal of Machine Learning and Computing*, Volume 3, Number 2, pages 224-228.

Ramanan, A., Suppharangsan, S., Niranjan, M., 2007. Unbalanced decision trees for multi-class classification. In *ICIIS 2007: IEEE International Conference on Industrial and Information Systems*, IEEE Press, pages 291-294.

Sobhani, P., Viktor, H., Matwin, S., 2014. Learning from imbalanced data using ensemble methods and cluster-based undersampling. In *PKDD nfMCD 2013: Lecture Notes in Computer Science*, Volume 8983, Springer, pages 38-49.

Sun, Y., Kamel, M., Wang, Y., 2006. Boosting for learning multiple classes with imbalanced class distribution. In *IEEE ICDM '06: Proceedings of the Sixth International Conference on Data Mining*, IEEE Press, pages 592-602.

Viktor, H.L., Paquet, E. and Zhao, J., 2013. Artificial neural networks for predicting 3D protein structures from amino acid sequences, In *IEEE IJCNN: International Joint Conference on Neural Networks*, IEEE Press, pages 1790-1797.

Wang, S., Yao, X., 2012. Multi-class imbalance problems: analysis and potential solutions. In *IEEE Transactions on Systems, Man, and Cybernetics,* Part B, Number 4, pages 1119-1130.

Yen, S. J., Lee, Y. S., 2009. Cluster-based under-sampling approaches for imbalanced data distributions. In *Expert Systems with Applications*, Volume 36, Number 3, pages 5718-5727.

Zhou, Z., Liu, X., 2010. On multi-class cost sensitive learning. In *Computational Intelligence*, Volume 26, No 3, pages. 232-257.