Hierarchical Classification for Dealing with The Class Imbalance Problem

Mohamed Bader-El-Den
School of Computing
University of Portsmouth
Portsmouth PO1 3HE, UK
Email: mohamed.bader@port.ac.uk

Eleman Teitei
School of Computing
University of Portsmouth
Portsmouth PO1 3HE, UK
Email: eleman.teitei@port.ac.uk

Mo Adda
School of Computing
University of Portsmouth
Portsmouth PO1 3HE, UK
Email: Mo.Adda@port.ac.uk

***Abstract*—The aim of classification in machine learning is to utilize knowledge gained from applying learning algorithms on a given data so as determine what class an unlabelled data having same pattern belongs to. However, algorithms do not learn properly when a massive difference in size between data classes exist. This classification problem exists in many real world application domains and has been a popular area of focus by machine learning and data mining researchers. The class imbalance problem is further made complex with the presence of associative data difficult factors. The duo have proven to greatly deteriorate classification performance. This paper introduces a two-phased data level approach for binary classes which entails the temporary re-labelling of classes. The proposed approach takes advantage of the local neighbourhood of the minority instances to identify and treat difficult examples belonging to both classes. Its outcome was satisfactory when compared against various data-level methods using datasets extracted from KEEL and UCI datasets repository.**

I. Introduction

Over the years, machine learning and data mining have become one of the most pronounced areas in the field of computer science. An important area frequently considered is supervised learning where classifiers deduce patterns from a known labelled data so as to build a model that could make predictions for an unknown labelled data having similar characteristics. However, the issue of classifiers learning accurately still presents an enormous challenge in real world domains where such is applicable. The problem is further accentuated in a class imbalance scenario where a class (minority) contains smaller number of instances when compared to instances belonging to another class (majority) [5]. Imbalance between classes cause algorithms to be biased towards the majority class during learning which results to the deterioration in classification performance. The Class imbalance problem can be found in many real-life application domains which include but not limited to hardware fault detection [31], medical diagnosis [27], fraud detection [32], image annotation [24], anomaly detection [21], oil spillage [1].

Many methods generally categorized under data, algorithmic and hybrid level have been introduced to tackle class imbalance. Notwithstanding, effectively identifying scenarios for their usage still poses a problem. This lapses is closely associated to the issue of fully comprehending the underlying key properties that leads to poor classification performance in an imbalanced data distribution. It can be deduced from other reviews that most authors concentrate more on comparing their methods on datasets while accentuating its performance over other methods [5], [39]. This approach does not dig deep into the underlying data characteristics. The used datasets are normally described with reference to their imbalanced global ratio or the positive class size. This yardstick for measurement does not adequately explain the difference between the compared methods in regards to classification performance. For example, some classifiers can still adequately recognize the minority class in a highly imbalanced dataset [14], [28], [37], [33].

Adding to the complexity of classifiers learning accurately are the underlying data difficult factors embedded in the targeted datasets. These factors have shown to contribute more to the deterioration in classification performance than the class imbalance itself [11]. Irrespective of the applied method, it has been discovered that an approach can effectively improve classification performance in a class imbalance scenario in which a particular data difficult factor is more accentuated. But is less effective when faced with a different factor; both datasets having same imbalanced ratio [26].

This paper proposes a two-phased data-level approach named TempC, which introduces the temporary re-labelling of classes aimed at reducing the level of class imbalance and also to identify and treat difficult areas of a dataset separately. In the original training set , the minority examples and their k nearest majority neighbours are given a new class label which serves as the new minority class. A classification model is built which is used to classify unlabelled instances in the test set. The derived minority class which captures the difficult areas is used as the training set in the $2nd$ phase. Another classification model is built and the test instances that were predicted as belonging to the minority class in the $1st$ phase are used as test set. The proposed approach is more robust to the different data difficulty factors associated with class imbalance.

The rest of the paper is structured as follows: Section 2 reviews existing various data-level approaches and other methods related to the proposed approach. Section 3 describes the proposed approach. Section 4 describes the datasets used in carrying out experiments. Section 5 describes the setup and analysis of the undertaken experiments while section 6 summarizes the paper.

II. Literature Review

As stated in section 1, methods employed to tackle class imbalance are generally classified into:

- **Data level:** aims at balancing the data distribution by adding artificial instances to the minority class (oversampling) or removing some of the majority class instances (undersampling) [44].
- **Algorithmic Level:** internally modifies the algorithm used for classification [28]. Algorithmic methods fall under categories such as: one class learning [19] , cost-sensitive learning [31] , and changing the internal bias [4].
- **Hybrid level:** systematically combines both data and algorithmic level methods.

We divert our focus to data-level methods as our motivation emanates from the identified drawbacks deduced from the oversampling, undersampling and rebelling of data whereby the original dataset is in most cases, permanently altered. Furthermore, data-level approaches have an advantage over other methods as it occurs during pre-processing and is not dependent on any succeeding learning algorithm and can be easily combined with other methods.

The class imbalance problem can easily be tackled by randomly duplicating or generating synthetic minority class instances (random oversampling) or randomly removing majority class instances (random undersampling); both aiming at balancing the class distribution. There individual drawbacks are that randomly oversampling instances may lead to overfitting while valued data might be lost during undersampling [44]. For these reasons, researchers have come up with more focused ways of re-sampling data. The underlying idea of these methods involves the analysis of the local neighbourhood of instances using estimated distance measurement so as to identify and process perceived redundant regions in a given dataset.

A. Focused Undersampling

Such focused undersampling methods include ENN (Edited Nearest Neighbour) which removes only instances in the majority class that their two or three nearest neighbours belongs to the opposite class [43]. Laurikkala [23], introduced N NCR tends to remove more instances than ENN. Furthermore, CNN (Condensed Nearest Neighbour) keeps only those majority class instances that were incorrectly classified by k nearest neighbours [16]. The logic behind CNN is that majority class examples that are located in safe areas and far from the decision boundaries are not important for learning. One-sided selection method (OSS) [22] is another effective undersampling method that combines CNN with Tomek Links [38]. Although removing instances have proven to reduce the imbalance ratio and also improve classification performance, permanently discarding instances might lead to losing important data.

B. Focused Oversampling

A popular focused oversampling technique is SMOTE (Synthetic Minority Over-sampling Technique) which generates new synthetic minority instances created from a chosen point of the line that links the selected instance to its nearest minority neighbour [9]. Its main drawback is that the number of generated examples is fixed in advance thereby restricting flexibility in the re-balancing rate. Furthermore, it does not put into consideration neighbouring instances belonging to the majority class which might increase the overlapping areas. Various researchers have attempted to modify SMOTE so as to correct its drawback by introducing more focused oversampling approaches where only a specific fraction of the minority examples are oversampled. Such focused oversampling methods include Borderline-SMOTE where only minority instances found around the border area are oversampled [15]. Other similar approaches are Safe-level , SMOTE [25] and LN-SMOTE [8] where new instances are created from the analysis of the nearest neighbours of safe instances within the local safe areas. Another concept introduced to handle the limitations of SMOTE is combining it with a post-processing technique such as ENNR or Tomek links [5] i.e discarding the perceived dangerous samples after applying SMOTE. This has proven to be effective when combined with decision tree and rule based classifiers.

C. Evaluating Performance for Class Imbalanced Scenario

Evaluating classifier performance shows how well a method improves classification. One good approach is to measure the classification accuracy which can be deduced from the confusion matrix which depicts the performance of classifiers on a test set where the true values are known. For a binary dataset, a 2x2 matrix shows the number of correctly classified minority instances (true positives), incorrectly classified minority instances (false positives), correctly classified majority instances (true negatives) and incorrectly classified majority instances (false negatives). The classification accuracy analyses the success of learning classifiers by the percentage of the correct predictions made. While accuracy is suitable for a balanced data distribution, it is strongly biased in a class imbalance scenario as it tends to give a high percentage accuracy even when all the minority class instances are incorrectly classified. The classification accuracy is defined as:

$$ACC = \frac{TP + TN}{P + N} \tag{1}$$

For this reason, better evaluative measures that are independent of the class imbalance ratio and sufficiently recognizes the minority class are preferred. One of such measures used in this paper is the Geometric Mean which calculates the geometric mean between the sensitivity (true positive rate) and specificity(true negative rate) [22].

$$Sensitivity = \frac{TP}{TP + FN} \tag{2}$$

$$Specificity = \frac{TN}{TN + FP} \tag{3}$$

The G-mean is an effective evaluative criterion as it is not dependent on the data distribution. The G-mean is defined as:

$$G - mean = \sqrt{Sensitivity.Specificity} \tag{4}$$

It accentuates the balancing between the specificity and sensitivity while maximizing the recognition between the minority and the majority class. Other evaluative measures used in the paper are Precision and Fmeasure with formulas as follows:

$$Precision = \frac{TP}{TP + FN} \tag{5}$$

$$F - Measure = \frac{2 * Recall * Precision}{Recall + Precision} \tag{6}$$

D. Related Works

An approach which also re-labels instances is the SPIDER(Selective Pre-processing for Imbalanced Data) method [37]. It systematically removes or re-labels majority class examples while difficult instances from the minority class are amplified. It differentiates instances by applying nearest neighbour rule (NNR) with the heterogeneous value distance metric (HVDM). SPIDER and NCR methods have shown to be more successful than some oversampling methods when applied to imbalanced datasets where the subset of various difficulties in the data is more pronounced. However NCR performed much better than SPIDER in datasets where borderline examples are more accentuated. On the other hand, SPIDER, as well as SMOTE proved to be more reliable, when experimented on outlier and rare instance accentuated datasets. The SPIDER method has however been criticized of modifying too many majority class instances thereby greatly reducing specificity at the cost of improving sensitivity. Hence, SPIDER2 [30] was introduced to reduce its drawbacks. Although SPIDER has proven to improve classification performance, permanently re-labelling examples might not be acceptable in some domains.

Another closely related concept to TempC is the multi-level classification model introduced in [17] and later enhanced in [18]. The concept was introduced while mining user behaviors and environments for semantic place prediction. The original classification problem is broken down into sub-classification problems. For instance, given a dataset having six raw data where the first three belong to class U and the remaining three belong to classes X,Y,Z respectively. A classification model is built which classifies data into U and $NOT\ U$. Another classification model is built which classifies data under $NOT\ U$ into X,Y,Z. A similar process is repeated during testing. The test data is first classified into U and $NOT\ U$. Test data that are not classified as U will further be classified into X,Y,Z by the low-level model. In summary, the original classification problem is divided into sub-classification thereby reducing the level of class imbalance in the data. While it maintains the original data, its concept is designed for multi-classes where the characteristic of the class labels are hardly distinguishable therefore not suitable for binary class problems.

E. Data Difficult Factors

Data difficulty factors when associated with class imbalance have proven to further degrade classification performance [11]. These data difficulty factors include:

- **Small Disjuncts:** A small cluster of instances belonging to a particular class positioned in an area dominated by the opposite class [41], [40].
- **Class Overlap:** It occurs when there are ambiguous regions in the data where the prior probability for both classes are approximately equal [13], [11].
- **Noisy Instances:** Are single instances found within the area dominated by the opposite class [7], [35].
- **Rare and Outlying Instances:** These instances have almost similar characteristics as the noisy instances. While noise emanates from labelling error, rare instances are just untypical examples while outliers are also located within the majority class which are far from the decision boundaries of both classes [29].
- **Borderline Instances:** They are located in the areas surrounding class boundaries where the minority and majority class overlap [12], [30].

III. The Proposed Approach

This section describes the proposed method, TempC, which originates from the hypotheses on the role of the mutual positions of learning examples in the attribute space. And the idea of assessing the minority examples and depicting difficult areas by analysing class labels of the opposite examples in its local neighbourhood.

TempC is aimed at reducing the level of imbalanced ratio between the minority and the majority class. It also identifies and treat difficult areas separately in the training set so as to enable learning algorithms sufficiently recognise the minority class. This is done by systematically extracting some of the majority instances and integrating them into the minority class. For example, given a class imbalance dataset, it is first divided into training and testing set. The k nearest majority neighbours of all the minority class instances are identified by employing the heterogeneous value distance metric (HVDM); aggregating the value distance metrics for qualitative features and euclidean distance metric for numerical attributes [42]. All minority instances and their identified nearest majority neighbours are given a new class label which in turn, replaces the original minority class thereby increasing the size of the minority class while reducing the majority class. A classification model is built from the newly derived dataset which is used to classify the test set. Another advantage is that since it is difficult for classifiers to learn from the minority instances that are located in areas dominated by the majority class, giving such instances depicted in those areas same label reduces the learning difficulties encountered by classifiers.

In the second phase, the derived minority class which captures the difficult areas of the input space is used as a training dataset as it is decomposed back into minority instances and their k nearest majority neighbours with their original class labels. The difference in size between both classes is reduced as only a subset of the original majority class is used. Another classification model is built.

Instances in the unlabelled testing set are first tested on the $1st$ level. If the instances are classified as minorities, the 2nd level classification model is used to confirm that the testing instance actually belongs to the minority class.

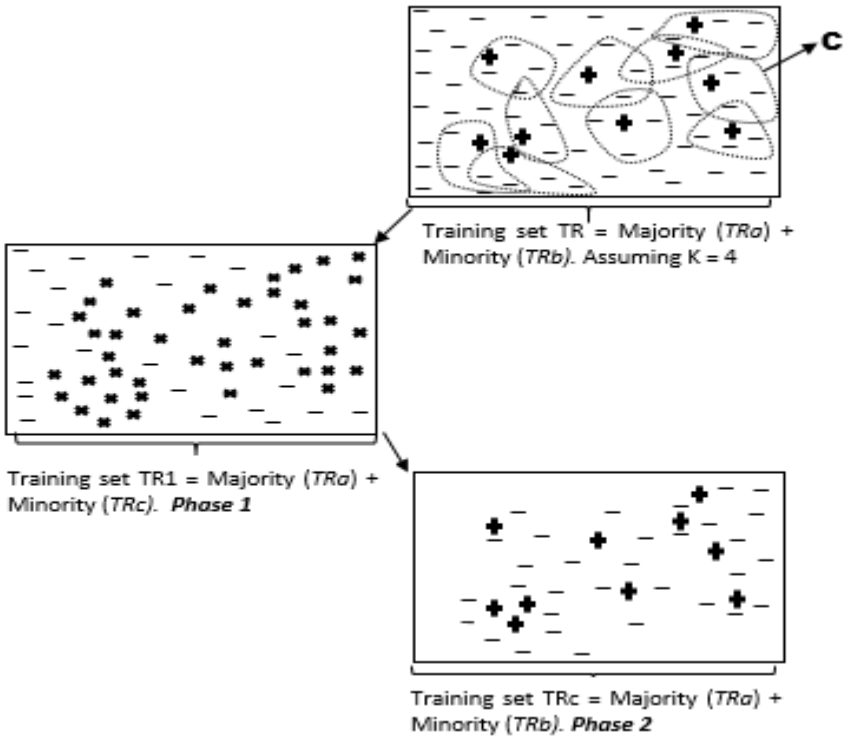

Fig. 1: illustrates the derived training sets used in TempC assuming $k = 4$

A. Algorithm Description

Algorithm 1 Two level

Split the dataset into 'training' TR and 'testing' TS sets
Split TR into majority TR_a and minority TR_b sets
for $i = 1$ to TR_b **do**
  find TR_{bi} n nearest neighbours from TR_a
  Relabel the class of TR_{bi} and its neighbours to C
  Copy TR_{bi} and the n nearest neighbours to TR_c
end for
Build level 1 classifier using $TR_a \cup TR_c$
Build level 2 classifier using TR_c

As shown in $Algorithm1$: Given a binary imbalanced dataset divided into training set TR and test set TS, a new label C is given to the minority class TR_b and their K-nearest majority neighbours TR_a in training set TR. The newly labelled TR_c is used to replace the original minority class TR_b. A classifier is learned on the new training set having TR_a and TR_c as majority and minority class respectively. TS is tested on the built classification model. In the 2nd phase, TR_c is decomposed into minority class TR_b and their k-nearest majority neighbours TR_a. Another classification model is built from TR_a and TR_b. All TS instances that where predicted as C in the 1st phase are used as TS in this phase.

Unlike many data-level approaches, TempC does not add synthetic minority instances. Neither does it permanently relabel nor discard majority instances thereby eliminating the problem of overfitting, removal of useful data and also the permanent alteration of class labels. Furthermore, both phases tend to reduce the imbalanced ratio and also reduce the level of classification difficulty thereby improving the chances of classifiers sufficiently recognizing the minority class. Secondly, TempC is more robust in the sense that it is independent of whichever way the minority instances are positioned around the majority instances owing to the fact that all minority instances and their K nearest neighbour are giving same label irrespective of there attributed difficulty factor. Furthermore, the possible application of different learning algorithms in the different phases makes the method more flexible.

IV. DATASETS

Experiments were carried out on datasets that are embedded with several data difficulty factors occurring together. Experiments relating to data difficulty factors in a class imbalance scenario have mostly been carried out using artificially generated datasets so as to be able to control their parameters. We implement our approach on seventeen class imbalanced datasets extracted from KEEL [2] and UCI [3] dataset repositories. Such datasets as briefly described in $Table1$, have been experimented with in [30], [36] and [37]. The chosen datasets present significant challenges for standard classifiers.

The KEEL datasets all have 800 examples with an imbalance ratio of 1:7. Their majority instances were uniformly distributed around the minority instances taking the shapes of paw, subclus and clover respectively. The minority instances in the paw dataset were broken down into three elliptic subregions with two of its regions positioned near each other. The minority instances in the clover dataset are arranged in a pattern that depicts a flower with elliptic petals which make them non-linear and more difficult for an algorithm to

learn from. While in the subclus dataset, minority instances were positioned in rectangular shapes which are uniformly surrounded by the majority instances. To accentuate their difficulty, the disturbance ratio of their underlying borderline and small disjunt examples were increased by 30%, 50% and 70% respectively . A detailed description of these datasets can be found in [36].

Furthermore, five real-world datasets were downloaded from the UCI datasets repository. These datasets have different degrees of imbalance ratio and contains a minute number of safe minority examples [36]. For example, the minority instances in Herbaman is made up of 51 borderline, 21 outlier and 10 safe examples.

TABLE I: Dataset Breakdown

Dataset	#Instances	#Features	# Min	#Maj	#IR
Paw	800	3	100	700	0.14
Clover	800	3	100	700	0.14
Subclus	800	3	100	700	0.14
Breast Cancer	286	9	85	201	0.30
Bupa	345	8	145	200	0.42
Haberman	306	19	81	225	0.26
Hepatiti	155	9	32	123	0.21
Pima	768	18	268	500	0.35

TABLE II: Dataset Breakdown

Dataset	#Instances	#Features	# Min	#Maj	#IR
Paw	800	3	100	700	0.14
Clover	800	3	100	700	0.14
Subclus	800	3	100	700	0.14
Abalone	4174	8	32	4142	129.44
Breast	286	9	85	201	0.30
Bupa	345	8	145	200	0.42
Car	1728	6	69	1659	24.4
Haberman	306	19	81	225	0.26
Hepatiti	155	9	32	123	0.21
Pima	768	18	268	500	0.35
Poker	2075	10	25	1460	82
Yeast	1484	8	51	1433	28.1
Zoo	101	16	5	96	19.1

V. INVESTIGATIONS

A. Experimental Description and Setup

Two experiments were conducted in this paper. The aims of the first experiment were to ascertain which tuning of k is most suitable to sufficiently recognize the minority instances across various datasets as compared against the outcome of standard classifiers on the original datasets without any modification. The second aim was to test how robust is TempC to the different difficult class imbalance datasets irrespective of their accentuated data intrinsic characteristics. We also investigated how effective is TempC in improving the ability of different learning algorithms in classifying instances correctly. In these experiments, 80% of the datasets was used for training while the remaining 20% was used for testing. The knn employed ranged between $1 - 50$. We ran each experiment 31 $times$ with $random\ seed = 100$ using 17 datasets and 3 learning algorithms drawn from different bases; J48(C4.5) [34] and Random Forest [6] tree classifiers and JRip [10] rule-based classifier. Geometric Mean (GM), F-Measure (FM), Accuracy (Acc), Precession (Pre), Sensitivity (Sen) and Specificity (Spe) were used to evaluate the classification performance.

Due to limited space, only a sample of the results were plotted. Figure 2 shows the performance of the TempC algorithm using different number of neighbours ranging between 1 to 50. The performance is evaluated using different measures. The base algorithm is Random Forest while the portrayed dataset is Pima. Also, the box-plot diagram shows the performance of the base algorithm with SMOTE. The box-plot shows the performance of 10 independent runs, the lower whisker shows the worst result, while the upper-whisker shows the best results obtained. The box shows the lower-quartile, median and upper-quartile values respectively.

Table III compares the performance of the proposed TempC (with and without SMOTE) against the base classifier (with and without SMOTE). The results are the average of 31 runs $\pm$ the standard deviation. J48(C4.5) [34], Random Forest [6] and JRip [10] (rule based) were used as base algorithms in this experiment. The number of neighbours in all the experiments is 40. The best overall results for each dataset is marked by (*). As shown in the table, the TempC based on RF has outperformed all the other algorithms on more than 95% of the datasets. Also, the results indicate that TempC tends to always improve the RF algorithm. However, this is not the case with JRip. But JRip performance was selected as it is the best rule based algorithm we have tested.

The second experiment was aimed at comparing TempC's performance against other data level approaches that have proven to improve classification performance in difficult class imbalance scenarios [26]. Such methods include random oversampling, Japkowiczs cluster oversampling [20] NCR [23], SMOTE [9], SPIDER [37] and SPIDER2 [30]. We focused on the ability of the various methods to sufficiently recognize the minority instances as well as the majority instances. We compared our results against results extracted from [30] and [36] using C4.5 as a learning classifier. C4.5 was run unpruned using 10 fold cross-validation so as to sufficiently describe the minority class and also to be inline with the experimental setup of the compared results.

It is very important to state at this point that we have been unable to implement the above compared algorithms and our results are compared with what have been reported in other literature using the same datasets and experimental parameters . Therefore, it cannot be guaranteed that the comparison in Table IV is accurate. The aim of this rough comparison is just to give an overall indication of the TempC algorithm. Table IV shows the overall best results for TempC (The average performance of TempC using $n = 40$ are shown in Table III). As shown in table IV, increasing the disturbance ratio of paw, clover and paw datasets from 0%, 30%, 50% and 70% greatly deteriorates the performance of the base classifiers with or without any pre-processing applied on the datasets. Notwithstanding, using Gmean as an evaluative criteria, $TempC$ performed much better than the other methods when applied on $Paw0$, $paw50$, $clover0$, $clover30$, $subclus0$, $subclus30$ and $subclus50$. On the other-hand, NCR performed better on $Paw30$, SP2 on $Paw70$, $Clover50$, $Clover70$ and $Subclus70$. All methods performed better than the baseline classifier. We therefore conclude that pre-processing datasets before classification greatly improves classification performance.

TABLE III: Compares the performance of the proposed TempC with $n = 40$ (with and without SMOTE) against the base classifier (with and without SMOTE). The results are the average of 31 runs $\pm$ the standard deviation. The best overall results for each dataset is marked by (*)

		Base		Base+SMOTE		TempC		TempC+SMOTE	
	Name	fm	gm	fm	gm	fm	gm	fm	gm
Random Forest	Paw0	0.906 ± 0.08	0.944 ± 0.05	0.899 ± 0.082	0.936 ± 0.049	**0.912 ± 0.075***	**0.95 ± 0.042***	0.902 ± 0.084	0.94 ± 0.051
	paw30	0.652 ± 0.15	0.82 ± 0.11	0.676 ± 0.12	0.805 ± 0.083	0.682 ± 0.148	**0.835 ± 0.08***	**0.695 ± 0.127***	0.814 ± 0.08
	Paw50	0.569 ± 0.09	0.784 ± 0.08	0.615 ± 0.053	0.752 ± 0.056	0.565 ± 0.088	**0.793 ± 0.079***	**0.618 ± 0.087***	0.757 ± 0.079
	Paw70	0.443 ± 0.17	0.735 ± 0.15	0.531 ± 0.135	0.708 ± 0.094	0.463 ± 0.099	**0.77 ± 0.06***	**0.563 ± 0.1***	0.734 ± 0.06
	Clover0	0.836 ± 0.09	0.928 ± 0.08	**0.845 ± 0.094***	**0.93 ± 0.067***	0.811 ± 0.073	0.929 ± 0.067	0.834 ± 0.081	0.928 ± 0.067
	Clover30	0.596 ± 0.09	0.641 ± 0.8	**0.686 ± 0.09***	0.786 ± 0.076	0.634 ± 0.121	**0.814 ± 0.099***	0.685 ± 0.117	0.78 ± 0.099
	Clover50	0.472 ± 0.12	**0.749 ± 0.13***	**0.564 ± 0.113***	0.721 ± 0.115	0.43 ± 0.167	0.72 ± 0.114	0.54 ± 0.095	0.714 ± 0.114
	Subc0	**0.962 ± 0.05**	**0.977 ± 0.03**	**0.962 ± 0.054**	**0.977 ± 0.031**	**0.962 ± 0.054**	**0.977 ± 0.031**	**0.962 ± 0.054**	**0.977 ± 0.031**
	Subc30	0.691 ± 0.07	0.873 ± 0.06	0.673 ± 0.103	0.824 ± 0.11	**0.696 ± 0.072***	**0.873 ± 0.108***	0.672 ± 0.095	0.83 ± 0.108
	Subc50	0.412 ± 0.14	**0.682 ± 0.13***	**0.446 ± 0.069***	0.667 ± 0.075	0.387 ± 0.134	0.656 ± 0.074	0.42 ± 0.089	0.633 ± 0.074
	Subc70	0.298 ± 0.12	0.592 ± 0.14	**0.397 ± 0.128***	0.609 ± 0.1	0.293 ± 0.125	0.596 ± 0.105	0.395 ± 0.13	**0.614 ± 0.105***
	Pima	0.212 ± 0.07	0.221 ± 0.05	0.159 ± 0.044	0.215 ± 0.033	0.659 ± 0.072	**0.746 ± 0.039***	**0.696 ± 0.059***	0.741 ± 0.039
C4.5	Paw0	0.531 ± 0.136	**0.871 ± 0.116**	**0.732 ± 0.152**	0.789 ± 0.12	0.453 ± 0.196	0.762 ± 0.289	0.711 ± 0.134	0.764 ± 0.108
	paw30	0.12 ± 0.211	0.214 ± 0.286	**0.581 ± 0.114**	0.674 ± 0.106	0.023 ± 0.049	0.098 ± 0.207	0.576 ± 0.144	**0.683 ± 0.156**
	Paw50	0.138 ± 0.2	0.268 ± 0.347	**0.57 ± 0.077**	**0.664 ± 0.069**	0.058 ± 0.184	0.071 ± 0.225	0.569 ± 0.077	0.658 ± 0.068
	Paw70	0 ± 0	0 ± 0	**0.434 ± 0.174**	**0.523 ± 0.194**	0 ± 0	0 ± 0	0.155 ± 0.209	0.206 ± 0.271
	Clover0	0.566 ± 0.125	**0.899 ± 0.058**	**0.622 ± 0.134**	0.897 ± 0.074	0.549 ± 0.135	0.89 ± 0.068	0.592 ± 0.138	0.894 ± 0.072
	Clover30	0.098 ± 0.141	0.248 ± 0.348	**0.483 ± 0.135**	**0.636 ± 0.092**	0.077 ± 0.138	0.209 ± 0.352	0.433 ± 0.166	0.616 ± 0.141
	Clover50	0 ± 0	0 ± 0	0.396 ± 0.201	0.536 ± 0.205	0 ± 0	0 ± 0	**0.441 ± 0.145**	**0.584 ± 0.087**
	Subc0	**0.962 ± 0.054**	**0.977 ± 0.031**	**0.962 ± 0.054**	**0.977 ± 0.031**	0.94 ± 0.081	**0.97 ± 0.036**	0.94 ± 0.081	**0.97 ± 0.036**
	Subc30	0.653 ± 0.108	0.926 ± 0.041	**0.755 ± 0.073**	**0.933 ± 0.054**	0.649 ± 0.105	0.921 ± 0.045	0.744 ± 0.075	0.915 ± 0.075
	Subc50	0.251 ± 0.15	**0.754 ± 0.297**	**0.442 ± 0.105**	0.695 ± 0.116	0.233 ± 0.165	0.658 ± 0.369	0.434 ± 0.116	0.692 ± 0.116
	Subc70	0 ± 0	0 ± 0	**0.3 ± 0.229**	**0.431 ± 0.253**	0 ± 0	0 ± 0	**0.3 ± 0.229**	**0.431 ± 0.253**
	Pima	0.258 ± 0.09	0.269 ± 0.061	0.175 ± 0.048	0.244 ± 0.033	0.598 ± 0.11	0.694 ± 0.066	**0.652 ± 0.067**	**0.696 ± 0.049**
JRip	Paw0	0.832 ± 0.134	0.933 ± 0.068	**0.836 ± 0.085**	**0.942 ± 0.055**	0.824 ± 0.135	0.934 ± 0.078	0.836 ± 0.1	0.939 ± 0.069
	paw30	0.596 ± 0.178	0.793 ± 0.155	0.613 ± 0.152	0.833 ± 0.115	0.623 ± 0.156	**0.876 ± 0.104**	**0.651 ± 0.137**	0.864 ± 0.11
	Paw50	0.54 ± 0.165	**0.801 ± 0.155**	0.539 ± 0.146	0.729 ± 0.123	0.505 ± 0.089	0.77 ± 0.094	**0.543 ± 0.092**	0.7 ± 0.073
	Paw70	0.285 ± 0.187	0.651 ± 0.362	0.461 ± 0.077	0.641 ± 0.086	0.202 ± 0.199	0.455 ± 0.413	**0.469 ± 0.136**	**0.692 ± 0.117**
	Clover0	0.599 ± 0.097	**0.819 ± 0.11**	**0.689 ± 0.16**	0.813 ± 0.106	0.594 ± 0.139	0.808 ± 0.1	0.674 ± 0.094	0.812 ± 0.11
	Clover30	0.369 ± 0.12	**0.702 ± 0.175**	**0.529 ± 0.175**	0.685 ± 0.164	0.345 ± 0.192	0.594 ± 0.279	0.438 ± 0.144	0.636 ± 0.128
	Clover50	0.205 ± 0.226	0.432 ± 0.403	0.424 ± 0.132	0.641 ± 0.091	0.326 ± 0.215	0.569 ± 0.324	**0.513 ± 0.12**	**0.711 ± 0.109**
	Subc0	0.827 ± 0.101	0.907 ± 0.106	**0.902 ± 0.07**	**0.986 ± 0.009**	0.87 ± 0.077	0.956 ± 0.046	0.883 ± 0.065	0.965 ± 0.046
	Subc30	0.594 ± 0.173	**0.859 ± 0.115**	**0.605 ± 0.098**	0.789 ± 0.124	0.538 ± 0.175	0.834 ± 0.161	0.593 ± 0.089	0.773 ± 0.115
	Subc50	0.209 ± 0.143	0.548 ± 0.335	0.444 ± 0.175	0.642 ± 0.165	0.267 ± 0.194	0.643 ± 0.379	**0.462 ± 0.136**	**0.676 ± 0.115**
	Subc70	0.064 ± 0.139	0.122 ± 0.259	0.415 ± 0.114	0.62 ± 0.112	0.056 ± 0.176	0.087 ± 0.275	**0.379 ± 0.138**	**0.596 ± 0.133**
	Pima	0.248 ± 0.059	0.257 ± 0.024	0.167 ± 0.065	0.229 ± 0.054	0.6 ± 0.114	0.71 ± 0.064	**0.646 ± 0.071**	**0.704 ± 0.05**

TABLE IV: G-mean for artificial data sets with varying degree of the disturbance ratio using C4.5

Datasets	TempC	Base	R0	NCR	CO	SP2
paw0	**0.9773**	0.6744	0.9318	0.6599	0.9326	0.7330
paw30	0.8429	0.3286	0.8374	**0.8527**	0.8334	0.8337
paw50	**0.9480**	0.3162	0.8013	0.8200	0.7858	0.8075
paw70	0.7834	0.0152	0.7618	0.7824	0.7472	**0.8204**
clover0	**0.9729**	0.6381	0.8697	0.6367	0.8872	0.6750
clover30	**0.9405**	0.2566	0.7875	0.6758	0.7652	0.7686
clover50	0.7248	0.1102	0.7453	0.6184	0.7570	**0.7772**
clover70	0.6373	0.0211	0.7140	0.6244	0.7027	**0.7665**
subc0	**0.1000**	0.9738	0.9715	0.9613	0.9715	0.9716
subc30	**0.9803**	0.6524	0.7933	0.7845	0.7847	0.8144
subc50	**0.7921**	0.3518	0.7198	0.7534	0.7113	0.7747
subc70	0.6739	0.0000	0.7083	0.6720	0.7374	**0.7838**

VI. Conclusion and Final Remarks

In this paper, we propose a novel approach, temporary relabelling of classes for dealing with binary class imbalance. The three main contributions are the presentation and experimental evaluation of our approach and also the comparison with various state-of-art data level methods. From the results, it is evident that the deterioration in classification performance increases as the disturbance ratio increases in datasets. Also, ascertaining which k tuning in TempC was suitable for best performance was difficult as it greatly depended on the size of the dataset and the imbalance ratio between the minority and majority class. TempC's simplicity and ability to be combined with any classifier are some of its advantages. Our method's performance was satisfactory owing to the fact synthetic or artificial minority instances were not added to the original dataset; neither majority instances removed or permanently relabelled.

In our future work we intend to extend TempC from a two-phased to a multi-level approach and also adapt its framework to tackle multi-class imbalance problems.

References

[1] Fahad AM Alawadi. *Detection and classification of oil spills in MODIS satellite imagery*. PhD thesis, University of Southampton, 2011.

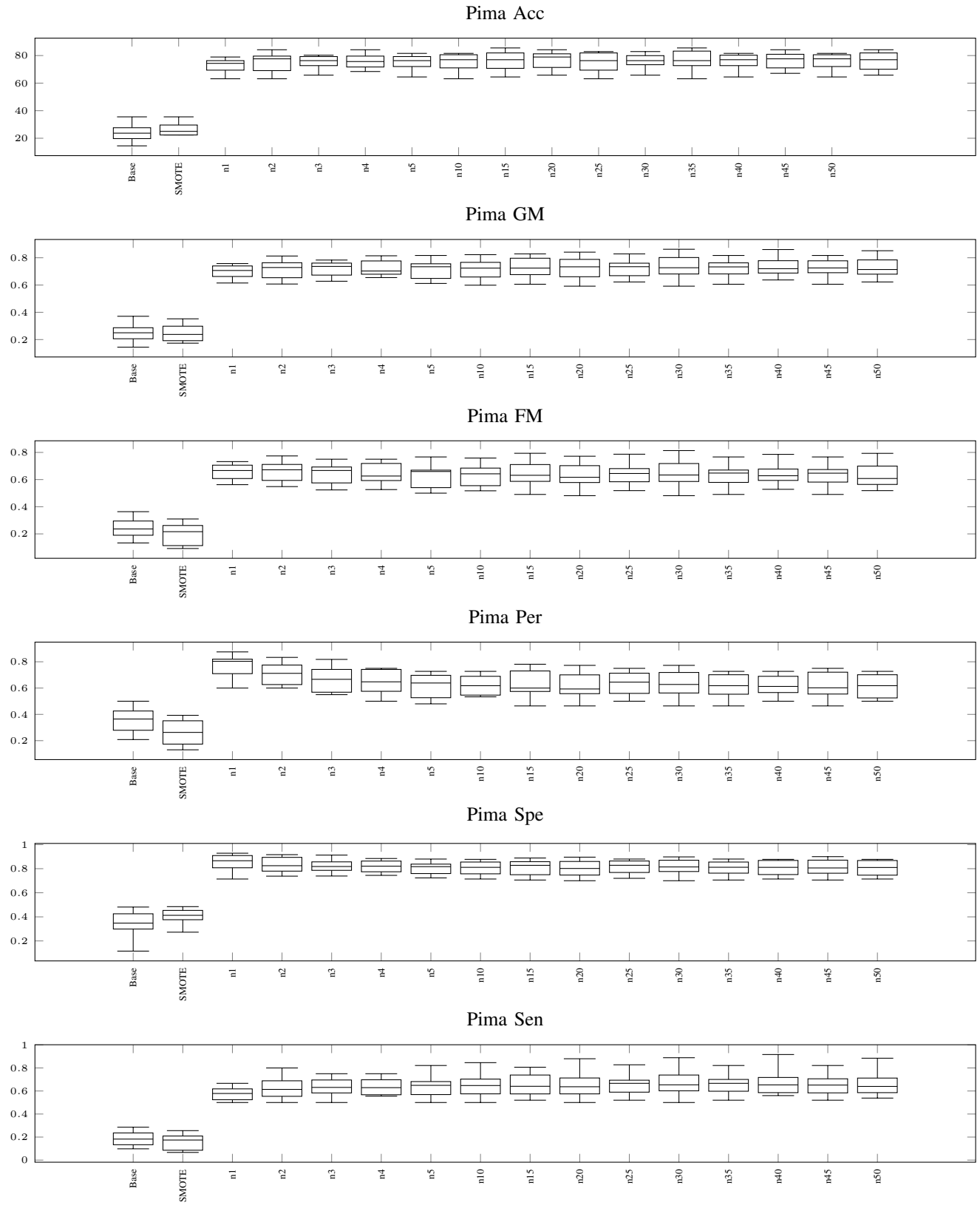

Fig. 2: Shows the performance of the TempC algorithm using different number of neighbours ranging between 1 to 50. The performance is evaluated using different measures. The base algorithm for Pima is the Random Forest.

[2] J Alcalá, A Fernández, J Luengo, J Derrac, S García, L Sánchez, and F Herrera. Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework. *Journal of Multiple-Valued Logic and Soft Computing*, 17(2-3):255–287, 2010.

[3] Arthur Asuncion and David Newman. Uci machine learning repository, 2007.

[4] Ricardo Barandela, José Salvador Sánchez, Vicente Garcıa, and Edgar Rangel. Strategies for learning in class imbalance problems. *Pattern Recognition*, 36(3):849–851, 2003.

[5] Gustavo EAPA Batista, Ronaldo C Prati, and Maria Carolina Monard. A study of the behavior of several methods for balancing machine learning training data. *ACM Sigkdd Explorations Newsletter*, 6(1):20–29, 2004.

[6] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[7] Carla E. Brodley and Mark A. Friedl. Identifying mislabeled training data. *Journal of Artificial Intelligence Research*, pages 131–167, 1999.

[8] Chumphol Bunkhumpornpat, Krung Sinapiromsaran, and Chidchanok Lursinsap. Safe-level-smote: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem. In *Advances in Knowledge Discovery and Data Mining*, pages 475–482. Springer, 2009.

[9] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, pages 321–357, 2002.

[10] William W Cohen. Fast effective rule induction. In *Proceedings of the twelfth international conference on machine learning*, pages 115–123, 1995.

[11] Misha Denil and Thomas Trappenberg. Overlap versus imbalance. In *Advances in Artificial Intelligence*, pages 220–231. Springer, 2010.

[12] Dennis J Drown, Taghi M Khoshgoftaar, and Naeem Seliya. Evolutionary sampling and software quality modeling of high-assurance systems. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 39(5):1097–1107, 2009.

[13] Vicente García, Ramón Alberto Mollineda, and José Salvador Sánchez. On the k-nn performance in a challenging scenario of imbalance and overlapping. *Pattern Analysis and Applications*, 11(3-4):269–280, 2008.

[14] Vicente García, Jose Sánchez, and Ramon Mollineda. An empirical study of the behavior of classifiers on imbalanced and overlapped data sets. In *Progress in Pattern Recognition, Image Analysis and Applications*, pages 397–406. Springer, 2007.

[15] Hui Han, Wen-Yuan Wang, and Bing-Huan Mao. Borderline-smote: a new over-sampling method in imbalanced data sets learning. In *Advances in intelligent computing*, pages 878–887. Springer, 2005.

[16] CG HILBORN. Dg lainiotis. *IEEE Transactions on Information theory*, 1968.

[17] Chi-Min Huang, Josh Jia-Ching Ying, and Vincent S Tseng. Mining usersâ behaviors and environments for semantic place prediction. In *Nokia Mobile Data Challenge 2012 Workshop. p. Dedicated task*, volume 1, 2012.

[18] Chi-Min Huang, Josh Jia-Ching Ying, Vincent S Tseng, and Zhi-Hua Zhou. Location semantics prediction for living analytics by mining smartphone data. In *Data Science and Advanced Analytics (DSAA), 2014 International Conference on*, pages 527–533. IEEE, 2014.

[19] Nathalie Japkowicz, Catherine Myers, Mark Gluck, et al. A novelty detection approach to classification. In *IJCAI*, pages 518–523, 1995.

[20] Taeho Jo and Nathalie Japkowicz. Class imbalances versus small disjuncts. *ACM SIGKDD Explorations Newsletter*, 6(1):40–49, 2004.

[21] Wael Khreich, Eric Granger, Ali Miri, and Robert Sabourin. Iterative boolean combination of classifiers in the roc space: An application to anomaly detection with hmms. *Pattern Recognition*, 43(8):2732–2752, 2010.

[22] Miroslav Kubat, Stan Matwin, et al. Addressing the curse of imbalanced training sets: one-sided selection. In *ICML*, volume 97, pages 179–186. Nashville, USA, 1997.

[23] Jorma Laurikkala. *Improving identification of difficult small classes by balancing class distribution*. Springer, 2001.

[24] Yi-Hung Liu and Yen-Ting Chen. Face recognition using total margin-based adaptive fuzzy support vector machines. *Neural Networks, IEEE Transactions on*, 18(1):178–192, 2007.

[25] Tomasz Maciejewski and Jerzy Stefanowski. Local neighbourhood extension of smote for mining imbalanced data. In *Computational Intelligence and Data Mining (CIDM), 2011 IEEE Symposium on*, pages 104–111. IEEE, 2011.

[26] Stan Matwin and Jan Mielniczuk. Challenges in computational statistics and data mining. 2015.

[27] Maciej A Mazurowski, Piotr A Habas, Jacek M Zurada, Joseph Y Lo, Jay A Baker, and Georgia D Tourassi. Training neural network classifiers for medical decision making: The effects of imbalanced datasets on classification performance. *Neural networks*, 21(2):427–436, 2008.

[28] Krystyna Napierała. Improving rule classifiers for imbalanced data. 2012.

[29] Krystyna Napierala and Jerzy Stefanowski. Identification of different types of minority class examples in imbalanced data. In *Hybrid artificial intelligent systems*, pages 139–150. Springer, 2012.

[30] Krystyna Napierała, Jerzy Stefanowski, and Szymon Wilk. Learning from imbalanced data in presence of noisy and borderline examples. In *Rough Sets and Current Trends in Computing*, pages 158–167. Springer, 2010.

[31] Todd Perry, Mohamed Bader-El-Den, and Steven Cooper. Imbalanced classification using genetically optimized cost sensitive classifiers. In *Evolutionary Computation (CEC), 2015 IEEE Congress on*, pages 680–687. IEEE, 2015.

[32] Clifton Phua, Damminda Alahakoon, and Vincent Lee. Minority report in fraud detection: classification of skewed data. *Acm sigkdd explorations newsletter*, 6(1):50–59, 2004.

[33] Ronaldo C Prati, Gustavo EAPA Batista, and Maria Carolina Monard. Class imbalances versus class overlapping: an analysis of a learning system behavior. In *MICAI 2004: Advances in Artificial Intelligence*, pages 312–321. Springer, 2004.

[34] RC Quinlan. 4.5: Programs for machine learning morgan kaufmann publishers inc. *San Francisco, USA*, 1993.

[35] Chris Seiffert, Taghi M Khoshgoftaar, Jason Van Hulse, and Andres Folleco. An empirical study of the classification performance of learners on imbalanced and noisy software quality data. *Information Sciences*, 259:571–595, 2014.

[36] Jerzy Stefanowski. Overlapping, rare examples and class decomposition in learning classifiers from imbalanced data. In *Emerging Paradigms in Machine Learning*, pages 277–306. Springer, 2013.

[37] Jerzy Stefanowski and Szymon Wilk. Selective pre-processing of imbalanced data for improving classification performance. In *Data Warehousing and Knowledge Discovery*, pages 283–292. Springer, 2008.

[38] Ivan Tomek. Two modifications of cnn. *IEEE Trans. Syst. Man Cybern.*, 6:769–772, 1976.

[39] Jason Van Hulse, Taghi M Khoshgoftaar, and Amri Napolitano. Experimental perspectives on learning from imbalanced data. In *Proceedings of the 24th international conference on Machine learning*, pages 935–942. ACM, 2007.

[40] Gary M Weiss. Mining with rare cases. In *Data Mining and Knowledge Discovery Handbook*, pages 765–776. Springer, 2005.

[41] Gary M Weiss. The impact of small disjuncts on classifier learning. In *Data Mining*, pages 193–226. Springer, 2010.

[42] D. Randall Wilson and Tony R. Martinez. Improved heterogeneous distance functions. *Journal of artificial intelligence research*, pages 1–34, 1997.

[43] Dennis L Wilson. Asymptotic properties of nearest neighbor rules using edited data. *Systems, Man and Cybernetics, IEEE Transactions on*, (3):408–421, 1972.

[44] Bee Wah Yap, Khatijahhusna Abd Rani, Hezlin Aryani Abd Rahman, Simon Fong, Zuraida Khairudin, and Nik Nik Abdullah. An application of oversampling, undersampling, bagging and boosting in handling imbalanced datasets. In *Proceedings of the First International Conference on Advanced Data and Information Engineering (DaEng-2013)*, pages 13–22. Springer, 2014.