A Comparison Study on State-of-the-art Minority Class Data Oversampling Techniques for Imbalanced Learning

Han Tang

A dissertation submitted in partial fulfilment of the requirements of
Technological University Dublin for the degree of
M.Sc. in Computing (Data Analytics)

January 2020

Declaration

I certify that this dissertation which I now submit for examination for the award of MSc in Computing (Data Analytics), is entirely my own work and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of my work.

This dissertation was prepared according to the regulations for postgraduate study of the Technological University Dublin and has not been submitted in whole or part for an award in any other Institute or University.

The work reported on in this dissertation conforms to the principles and requirements of the University's guidelines for ethics in research.

Signed: Han Tang

Date: 31.12.2019

Abstract

Learning from imbalanced data is a common problem in data mining and pattern recognition. Classifiers are usually unable to present ideal performances when they trained from imbalanced datasets. Approaches to solving this problem are data level and algorithmic level. This research focuses on data level approaches to solve the imbalanced learning problem, that is to resample the dataset to make the class distribution balance. The most popular method in this domain is Synthetic Minority Oversampling TEchnique, abbreviated as SMOTE. This research reviews SMOTE carefully, as well as extensions from it, and conducts a comparison study to find out the differences between different families of SMOTE extensions.

Reviewed SMOTE extension families in this research include methods combine undersampling and oversampling, range-restricted SMOTE extensions, and clustering-based SMOTE extensions. The oversampling methods tested with decision trees, k-Nearest Neighbours and Support Vector Machine classifiers, applied on 35 imbalanced datasets to produce statistically significant results. Experiment results indicate the SMOTE extensions do not perform differently in statistic level.

Keywords: oversampling, SMOTE, imbalanced learning, machine learning

Acknowledgments

I would like to express my sincerest thanks to my project supervisor Dr. Svetlana Hensman without whose guidance and valuable advice, this thesis wouldn't have come about. You are an amazing guide, an excellent teacher and a wonderful person!

I would also like to thank Dr. Luca Longo, M.Sc. theses coordinator, for his useful inputs in the formulation and design of research proposal.

Thank you Dr. David Leanard, Dr. Brendan Tierney for teaching me the knowledge of Machine Learning and Data Mining.

Lastly, love and best regards to my family for being with me throughout my tough times and for providing me all the strength and courage to endure this journey.

Contents

List of Figures

List of Tables

List of Acronyms

ROC	Receiver Operating Characteristic
AUC	Area Under Curve
SMOTE	Synthetic Minority Oversampling TEchnique
CNN	Condensed Nearest Neighbour
KNN	K-Nearest Neighbour
ENN	Edited Nearest Neighbour
ADASYN	ADAptive generation of SYNthetic examples
SLS	Safe-level SMOTE
AHC	Agglomerative Hierarchical Clustering
CBSO	Cluster Based Synthetic Oversampling
SOI	Synthetic Oversampling of Instances
SOMO	Self-Organising Map Oversampling
MOT2LD	Minority Oversam-pling Technique based on Local Densities
A-SUWO	Adaptive semi-unsupervised weighted oversampling
ProWSyn	Proximity Weighted Synthetic Over-sampling Technique
CURE-SMOTE	Clustering Using REpresentatives SMOTE
DGSMOTE	SMOTE based on Granulation Division
DBSMOTE	Density-Based SMOTE
MST-SMOTE	Minimum Spannings Tree SMOTE
ANOVA	ANalysis Of VAriance
MWMOTE	Majority Weighted Minority Oversampling TEchnique

Chapter 1

Introduction

1.1 Background

Developing classifiers based on imbalanced datasets is one of the critical challenges in data mining and pattern recognition. Imbalanced dataset refers to a dataset with classes not equally presented. For example, in a dataset with two classes (negative and positive), the number of one class is higher than it is of the other class. We call the overrepresented class Majority class and the underrepresented class Minority class. Machine learning from the imbalanced dataset is called imbalanced learning.

The imbalanced learning problem is prevalent in real life. For instance, if we are developing a classifier to identify cancerous tissues, only a small portion of people would have cancer, though our real desire is to identify those actual patients(Mazurowskia et al., 2008). We also encounter imbalanced data in text classification(Sun, peng Lim, & Liu, 2009), transaction fraud detection(Chan & Stolfo, 1998), credit risk analysis(Moradi & Rafiei, 2019).

The imbalanced dataset could influence the performance of the trained classifiers dramatically, as the classifier could overfit the data of the majority class, and underfit the data of the minority class, makes the classifier fail to give a precise prediction. In some cases, the imbalance between classes is extreme, such as on the order of 100:1, 1,000:1, or even 10,000:1, makes predictive analysis even harder to conduct(He & Shen, 2007)(Kubat, Holte, & Matwin, 1998)(Pearson, Gonye, & Schwaber, 2003).

Another problem of imbalanced learning is that it would be difficult to measure the performance of the classifiers trained from imbalanced data. For example, if an imbalanced dataset contains 99% of data of class "True", and only 1% of data of class "False", the performance of a classifier would be difficult to measure when we want to identify those samples of class "False" precisely. As the classification accuracy would still reach 99% for a classifier only produce "True" results, which is not our desire. Receiver Operating Characteristic (ROC) curve and F1-score are more suitable measuring methods for evaluating imbalanced learning. This part would be introduced detailly in Section 3.4.

Imbalances could not only exist in binary datasets but also could in datasets with various classes(Sun, Kamel, & Wang, 2006)(?, ?)(Zhi-Hua Zhou & Xu-Ying Liu, 2006). The imbalance of subclass within a class also raises concerns.

1.2 Research Project/problem

This research aims to propose a different method of dealing with imbalanced learning. The performances of classifiers trained on data selected oversampling methods would be measured by AUC and F1-score, to compared with other imbalanced learning technique. Several problems are going to be processed by different oversampling techniques, to train several state-of-the-art machine learning algorithms.

1.2.1 Research Problem

'Is there statistically significant difference in the performances of the combination of data resampling method, range-restricted SMOTE extensions, and clustering-based SMOTE extensions, when measured by F1 and AUC?'

Null hypothesis: There is no statistically significant difference in the performances of the combination of data resampling method, range-restricted SMOTE extensions, and clustering-based SMOTE extensions when measured by F1 and AUC.

Alternative hypothesis: There is a statistically significant difference in the performances of the combination of data resampling method, range-restricted SMOTE

extensions, and clustering-based SMOTE extensions when measured by F1 and AUC.

1.3 Research Objectives

The goal of this research is to propose a distinct data resampling technique and to compare its performance with the existed methods.

- Perform a comprehensive evaluation and analysis of the existing research relating to imbalanced learning, especially extensions of SMOTE.
- Propose a new technique to oversampling data of minority class base on SMOTE.
- Select 35 existed unbalanced datasets to compare the resampling methods.
- Apply several oversampling techniques to state-of-the-art machine learning algorithms, to produce classifiers for each problem.
- Evaluate the results by comparing the statistical results of the data oversampling techniques with the results of SMOTE with this testing hypothesis H0.
- Identify the limitations of this research study and suggest further space for improvement.

1.4 Research Methodologies

The research methodology used in this study is quantitative research.

There are Secondary data from 35 imbalanced datasets used for comparing different imbalanced learning techniques.

Multiple techniques resample imbalanced data of different domains, and to train several classifiers. The quantitative results are tested for significance, over the average of each dataset, to confirm or reject the null hypothesis.

1.5 Scope and Limitations

Imbalanced learning is a critical problem in the field of machine learning and data mining. In terms of the scope, this research focuses on different data resampling techniques dealing with imbalanced learning problems. Due to the constraint of time, this research could not compare multiple data resampling techniques, nor could improve the proposed technique for better performance.

1.6 Document Outline

The rest of the dissertation is structured as follows: Chapter 2 reviews the state-of-the-art approaches of imbalanced learning, start with the major categories of approaches, then focus on the oversampling techniques, especially SMOTE and its extensions. This part also includes the significant challenges in imbalanced learning. Chapter 3 proposes the experiment methodology, and introduce the way of measuring for this research. Chapter 4 includes the evaluation and discussion of the experiment results. Chapter 5 concludes the paper.

Chapter 2

Review of existing literature

2.1 The Nature of Imbalanced Learning

Class distribution can influence the process of classifier training. The problem of Imbalanced Learning has received a considerably large amount of attention for many years. In some cases, data imbalance would severely detriment classifier learning when the imbalanced ratio is extreme ($ImbalancedRatio \leq 0.6$). We define the imbalanced ratio as:

$$ImbalancedRatio = S_{min}/S_{maj}$$

Where S_{min} is the sample value of minority class, S_{maj} is the sample value of the majority class.

This section answers two questions of imbalanced learning: How class distribution influences the performance of classifier learning? Is there an optimal class distribution for classifier learning?

2.1.1 How Class Distribution Influences Classifier Learning?

Weiss analyses how the class distribution influences the performance of classifier learning and then inspects the best distribution for classifier training in his study(Weiss & Provost, 2001).

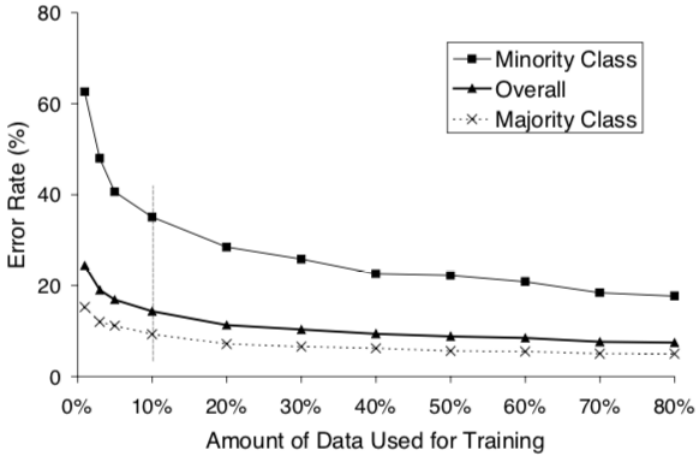


Figure 2.1: Learning Curves for the Letter-Vowel Data set by Classes (Weiss, 2001)

Researches about the impact of class distribution over classifiers performances and literature reviews cover this topic also produced by Shaza(Elrahman & Abraham, 2014), Ali(Ali, Shamsuddin, & Ralescu, 2015), and Satyam(Maheshwari, Jain, & Jadon, 2017).

His experiment covers over 25 imbalanced datasets, with imbalanced ratio ranges from 0.04 to 1. For each dataset, the class distribution of the training set varies by random oversampling so that the minority class accounts for the proportion of the training set, which ranges from 2% to 95%. The test set is formed by randomly selecting 25% of minority class samples and 25% of majority class samples. The rest samples form the training set.

His experiment shows that trained classifiers always have higher error rates on predicting minority class than majority class (In the Figure2.1(Weiss & Provost, 2001)).

The experiment result of the decision tree's error rate, which learnt from the Letter-Vowel Data set (imbalanced ratio of 0.25), showed in the Figure 2.1, is a typical example of how a standard classifier would perform when learnt from an imbalanced dataset. The Letter-Vowel Data set is described in Section 3.2.2, also named as the ISOLET dataset, is a dataset generated by 150 speakers name each letter of the alphabet. In the research of Weiss, the class letter 'a' selected as the minority class. According to Weiss's research, the error rate of the minority class is always higher than the majority class. Moreover, the error rate of the minority class shows a steeper

slope of decreasing when the number of training examples increases.

This finding can be explained by two reasons, from the perspective of the training dataset. Firstly, the test set would be imbalanced when the whole dataset is imbalanced. Classifiers would naturally have lower error rates to predict majority class. Imagine a test set with two classes, with 90% of majority class samples, and the rest 10% belong to the minority samples. A randomly generated and randomly labelled classifier would have 10% of error rate when it predicts an instance as majority clas. In contrary, the error rate of predicting a minority class sample for this classifier is naturally 90%. The other reason is that trained classifiers would always have more instances to build rules to predict majority samples, compare to the minority class. For the process of training a decision tree as an example, the majority-labelled leaves generally formed by more training samples than those minority-labelled leaves. The error rate for classifiers to predict those *small disjuncts* which only cover a small group of data, increases due to the classifiers are not statistically confident enough to generalize rules from them.

When viewing the problem from the perspective of classifier algorithms, we would find class distribution can influence the structures of trained classifiers. One way is that classifiers can strongly bias the majority instances, which improves the performance on predicting samples of the majority class but degrades the performance on predicting samples of the minority class. When we view the decision tree algorithm as an example, as it must cover all the feature value, it is possible for some leaves with no correspond training samples. Some Decision Trees could adopt the strategy to label these leaves to majority class, to improve the performance on the majority class examples, though the performance on the minority class examples would be the expense. The other way imbalanced data influence classifier training is that for majority class samples, the sample space is sound, the class boundary is clear. Though minority class, especially those small disjuncts within the minority class, are usually not well presented. The boundary between two classes, on the minority class side, can be vague. These make it hard for the classifier to define a clear boundary line between the two classes. That is to say, moving boundary line to minority class can cause a lower error rate on

predicting majority class samples, but a higher error rate on predicting minority class samples.

2.1.2 What is the Optimal Class Distribution for Classifier Learning?

Weiss(Weiss & Provost, 2001) also researched on the optimal range of class distribution for classifier learning, measured both in the learning error rate and the AUC. He experiments on 30 imbalanced datasets, by 13 different class distributions to produce a statistically significant result. It turns out none of which shows the natural distribution is within the range of optimal class distribution, among those datasets present statistical confident results. For this reason, it is responsible for saying that imbalance datasets are not appropriate to be used directly for classifier training with their natural distributions.

As for the optimal range of class distribution, classifiers could generally present better performances when oversampled minority class account for the class distribution range from 50% to 90%. Though the optimal class distribution varies in datasets, and it does not always present the best result when there are equal numbers of samples for each class, Weiss suggests that datasets could be re-sampled to 50%-50%, which would present results no worse or superior to those which use the natural class distribution.

2.2 Approaches to Imbalanced Learning

He classifies the approaches to imbalanced learning into several categories in his literature review about imbalanced learning(?, ?).

Apart from the approach of data sampling, which is described in detail from the Section 2.3 to the Section 2.6 and would be the focus of this dissertation, Cost-sensitive methods are also applied frequently to tackle imbalanced learning problems.

2.2.1 Sampling Method

The principle of sampling methods is to modify the imbalanced datasets by some mechanisms, to deliver a balanced distribution. As mentioned in Section 2.1, classifiers learned from balanced datasets generally perform better, or at least not worse than those trained from the original imbalanced datasets(Weiss & Provost, 2001)(Laurikkala, 2001)(Estabrooks, Jo, & Japkowicz, 2004).

2.2.2 Cost-sensitive Methods

The general idea of cost-sensitive learning is to assign different loss to each class, as the penalty to a model when misclassifies a sample of the minority class would be higher than of the majority class(Elkan, 2001). The most critical part of cost-sensitive learning is to define the cost matrix. Obviously, for a trained classifier, the cost of correctly classifying an instance should be lower than of wrongly classifying it. The difference of misclassification costs between classes always corresponds to the imbalance ratio of the dataset, and different methods are applied to compute them(Elkan, 2001).

The process of defining a cost matrix would not be explained in detail as it is not the focus of this dissertation.

Cost-sensitive learning aims to solve the problem of imbalanced learning at the algorithmic level. It is applied to multiple standard machine learning algorithms, includes decision tree(Maloof, 2003) and neural networks(Kukar & Kononenko, 1998).

For this dissertation, the focus is on tackling the imbalanced learning problem through data sampling method. The most intuitive way to manipulate an imbalanced dataset is to remove or duplicate some samples randomly, which is the principle of Random Sampling Method.

2.3 Random Sampling Method

The mechanics of Random Oversampling is to choose the samples of minority class to duplicate randomly, so the sample number of the minority class S_{min} would equal to that of the majority class S_{maj} by duplicating samples of minority class to introduce additional E samples.

$$S_{Oversampled} = S_{maj} + S_{min} + E$$

Vice versa, Random Undersampling technique is to pick data of the majority class to remove randomly.

$$S_{Undersampled} = S_{maj} + S_{min} - E$$

We can thus divide sampling techniques into two categories: Either to oversampling the minority class or to undersampling the majority class. In practical, they are both frequently applied.

Moreover, we apply these two methods at the same time in some cases(Batista, Bazzan, & Monard, 2003)(Batista, Prati, & Monard, 2004).

2.3.1 The Problems of Random Sampling Method

Though direct Random Sampling method could improve the performances of classifiers in some instances(Weiss & Provost, 2001), the method's superficiality also rises many other problems, limit the performances of trained classifiers. For Random Undersampling, the problem is more obvious: removing of instances in real datasets would inevitably cause the loss of concepts in the majority data, which is detrimental. On the other hand, random oversampling method replicates some instances for multiple times, makes classifiers 'lean' to the replicated instances, which means overfitting(?, ?)(Mease, Wyner, & Buja, 2007).

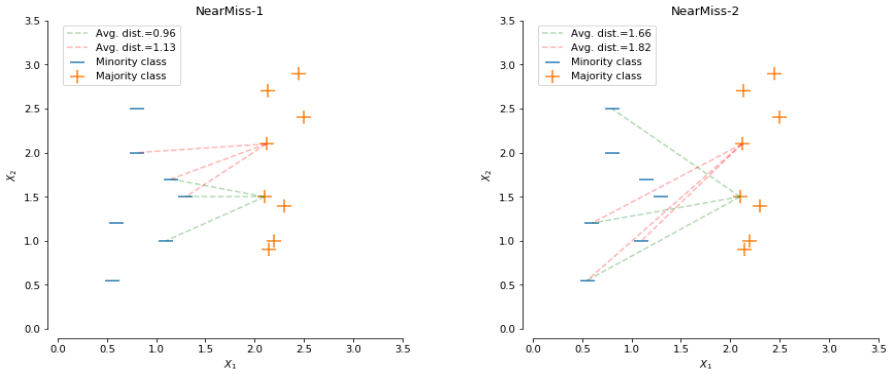
2.4 Informed Undersampling Method

As mentioned in the last section, random undersampling has its drawbacks and may not lead to what we expect. Thus other undersampling methods intend to undersample datasets based on their class distribution patterns, to pertain as much information as possible, and to remove the redundant samples at the same time. They are categorised as Informed Undersampling, distinguished from Random Undersampling. When we divide them furtherly, we can categorise them as Controlled Undersampling and Cleaning Undersamplng, based on if the result class distribution is controllable. They are both frequently performed, but for different purposes.

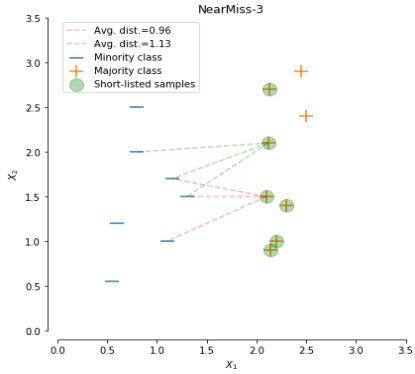
2.4.1 Controlled Undersampling Methods

NearMiss

Instead of randomly choosing samples of the majority class to remove, Zhang(Zhang & Mani, 2003) proposed the method of NearMiss Undersampling, a controlled way to undersample data of the majority class. The samples close to the decision boundary viewed as which can bring additional information to the model, are those we would like to retain. NearMiss method also intends to do that. It has three branches. The first method (NearMiss-1) selects the majority samples whose average distance to 3 of their closest minority class neighbours is the smallest to retain, as Figure 2.2a(Lemaître, Nogueira, & Aridas, 2017) shows. The number of majority samples to retain could be assigned and controlled in this way by repeating NearMiss-1 again and again till the number of retained samples meet the requirement. However, the presence of noise could limit the performance of this method, as this method could retain samples close to noises, instead of the class boundary. As Figure 2.3(Zhang & Mani, 2003) shows, the majority class in yellow is in greater variety before undersampling. However, the undersampling process removes some subgroups of the yellow class thoroughly and keeps too many samples containing little information at the same time. In this way, the undersampling process removes the samples supposed to be more valuable and reduces too much variety in the dataset, which is not wanted. Thus, we suffer a



(a) NearMiss1 Illustration (b) NearMiss2 Illustration



(c) NearMiss3 Illustration

Figure 2.2: Graph Illustration of the NearMiss Undersampling Method (Lemaître, Nogueira, 2017)

considerable amount of information loss during the undersampling process.

On the other hand, NearMiss-2 chooses the average distances to the farthest samples as the criteria instead, to retain those with the smallest average distance, as Figure 2.2b shows. It minimises the influence introduced by normal noises through the method of looking at the distances to the farthest samples, though noises could still hinder the undersampling result in the form of marginal outliers. As the figure 2.4(Zhang & Mani, 2003) shows, we still could not get an expected undersampled dataset sometimes.

NearMiss-3 is the version most likely to bring us the expected result. It contains two steps. Firstly, it selects a given number of minority class nearest neighbours for each majority class sample. Then it removes the samples whose average distance to

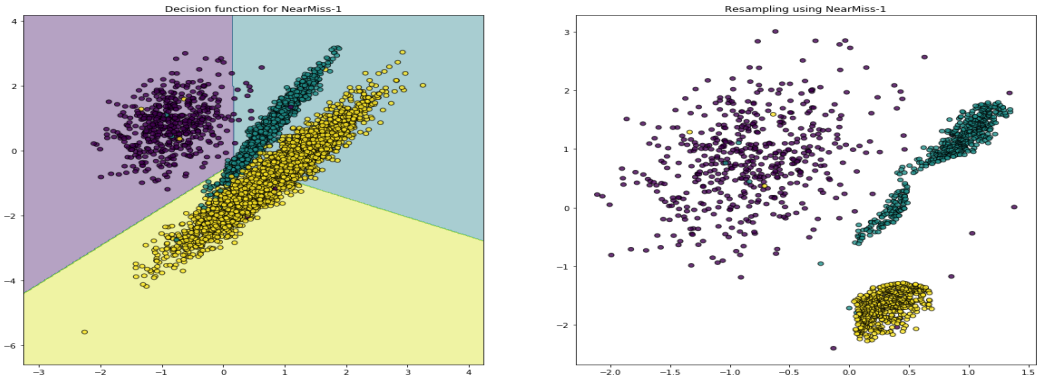


Figure 2.3: Example of NearMiss-1 on Artificial Dataset (Lemaître, Nogueira, 2017)

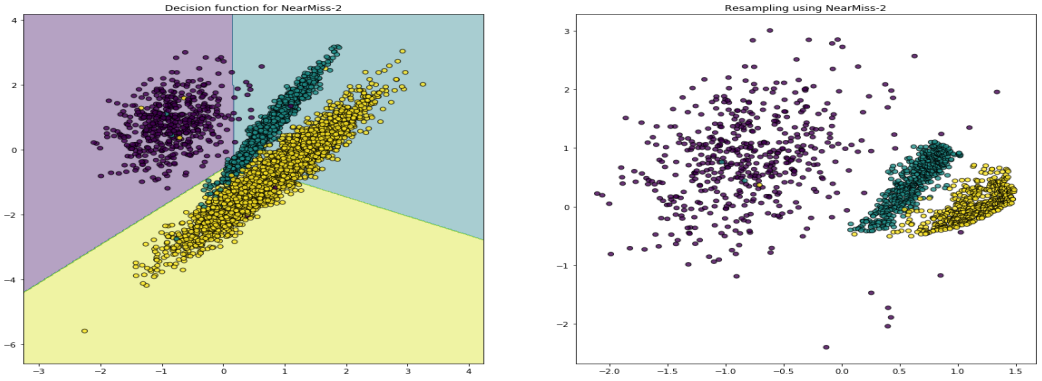


Figure 2.4: Example of NearMiss-2 on Artificial Dataset (Lemaître, Nogueira, 2017)

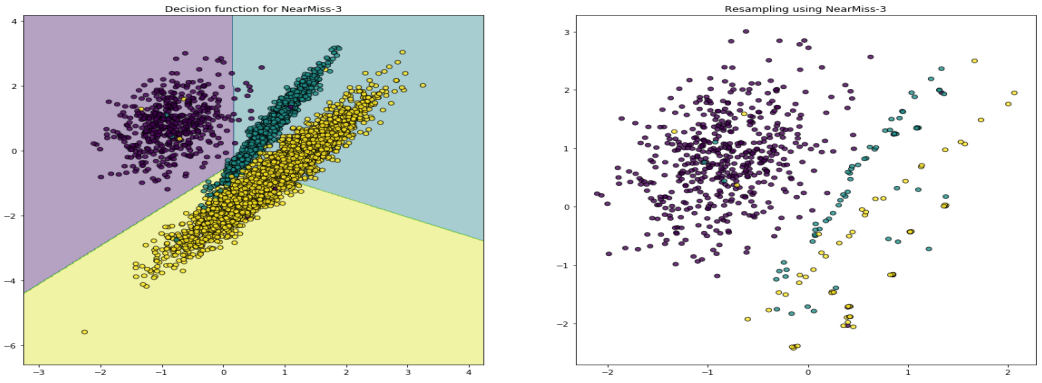


Figure 2.5: Example of NearMiss-3 on Artificial Dataset (Lemaître, Nogueira, 2017)

their nearest neighbours is the largest. It is less affected by noise through the first step selection. As the figure 2.5(Zhang & Mani, 2003) shows, this version brings an ideal result compared to the previous two versions, in this example.

NearMiss and Random Undersampling are classified as Controlled Undersampling Method as the user can specify the number of samples to remove. On the other hand, there are some methods applied meant for data cleansing and do not allow this kind of specification. We call them Cleaning Undersampling Method.

2.4.2 Cleaning Undersampling Methods

Condensed Nearest Neighbour

Hart(Hart, 1968) introduced the Condensed Nearest Neighbour method (CNN), an adjustment to K-Nearest Neighbour classifier. This method initially proposed to remove samples away from the decision boundary. As the task of machine learning algorithms is to define a decision boundary for classification, those close samples with different classes are more valuable to define the decision boundary. Thus samples close to the decision boundary are of higher value to classifiers, and we would like to remove the samples away from the decision boundary show little value to reduce the storage space of KNN models. Then Kubat(Kubat, 2000) extends this method to only remove samples of the majority class, to re-sample the imbalanced data, is called One-Sided Selection. One-Sided Selection goes over the class to be under-sampled, sample by

sample. It classifies each sample as the class of its nearest neighbour. A misclassified sample means it brings new information to the model, and then this sample would be retained. Otherwise, it goes to a set S. CNN would go through the set S again and again, until no further samples added to it. Then remove the set S from the original dataset.

Ever since introduced, Condensed Nearest Neighbour is developed further and applied to more areas. Liang(Liang, Xu, & Xiao, 2017) applies condensed nearest neighbour technique to convolutional neural networks, as a new method of image recognition. Siddappa(Siddappa & Kampalappa, 2019) and Angiulli(Angiulli, 2005) also introduce adjusted condensed nearest neighbour techniques.

Tomek's Links

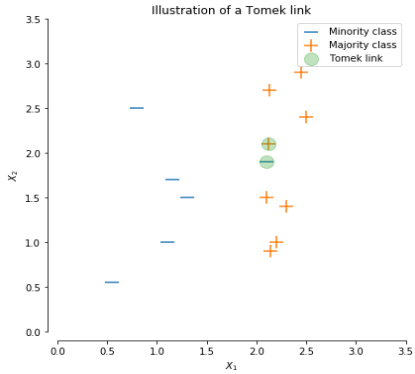
Another Cleaning Undersampling method called Tomek's Links ("Two Modifications of CNN", 1976) is applied as a modification of CNN. The general principle of this method is to introduce a Tomek's link between two nearest neighbours if they are of different classes (Figure 2.6a). Then it would either remove all the samples in the Tomek's links or just the samples of the majority class as Figure 2.6b depicts.

Edited Nearest Neighbour

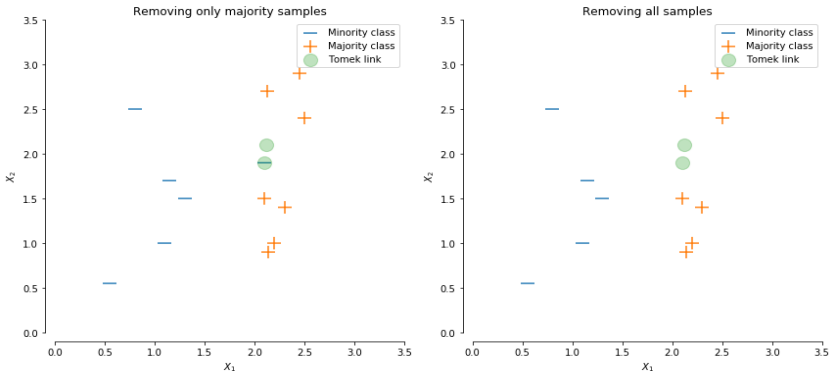
The Edited Nearest Neighbour method proposed by Wilson is also a method to under-sample datasets by removing their noises(Wilson, 1972). It edits the datasets by computing the K-Nearest neighbours of the samples in the class to be under-sampled. Two criteria applied to their Nearest Neighbours. They are (i) majority and (ii) all.

When the criterion is 'majority', ENN will remove the sample if most of the sample's K-Nearest neighbours are of the other class (of course, K should be an odd number). Vice versa, the other criterion means ENN will remove the sample when all of the sample's K-Nearest neighbours are of the other class.

There are also methods extended from the Edited Nearest Neighbours method. From the experiments conducted by Evan Tomek("An Experiment with the Edited Nearest-Neighbor Rule", 1976), Repeated Edited Nearest Neighbours and AllKNN are



(a) The illustration of Tomek's link



(b) Two undersampling methods

Figure 2.6: Graph Illustration of the Tomek's Links Undersampling Method (Lemaître, Nogueira, 2017)

extended from the original method and have similar impacts on the dataset, on the other word, datasets under-sampled by these methods all follow the similar pattern of distribution. However, this dissertation would not cover them in detail for considering the content length.

2.5 Oversampling Method

Considering the problem of overfitting caused by Random Oversampling, amended Oversampling methods are supposed to add new samples which are not precisely the same as the existed samples to the dataset without contaminating it. So methods are supposed to be capable of 'creating new samples' instead of merely replicating

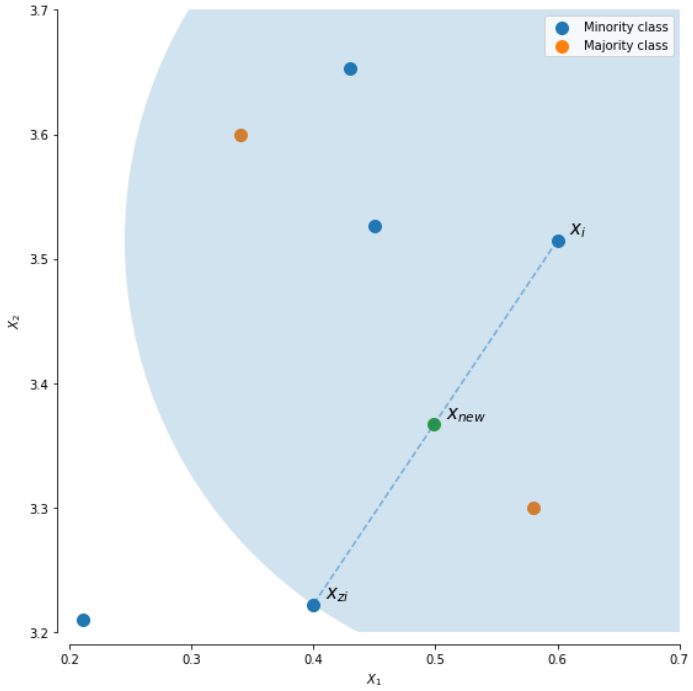


Figure 2.7: Graph illustration of SMOTE (Lemaître, Nogueira, 2017)

them. Synthetic Minority Oversampling Technique (SMOTE)(Chawla, Bowyer, Hall, & Kegelmeyer, 2002) is proposed for this purpose.

2.5.1 Synthetic Minority Oversampling Technique

SMOTE can over-sample the minority data in a controllable way, similar to the Random Oversampling method. One can decide the result class distribution of the dataset after re-sampling.

The first step of SMOTE is to choose K-nearest neighbours of each minority class sample and choose one of them. Then SMOTE would calculate the difference between each attribute of the two instances, which would multiply with a random number between 0 and 1. In this way, a new instance would be generated, with each of its attributes has a value different from the two instances used for creating it. In a dataset with two attributes for classification, the synthetic instance will generate along the line between the two original instances.

The method of SMOTE can generate artificial distinguish samples, which decrease

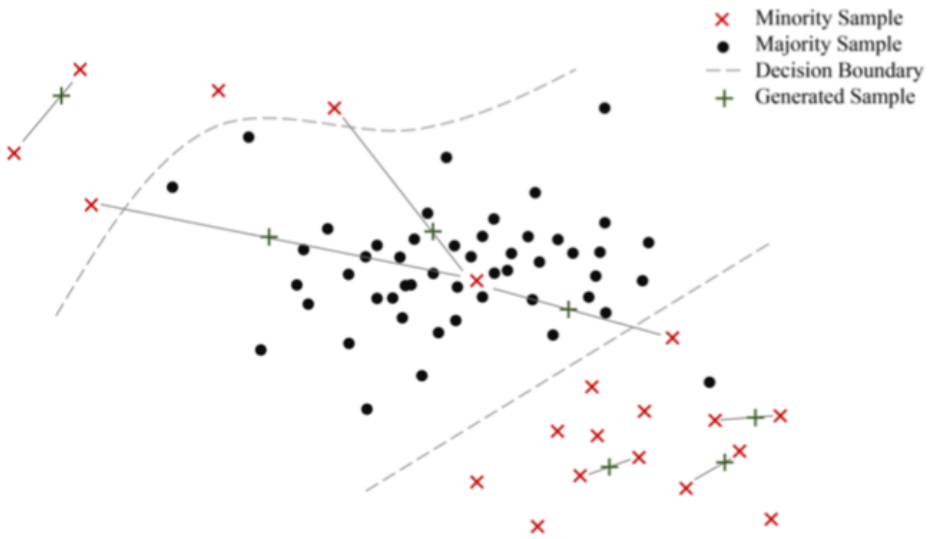


Figure 2.8: The Behaviour of SMOTE when Noise Exists (Last, Douzas, 2017)

the chance of overfitting caused in the process of over-sampling.

Ever since proposed in 2002, SMOTE has become the foundation for over-sampling with the artificial generation of minority class instances. Any artificial data generation method has some level of similarity with SMOTE(Fernández, Garcia, Herrera, & Chawla, 2018).

2.5.2 The Problems of SMOTE

Though SMOTE is widely applied, it still has many underlying problems. SMOTE shows a good result tackling the problem of between-class imbalance though it ignores the existence of within-class imbalance and small disjuncts of data. Study shows that the existence of small disjuncts in datasets can influence the classifiers significantly(Prati, Batista, & Monard, 2004). The small disjuncts in minority class are the major contributors of misclassifying the majority class samples.

Because the principle of the traditional SMOTE is to generate samples along the line between neighbour samples of the minority class, when SMOTE faces small disjuncts and subclasses in the minority class, it would potentially create samples between different subclasses when each subclass contains samples are nearest neighbours of each other.

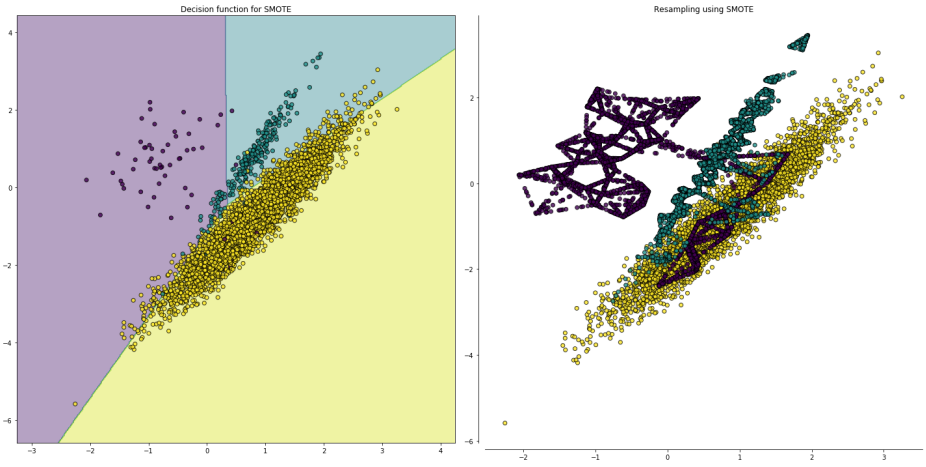


Figure 2.9: Noise Affects SMOTE(an Example on Simulation Data) (Lemaître, Nogueira, 2017)

As Figure 2.8(Last, Douzas, & Bação, 2017) shows, SMOTE would be affected by the noise of the minority class in this case. Generate artificial samples along the line between noise, and regular samples will contaminate the dataset, cause catastrophic consequences.

A more practical example as it shows in Figure 2.9. Purple denotes the minority class samples. As some purple samples exist outside of the minority class decision region and circled by other classes, when they are the nearest neighbour and samples are generated along the line between them and others, SMOTE will generate new samples inside the decision region of other classes. As a result, noise oversampled by mistake.

Noise and small data disjuncts are common in real-life datasets, and they are two of the significant reasons traditional SMOTE could not bring the ideal result of data oversampling. After the proposition of SMOTE, many other extensions of the original SMOTE are invited, aim to tackle the problems posed on SMOTE. They are based on different theories, implemented SMOTE from the ideas of multiple areas of knowledge and fork to further branches. In general, they improve the performances of SMOTE in some ways as it varies by the distribution of samples. Section 2.6 would cover some approaches to extent SMOTE, and categorise them by their methods of interpolation. The extensions described in the sections are either classic or state-of-

the-art or representative.

2.6 Extensions to SMOTE

2.6.1 Steps of SMOTE Extensions

In the literature review of SMOTE by Chawla(Fernández et al., 2018), SMOTE-based extensions can be categorised based on the steps it takes. Within this framework, researchers can review these over 80 extensions of SMOTE in an organised way.

The steps of each SMOTE extension process can summarise as:

- Initial selection of instances to be over-sampled.
- Integration with Under-sampling.
- Type of interpolation
- Operation with dimensionality change.
- Adaptive generation of synthetic examples.
- possibility of relabeling.
- Filtering of noisy generated instances.

Thus, we can categorise SMOTE extensions according to the method it takes at each step.

Initial selection of instances to be over-sampled

It intends to generate synthetic samples based on subgroups of the whole minority class samples. To prevent overlappings between classes, and to avoid introducing additional noises into datasets. For instance, Borderline-SMOTE applies this technique, which focuses on generating instances near the decision boundary(Han, Wang, & Mao, 2005).

Integration with Under-sampling

Under-sampling methods could be conducted at the beginning of over-sampling, to remove some instances of the majority class. Generally, under-sampling comes before over-sampling.

Type of interpolation

This property defines the category of mechanisms applied in the extension, which contains several groups. The way of interpolation is the originality for the most extension methods and is the crucial part in this section.

Operations with dimentionality change

Some approaches would include this step as reducing or augmenting dimensions before or during generating artificial instances. Common approaches include Principle Component Analysis (PCA)(Abdi & Hashemi, 2016)(Gu, Cai, & Zhu, 2009)(Xie, Jiang, Ye, & li, 2015), feature selection(Koto, 2014).

Adaptive generation of synthetic examples(ADASYN)

This step inspired by the adaptive generation method suggested in 2008 by He, Bai, Garcia, and Li(Haibo He, Yang Bai, Garcia, & Shutao Li, 2008). Classifiers always make mistakes in predicting. ADASYN intends to introduce a weighted distribution for each minority class samples. When classifier fails to predict the sample correctly, ADASYN will assign it a higher weight. The more times classifier fails to predict the same sample means the sample is more complicated and needs to be assigned a higher weight. In this way, ADASYN introduces a weighted distribution for different minority class examples according to the degree of difficulty in learning. Since came out, it has been adopted by many state-of-the-art approaches, to improve the quality of synthetic samples(Alejo, García, & Pacheco, 2015)(Rivera, 2017).

Relabeling

This technique would relabel some of the majority class samples to the other class, reaches decent performances in several cases(Dang, Tran, Hirose, & Satou, 2015)(Błaszczyński, Deckert, Stefanowski, & Wilk, 2012). Dang, Tran, Hirose and Satou(Dang et al., 2015) proposed SPY method, which changes the label of some majority class samples close to decision boundary to the minority class. Dang tests this method on datasets *yeast*, *haberman*, *blood*, and *Breast Cancer*(This dissertation includes these datasets also, see Section 3.2). According to his literature, SPY performs better the best on them, when applied with Support Vector Machine, k-NN, and Random Forest classifiers, and measured by Sensitivity, Specificity, and G-mean.

Filtering of noisy generated instances

To decrease the overlapping and to reduce noisy examples created during SMOTE, some extensions would involve a step of filtering noisy instances through under-sampling techniques. Common extensions of this technique include SMOTE-TomekLinks(Batista et al., 2004) and SMOTE-ENN(Batista et al., 2003). These undersampling techniques for noise filtering are in Section 2.4.2.

2.6.2 Common Interpolation Types in the SMOTE Extensions

The alternative mechanisms of SMOTE include range restriction(Han et al., 2005), interpolation following topologies based on geometric shapes, such as ellipse(Abdi & Hashemi, 2016), graph(Bunkhumpornpat & Subpaiboonkit, 2013), and clustering-based interpolation methods(Cohen, Hilario, Sax, Hugonnet, & Geissbuhler, 2006)(Barua, Islam, & Murase, 2011), which is also the focus of this dissertation. There are also many other ways of interpolation, which would exceed the scope of this dissertation.

2.6.3 Range Restricted Methods

SMOTE extensions applied interpolation method of range-restricted are similar in they choose some subgroups of datasets, especially those samples close to the decision boundary which would bring additional information to the model to oversample instead of all the minority class samples.

Borderline-SMOTE

The most popular range-restricted SMOTE extension is Borderline-SMOTE(Han et al., 2005), which proposed based on the assumption of examples away from the class border contribute little to classification. Thus it focuses on oversampling those "Borderline Samples". It generally contains two steps: First out the borderline examples, then generate new samples from them.

For every minority sample, borderline-SMOTE calculates its m nearest neighbours. Then it computes how many samples are of majority class, among the m samples. This number is *m'*.

The relation between m and *m'* leads to 3 ways of processing a sample. We denote the chosen sample as P.

$$m' = m$$

It means all majority samples surround P. In this case, P is noise, and the method would exclude it.

$$m/2 < m' \leq m$$

This means more than half but not all samples around P are majority samples. Though noise can also lead to this case on some occasions, this usually means P is close to the borderline between classes. In this case, P considered being easily misclassified and put into a dataset *DANGER*.

$$0 \leq m' < m/2$$

It means only a few majority samples around P. P is possible away from the decision boundary and contribute little to classification. Thus P would be kept but not processed further.

Finally, we use traditional SMOTE method to generate samples from the dataset *DANGER*.

Borderline-SMOTE2 differs from Borderline-SMOTE1 as it would randomly choose a number between 0 and 0.5 to multiply with the difference between the two selected samples when it generates samples between two classes. So the generated samples would be closer to the minority decision space. Experiments show the performance of Borderline-SMOTE is significantly higher than of that from Random Oversampling and traditional SMOTE in the datasets *'Pima'*, *'Satimage'*, and *'Haberman'*, either measured by Recall or F1-score.

Figure 2.10 shows another example of Borderline-SMOTE. The two figures in the left are the decision boundary of SVM trained from the original dataset. The right two figures are visualising the result of oversampling by Borderline-SMOTE.

The upper one is the oversampled result of Borderline-SMOTE1, and the bottom one is the oversampled result of Borderline-SMOTE2. The graphs show that this method performs well on amplifying the border examples without introducing too much overlapping at the same time, compared to Figure 2.9 though Borderline-SMOTE2 introduces more overlapping than Borderline-SMOTE1.

Safe-level SMOTE(SLS)

Inspired by the idea of selecting samples before sample generation, Bunkhumpornpat(Bunkhumpornpat, Sinapiromsaran, & Lursinsap, 2009) proposed a new method to select informative samples in a more elegant way. Safe level(SL) defined as

Safe-level(sl) = the number of minority samples in k nearest neighbours

For each minority class sample p, we define n as a selected nearest neighbours of p. Thus we can define $\mathrm{sl_p}$ and $\mathrm{sl_n}$, to calculate Safe-level Ratio:

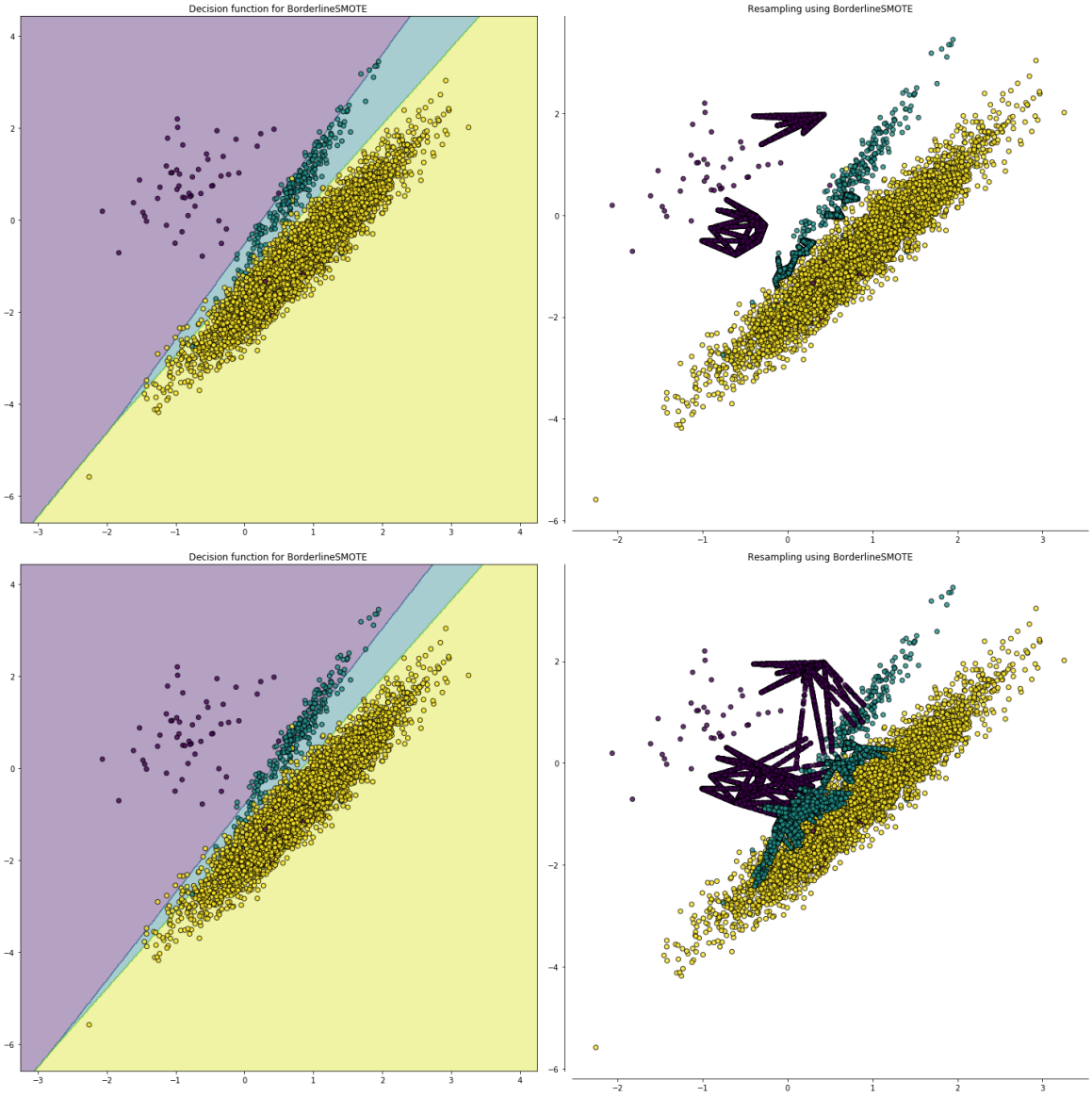


Figure 2.10: Example of Borderline-SMOTE (Lemaître, Nogueira, 2017)

Safe-level Ratio = sl of a positive instance p / sl of a nearest neighbour n

Higher safe level means the lower possibility the instance is noise. Safe-level Ratio measures the difference of information contribution between sample p and sample n. There are five distinct ways of generating samples, which depends on the values of Safe-level Ratio, $\mathrm{sl_p}$, and $\mathrm{sl_n}$.

1) Safe-level Ratio = $\infty, sl_{\mathrm{p}} = sl_{\mathrm{n}} = 0$. Both p and n are noise in this case. No samples would be generated.

2) Safe-level Ratio = $\infty, sl_{\mathrm{p}} \neq 0$. $\mathrm{sl_n} = 0$ means n is noise. Synthetic sample would be far from n by duplicating p.

3) Safe-level Ratio = 1. $\mathrm{sl_p}$ and $\mathrm{sl_n}$ are of the same value in this case. Synthetic samples would be generated randomly along the line between p and n, as the traditional SMOTE does.

4) Safe-level Ratio < 1. $\mathrm{sl_p}$ is greater than $\mathrm{sl_n}$. The region closer to p is safer for SMOTE, so synthetic samples would be generated closer to p as the distance [0, 1/Safe-level Ratio].

5) Safe-level Ratio > 1. In opposite to the case 4, the safe region is closer to n. The algorithm would generate samples closer to n as the distance [1/Safe-level Ratio, 1].

Safe-level SMOTE shows a better performance than SMOTE and Borderline-SMOTE in datasets *'Satimage'* and *'Haberman'*, according to the experiment results of (Bunkhumpornpat et al., 2009). Qorry(Qorry Meidianingsih, Erfiani, & Bagus Sartono, 2017) researches the performance of Safe-level SMOTE in imbalanced data in detail. Researchers test Safe-level SMOTE, and the original SMOTE on multiple simulation datasets, in different distribution patterns.

As the Figure 2.11(Qorry Meidianingsih et al., 2017) shows, the subplots show

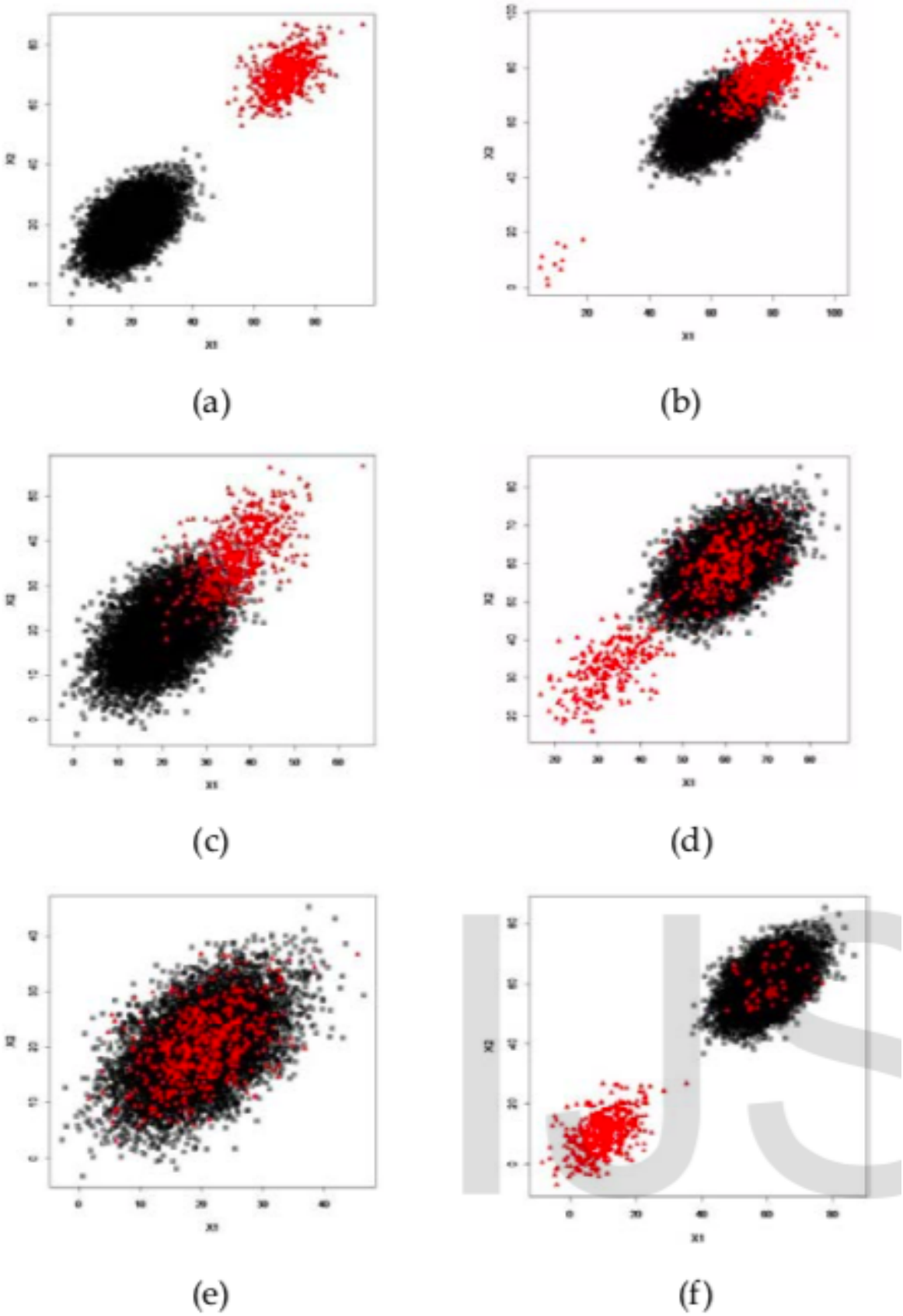


Figure 2.11: Illustration of Simulation Data Distribution of 6 Patterns (Qorry, Neidianingsih, 2017)

the distribution patterns of (a) separated, (b) intersected, (c) overlaid, (d) separated-intersected, (e) intersected-overlaid, (f) overlaid-separated respectively.

Qorr's(Qorry Meidianingsih et al., 2017) test shows Safe-Level SMOTE outperforms SMOTE on intersected, overlaid, separated-intersected, intersected-overlaid, and overlaid-separated datasets, when the learning algorithm is Support Vector Machine. In other words, the SVMs trained from datasets oversampled by Safe-Level SMOTE is always better than of the traditional SMOTE, as long as the two classes have any degree of intersections or overlaid. Though SMOTE can perform a result as well as it from Safe-level SMOTE when the two classes are separated, the commonality of intersection and overlay between classes in real-life datasets makes Safe-Level SMOTE a more suitable method than SMOTE.

Some other Range Restricted SMOTE extensions include Safe-level Graph SMOTE(Bunkhumpornpa & Subpaiboonkit, 2013). It is a technique that improves the Safe-level SMOTE by plotting the Safe-level graph of all the minority samples in the dataset, to inspect the distribution patterns. Extensions combine the technique of Threshold SMOTE and Attribute Bagging proposed by Wang(Wang, Yun, Huang, & Liu, 2013), SMOTE-out, SMOTE-Cosine, and Selected-SMOTE proposed by Fajri(Koto, 2014), Borderline Kernel-Based Over-sampling(Pérez-Ortiz, Gutiérrez, & Martínez, 2013), Local Neighbour SMOTE(Maciejewski & Stefanowski, 2011), and Deterministic Version of SMOTE(Torres, Carrasco-Ochoa, & Martínez-Trinidad, 2016). They are all proposed and tested in the literature. However, this dissertation will not cover them in detail.

2.6.4 Clustering-based Methods

Clustering-based SMOTE extensions aim to tackle the unwanted results of oversampling because of noise and small disjuncts, which can affect SMOTE(see section 2.5.2). Synthetic samples are generated either in the centroids of clusters or belong to the same cluster as the related samples. The method of clustering and the method of generating new samples are the novelties for newly proposed clustering-based SMOTE extensions.

2.6.5 Cohen's Cluster-based Approaches

Cohen(Cohen et al., 2006) proposed to apply Clustering techniques on minority samples before synthesising samples to avoid amplifying noise during the sample generation process, aims to augment the performance of SMOTE on datasets with more complicated distribution patterns. This approach firstly introduced to train proper models for nosocomial infection surveillance. Then it was reviewed by more researchers as it is also applicable to many other machine learning tasks.

Cohen firstly applies K-means Clustering to choose samples to retain/remove. K-means is an algorithm that can partition the dataset into K clusters and iterate the process for the optimised partition with an initially specified K. The algorithm randomly initialises each sample's belonging, then calculates the centroid of each cluster. The centroid is calculated by averaging the value of all the samples in the cluster, by all the attributes. The algorithm computes the squared distance between each sample and their cluster centroid, then to cumulate all the values within the same cluster. This process would repeat to minimise the accumulative squared distance within each cluster. Once the value of the sum of squared distance reaches the local minimum, the centroids would not move in further iterations any more.

Figure 2.12 and 2.13 shows the initial state and the final state of the K-Means algorithm partition a dataset into 3 clusters. The big black circles denote the centres of cluster centroids. At first, AHC assigns samples randomly. Then the algorithm optimises to minimise the sum of the squared distance to clarify the shape of three clusters. The black circles move to different directions, respectively. Cohen applies K-Means Clustering to undersample the majority class data. He assigns K to the size of the minority class. With $K = Smin$, he replaces the original majority class data with the K centroids computed from K-Means, equalises the two classes.

However, K-Means is not adequate for Minority Class Oversampling. When assigning K to $Smin$, the result would be K clusters, as each minority class sample is the centroid of its cluster, which contains itself. For this reason, K-Means could not produce ideal results for oversampling. Cohen applies Agglomerative Hierarchical Clustering(AHC). The number of clusters is adjustable, instead of being assigned ini-

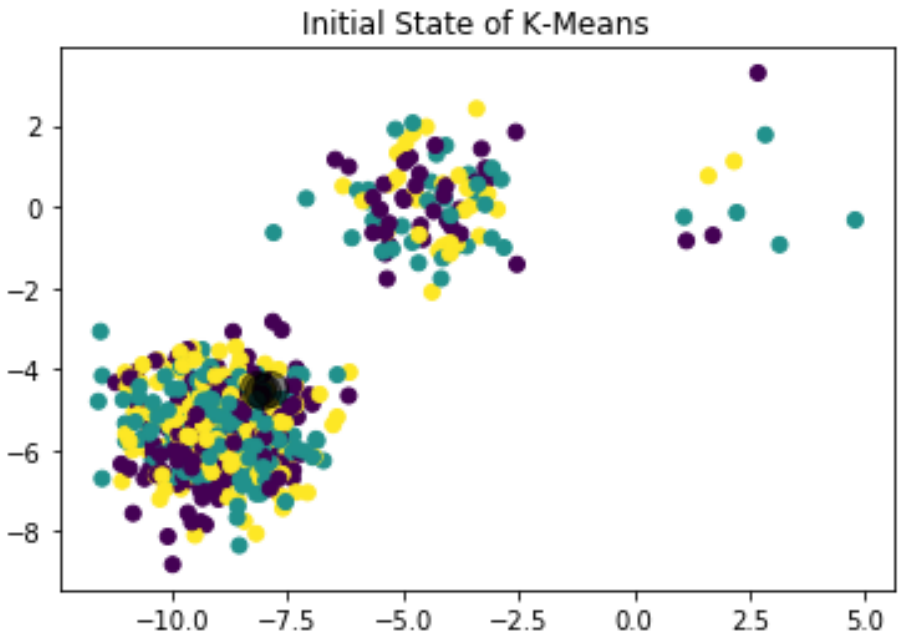


Figure 2.12: Random Initialize the dataset

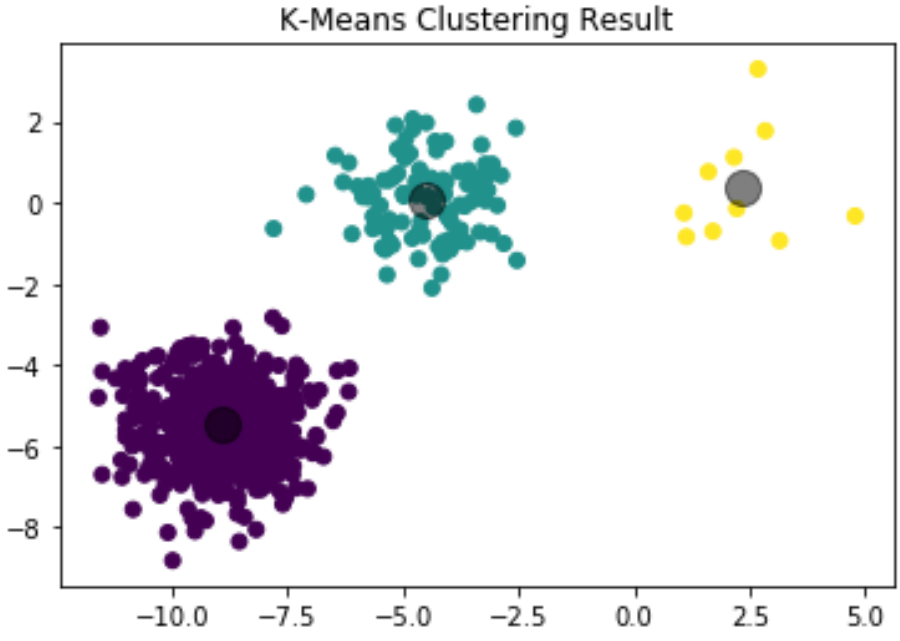
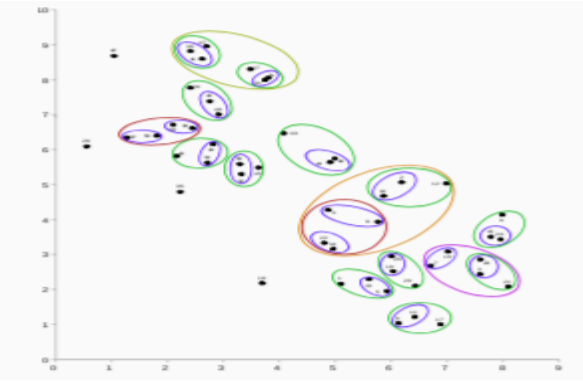
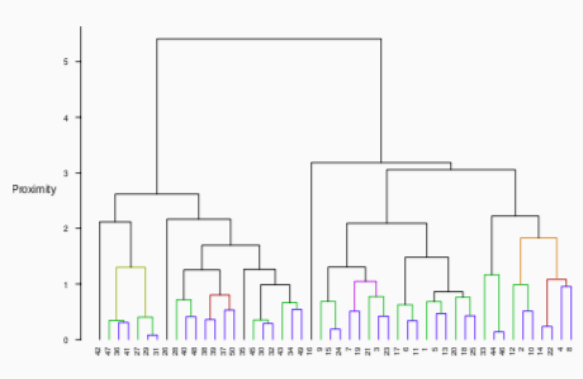


Figure 2.13: The Result of K-Means Clustering

tially in AHC. Agglomerative Hierarchical Clustering does not require to assume any particular number of clusters. It starts with merging (agglomerating) samples with the smallest distances as the first group of clusters. Then the distance increases gradually to form more clusters. The previously formed cluster would also merge with new samples to form nested clusters. As selected distance increases, the algorithm would produce further nesting. Finally, only one cluster would remain, and the process stops. The illustrations of AHC are in Figure 2.14 (Leonard, 2019)



(a) AHC Example Plot



(b) AHC Hierarchical Relations Illustration

Figure 2.14: Agglomerative Hierarchical Clustering Illustration (Leonard, 2019)

In Cohen's literature, his method only applied to a nosocomial dataset collected by The University Hospital of Geneva (HUG), which is a survey to assess the presence of an active nosocomial infection. The dataset contains 683 cases and 49 variables. Cohen applies the method of AHC to minority class oversampling through concatenating centroids of clusters by all the levels of the resulting dendrogram. Moreover, another

method of clustering-based resampling by Cohen is to combine the AHC based oversampling and the K-Means based undersampling. Classifiers trained by Naive Bayes, Decision Tree, and SVM. Those three data resampling methods all improve the classifiers' performances significantly compare to the models trained from raw data and random oversampled/undersampled data in terms of sensitivity, though the specificities suffer some decline.

2.6.6 Cluster Based Synthetic Oversampling(CBSO)

Another approach to solving the problem of data contamination in the process of synthesising data, which is introduced by the existence of subclasses and noise in the minority class, is Cluster-Based Synthetic Oversampling(CBSO) method, developed by Barua(Barua et al., 2011). The basic idea of CBSO is adopted from the ADASYN method, as they both apply a K-nearest neighbours based function to compute the value of each minority class sample, as well as which samples are most likely to be noise. CBSO differs from ADASYN as they apply a different method to generate synthetic data samples. ADASYN uses the k-NN approach for synthetic data sample generation, though CBSO does not.

Cluster Based Synthetic Oversampling starts with calculating the K nearest neighbours of each minority class samples, no matter which class their neighbours are. After that, it calculates the ratio r_i of the selected minority sample x_i as:

$$r_i = \triangle_i / K, i = 1...m_s$$

Where $\triangle_i$ is the number of majority class samples in K nearest neighours of x_i, m_s is the number of minority class samples. Therefore, the volume range of r_i by definition is [0,1]. When r_i is equal to 0, it means the K samples closest to x_i are all of the minority class. Contrary, $r_i = 1$ means the K nearest neighbours of x_i are all of the majority class. In other word, r_i is the measurement of the surrounding area of x_i, to justify the probability of whether x_i belongs to a data cluster.

After that, CBSO normalises the values of r_i, using the formula

$$\hat{r_i} = r_i / \sum_{i=1}^{m_s} r_i$$

This formula is the density function of the minority class samples clusters.

As

$$\sum \hat{r_i} = 1$$

In this way, CBSO finds the clusters in the minority class. Then it applies traditional SMOTE to generate samples within each cluster.

In the experiments conducted by Barua, Neural Networks trained from data re-balanced by CBSO presents better performances than trained from data re-balanced by original SMOTE, either inaccuracy, G-mean and F1. The tested datasets in this literature are in Table A.1.

2.6.7 Majority Weighted Oversampling Technique(MWMOTE)

MWMOTE, proposed by Barua, Islam and Yao(Barua, Islam, Yao, & Murase, 2014), is another method designed to generate new samples to resample the dataset without amplifying noises. The process of MWMOTE is composed of three phases. The first phase is to identify those hard-to-learn samples of the minority class, the set of all the minority class sample is denoted as S_{min}. The set of identified samples, which denotes as S_{imin}, forms a subclass of the minority class. Then MWMOTE gives a selection weight, denotes as S_w to each identified sample. In the third phase, MWMOTE generates new samples from samples in S_{imin}, using the corresponding S_w of each identified sample. The output set S_{omin} is added to S_{min}, as all the artificial samples.

Firstly we should define some terms of x for describing the process of the MWMOTE.

- (NN(x)) – The set of *k1* nearest neighbours of x
- ($N_{maj}(x)$) – The set of *k2* nearest neighbours of x
- ($N_{min}(x)$) – The set of *k3* nearest neighbours of x

Figure 2.15: Filterd minority set in MWMOTE(Barua, 2014)

Construction of Set S_{imin}

In the first phase, MWMOTE identifies the samples could be the most valuable to oversample. Those identified samples are usually near the decision boundary or belong to small clusters. The first step of phase 1 is to filter the noises in the dataset. It finds out the k nearest neighbours of each minority class sample, then removes those with all their k nearest neighbours are of the majority class. Similar to Borderline-SMOTE, this step prevents noises from taking part in the process of generating samples. The filtered dataset denoted as S_{minf}.

As the Figure2.15 shows, sample A and sample B removed from being used to generate new samples.

The next step MWMOTE looks at the majority class samples in the set $(N_{maj}(x))$. Samples in this set are close to decision boundary when (k2) is small. MWMOTE combines all the $(N_{maj}(x))$ (one (k2) nearest neighbours set for each minority sample) to form a borderline majority set S_{bmaj}.

As Figure 2.16 shows. When $k2 = 3$, MWMOTE finds out the 3 nearest neighbours of the majority class for each sample in S_{minf}. All the circled stars are samples of the formed set S_{bmaj}.

Similar to the step 2, for each sample in S_{bmaj}, MWMOTE finds out the $k3$ minority class nearest neighbours of it, forms a set $(N_{min}(x))$. Then we combine all the $(N_{min}(x))$, to form the subset S_{imin}, which contains all the identified samples. The $k3$ in this step should be large enough, so S_{imin} would contain all the informative samples.

Figure 2.16: Borderline majority set in MWMOTE(Barua, 2014)

Figure 2.17: Identified samples set in MWMOTE(Barua, 2014)

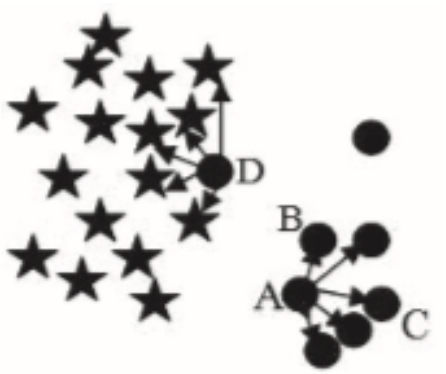


Figure 2.18: MWMOTE can outperform Borderline-SMOTE(Barua, 2014)

The Figure 2.17 shows the process of how S_{imin} is formed from S_{bmaj} in a simulated dataset when $k3 = 3$.

Though the first phase looks similar to the range-restricted process in Borderline-SMOTE, it can outperform Borderline-SMOTE in some specific ways of distribution. The Figure 2.18 shows this situation, where samples in a minority class cluster (the cluster sample A, B, and C belong to) are very close to each other, and the distance between the minority class cluster and the majority class sample is further. In this case, Borderline-SMOTE would only view sample D as the borderline sample, which is not the best approach. Though the identified sample set S_{imin} formed by MWMOTE would pick all the 'real' borderline samples.

Finding the Selection Weights $S_w(x_i)$ for members in S_{imin}

Due to the samples in S_{imin} could not be equally crucial for training, MWMOTE needs to assign weights to the samples, based on their importance. That is phase two of the MWMOTE process. For the more vital samples, MWMOTE would assign higher weights to them, which means there would be more artificial samples generated around them.

Three observations establish the principle for assigning weights:

Observation 1: Samples close to the decision boundary contain more information than those of further.

This observation is intuitive and straightforward. The closer the sample is to the

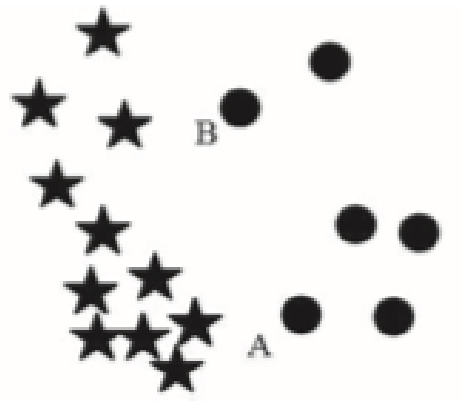


Figure 2.19: Illustration of principle for synthetic sample generating(Barua, 2014)

decision boundary, the more informative it is for training classifiers, the higher the weight we should assign to them.

Observation 2: The minority class samples in a sparse cluster are more critical than those in a dense cluster.

Samples in sparse clusters are more important than those in dense clusters during the process of generating synthetic samples. For the reason of dense clusters always contain more information than the sparse clusters, more synthetic samples should be generated in the sparse clusters, to avoid within-class imbalance. So there are higher weights for the minority class samples in sparse clusters.

Observation 3: The minority class samples near a dense majority class cluster are more critical than those near a sparse minority class cluster. In Figure 2.19, sample A and sample B are equally far from majority class clusters. However, the two clusters are of different levels of density. The sample near the cluster of higher density would be more difficult for classifiers to learn. In Figure 2.19, sample A is more challenging to learn and should be assigned a higher weight.

The information weights S_w for each sample in S_{imin} is assigned based the 3 observations. Each sample in the set S_{bmaj}, denotes as y_i gives a weight to each sample in the set S_{imin}, denotes as x_i. The weight is the information weight $I_w(y_i, xi)$.

For each x_i, the sum of all its information weights $I_w(y_i, xi)$s forms its selection weight $S_w(x_i)$.

The above statements are

$$S_w(x_i) = \sum_{y_i \in S_{bmaj}} I_w(y_i, x_i)$$

where

$$x_i \in S_{imin}$$

and

$$y_i \in S_{bmaj}$$

$I_w(y_i, xi)$ is computed by the formula:

$$I_w(y_i, x_i) = C_f(y_i, x_i) \times D_f(y_i, x_i)$$

where $C_f(y_i, x_i)$ is the closeness factor, according with the *observation 1*, and $D_f(y_i, x_i)$ is the density factor, according with the *observation 2*. The summation formula of the selection weight $S_w(x_i)$ facilitates the *observation 3*, as samples in S_{imin} which have more neighbors in S_{bmaj} would have more information weights $I_w(y_i, xi)$ to add to its selection weight $S_w(x_i)$, makes them higher.

Closeness factor $C_f(y_i, x_i)$ can be computed as the formula,

$$C_f\left(y_i, x_i\right) = \frac{f\left(\frac{1}{d_n(y_n, x_1)}\right)}{C_f(th)} * CMAX$$

where

$$d_n(y_i, x_i) = \frac{dist(y_i, x_i)}{l}$$

$dist(y_i, x_i)$ is the Euclidean distance between y_i and x_i and l is the number of dimensions in the data.

f is a cut-off function, $C_f(th)$ and $CMAX$ are parameters defined by the user.

The cut-off function f is defined as

$$f(x) = \begin{cases} x & \text{if } x \leq C_f(th) \\ C_f(th) & \text{otherwise} \end{cases}$$

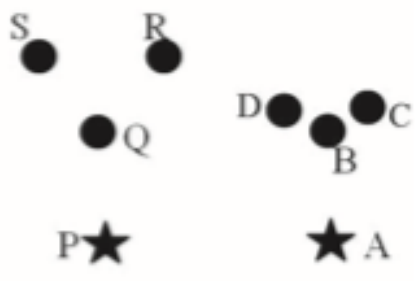


Figure 2.20: Illustration of the Density Factor(Barua, 2014)

Thus, the lower the distance between y_i and x_i is, the higher the $C_f(y_i, x_i)$ would be, though it would be infinitely large when the $dist(y_i, x_i)$ is very small. Its value would only be as large as $C\ M\ A\ X$ when the inverse of $dist(y_i, x_i)$ reaches the defined parameter $C_f(th)$.

The density factor $D_f(y_i, x_i)$ is to include the *observation 2*, where is supposed to be higher when the sample x_i belongs to a more sparse cluster.

The density factor $D_f(y_i, x_i)$ is to include the *observation 2*, where is supposed to be higher when the sample x_i belongs to a more sparse cluster. $D_f(y_i, x_i)$ is defined as the normalization of $C_f(y_i, x_i)$, to fit to the *observation 2* without violating *observation 1*. $D_f(y_i, x_i)$ is computed as:

$$D_f\left(y_i, x_i\right) = \frac{C_f\left(y_i, x_i\right)}{\sum_{q \in S_{imin}} C_f\left(y_i, q\right)}$$

We use the Figure 2.20 to illustrate the designing method of density factor. Though $C_f(P, Q) = C_f(A, B)$, $(C_f(A, C) = C_f(A, D)) > (C_f(P, S = C_f(P, R)$, makes the summation of $C_f(y_i, x_i)$ for the samples around A larger than those of the samples around P. That is a smaller $D_f(y_i, x_i)$ for the samples around A than it for the samples around P, makes the S_w around P higher.

Clustering and Synthetic Samples Generation

The last phase of MWMOTE is to generate new samples. It would only generate samples within the clusters formed by average-linkage agglomerative hierarchical clustering (AHC), which is in Section 2.6.5.

The method of generating new samples is still by the formula.

$$s = x + \alpha \times (y - x)$$

where x and y are original samples for synthetic sample generation, α is randomly picked in [0,1], and s is the generated new sample.

The Experiment Results of MWMOTE in works of literature

Barua tests MWMOTE on several artificial datasets and some real-world datasets (see Table A.1). It turns out MWMOTE outperforms SMOTE, ADASYN, and RAMO(Chen, He, & Garcia, 2010) on the datasets CTG, Semeion, Statlandsat, Breast Cancer, Breast tissues, Ecoli, and Glass, when the classifiers are single neural networks and AdaBoost Ensemble of neural networks, measured by the matrices F1, G-mean, and AUC.

2.6.8 Other Clustering-based Methods

Other Clustering-based synthetic sample generating methods, apart from those described above, include Synthetic Oversampling of Instances (SOI) by sánchez (Sánchez, Morales, & Gonzalez, 2013), Self-Organizing Map Oversampling (SOMO) by Douzas(Douzas & Bação, 2017), K-means SMOTE by Douzas(Last et al., 2017), Minority Oversampling Technique based on Local Densities (MOT2LD) by Xie(Xie et al., 2015), Adaptive semi-unsupervised weighted oversampling (A-SUWO) by Nekooeimehr(Nekooeimehr & Lai-Yuen, 2015), Proximity Weighted Synthetic Oversampling Technique (ProWSyn) by Barua(Barua, Islam, & Murase, 2013), Clustering Using REpresentatives SMOTE (CURE-SMOTE) by Ma(Ma & FAN, 2017), and SMOTE based on Granulation Division (DGSMOTE) by Gong(Gong & Gu, 2016).

2.6.9 SMOTE of Other Interpolation Methods

Apart from the Clustering-based and Range Restricted methods of interpolation, researchers adopt some other interpolation methods. Some interpolation methods

create samples by topologies based on geometric shapes. For instances, two methods of generating synthetic samples by Gazzah, one approach is based on polynomial fitting(Gazzah & ESSOUKRI BEN AMARA, 2008), the other approach applies a hybrid method of oversampling through "SMOTE star topology" and undersampling by removing considered irrelevant instances(Gazzah, Hechkel, & Essoukri Ben Amara, 2015). The geometric shapes for topology-based sample interpolation can be ellipses(Abdi & Hashemi, 2016), graph-based(such as Density-Based SMOTE (DBSMOTE) by Bunkhumpornpat(Bunkhumpornpat, Sinapiromsaran, & Lursinsap, 2011), and Minimum Spanning Tree SMOTE (MST-SMOTE) by Wang(Wang, Yao, Zhou, Leng, & Chen, 2013))

Another interpolation method is through random distribution, such as the gaussian. Instances of this interpolation method include Similarity-based UnderSampling and Normal Distribution-based Oversampling (SUNDO) by Cateni(Cateni, Colla, & Vannucci, 2011), probability density function estimation based over-sampling (PDFOS) by Gao(Gao, Hong, Chen, Harris, & Khalaf, 2014), Stochastic Sensitivity Oversampling (SSO) by Rong(Rong, Gong, & Ng, 2014), and a hybrid method by Sandhan(Sandhan & Choi, 2014).

2.6.10 Summary and Gaps of SMOTE Extensions

This literature review classifies SMOTE extensions based on the interpolation method it takes. The classes are the combination of undersampling and oversampling, range-restricted SMOTE extensions, and clustering-based SMOTE extensions. For the extensions covered in this chapter, the first category includes SMOTE-TomekLinks and SMOTE-ENN, the second category contains Borderline-SMOTE and Safe Level SMOTE, and the last category contains AHC, Cluster-Based Oversampling, and Majority Weighter Oversampling Technique. These three categories of extensions, along with traditional SMOTE and random oversampling as baseline methods, are methods to test for a statistically significant difference in this dissertation.

However, there is a common disadvantage in all the literature covered in this chapter. That is the methods proposed in their work are testes on too few datasets, and

the selected datasets for testing varied between method. The lack of testing results on a large number of datasets and not statistically confident enough results make the capability of proposed methods less convincing. Therefore, an experiment of applying these data oversampling methods on a more massive amount of datasets to prove whether there is a statistically significant difference between the performances of a classifier trained from all the proposed data resampling methods is essential.

2.7 Summary

This chapter carefully looks at state-of-the-art approaches to imbalanced learning. It starts by discussing how an imbalance in class distribution can influence the classifiers' performances. Then it looks at the approaches to solve it, includes data level approaches and algorithmic level approaches. After that, this research digs deeper into data-level approaches and focuses on minority class data oversampling methods. Methods of this kind extended from SMOTE, so this research reviewed popular SMOTE extensions and found out three of the most famous families of them, which are the combination of undersampling and oversampling method, restricted range method, and clustering-based method. This literature review explains all the mechanisms of methods tested in this dissertation in detail, as well as techniques and algorithms related to them, includes techniques about data clustering and noise removal.

Chapter 3

Experiment design and methodology

3.1 HYPOTHESIS

H0: There is no statistically significant difference in the performances of the combination of data resampling method, range-restricted SMOTE extensions, and clustering-based SMOTE extensions when measured by F1 and AUC.

H1: There is a statistically significant difference in the performances of the combination of data resampling method, range-restricted SMOTE extensions, and clustering-based SMOTE extensions when measured by F1 and AUC.

3.2 DATA

3.2.1 Data Selection

In this project, the Data Resampling methods are applied to 35 imbalanced datasets to compare their performances.

The datasets are from previous researches. Different resample methods cooperate with different classifiers produce all the experiment results, which compared with Random Oversampling, and SMOTE. The table A.1 outlines all the datasets selected

by the reviewed works of literature. This project uses the 35 datasets applied most frequently within them to bring more statistical significance and to minimize the bias. The 35 datasets (Datasets with multiple minority classes are tested for several times) are ecoli (target = imU), optical digits (target = 8), satimage (target = 4) , pen digits (target = 5), abalone (target = 7), sick euthyroid (target = sick euthyroid), spectrometer (target ¿= 44), car eval 4 (target = good), isolet (target = A, B), us crime (target ¿ 0.65), yeast ml8 (target = 8), scene (target = one label), libras move (target = 1), thyroid sick (target = sick), coil 2000 (target = minority), arrhythmia (target = 06), solar flare m0 (target = 0), oil (target = minority), car eval 4 (target = vgood), wine equality (target ¡= 4), letter img (target = Z), yeast ME2 (target = me2), webpage (target = minority), ozone level, mammography (target = minority), protein homo (target = minority), abalone (target = 19), haberman (target = minority), glass (target = 5,6,7), heart (target = minority), ionosphere (target = b), pima (target = minority), and Breast Cancer (target = malignant).

3.2.2 Data Description

This section describes the datasets applied in this dissertation in more detail, to understand both the data and how they correspond to real-life situations.

Ecoli

Ecoli is a dataset from UCI Machine Learning Repository, created and maintained by Nakai(Horton & Nakai, 1996). The dataset contains structured information to classify the type of E.coli proteins. The selected class of minority is imU, among the eight target classes. The rest classes are all viewed as the majority class.

Optical Digits

Optical Recognition of Handwritten Digits Data Set is an image dataset of handwritten digits. The images are of 8 × 8 pixels. The dataset is from the UCI Machine Learning Repository, created by Kaynak(Alpaydin & Kaynak, 1998)(Xu, Krzyzak, & Suen,

1992).

Satimage

The Statlog (Landsat Satellite) Data Set is about satellite images. The dataset contains multi-spectral values of pixels in 3x3 neighbourhoods in a satellite image and the classification associated with the central pixel in each neighbourhood. It uses information extracted from images instead of using image data directly. It is a dataset of structured information.

Pen digits

Pen-Based Recognition of Handwritten Digits Data Set is from UCI Machine Learning Repository, created by Alimoglu(Alimoglu, 1996)(Alimoglu & Alpaydin, 1997), contains information to classify handwritten digits. It collects the structured data processes from the raw collected images to improve classification accuracy.

Abalone

Abalone Data Set is from the UCI Machine Learning Repository. It contains information of abalones to predict their age. The class of age seven and the age 19 abalones applied as the minority class.

Sick Euthyroid

Thyroid Disease Data Set is from the UCI Machine Learning Repository. The structured dataset is for classifying sick euthyroid.

Spectrometer

The Infra-Red Astronomy Satellite(IRAS) collects Low-Resolution Spectrometer Data Set, to classify the class of Low-Resolution Observation (LRS). The LRS of above 44 viewed as the class of minority.

Car Evaluation

Car Evaluation Data Set is from the UCI Machine Learning Repository. The dataset is to evaluate the acceptability of cars based on their status information. Cars are 'unacc', 'acc', 'good', and 'very good'. The last two classes viewed as minority class in this dissertation.

ISOLET

ISOLET Data Set is generated by 150 speakers name each letter of the alphabet. The features for classification include spectral coefficients; contour features, sonorant features, pre-sonorant features, and post-sonorant features.

US Crime

Communities and Crime Data Set is to predict the total number of violent crimes per 100K population in communities. The minority class is assigned as ViolentCrimesPerPop > 0.65.

Libras Move

Libras Movement Data Set contains 15 classes of 25 instances each. Each class references to a hand movement type in LIBRAS (Portuguese name 'LÍngua BRAsileira de Sinais', official Brazilian signal language). The minority class assigned as 1, the rest classes are viewed together as the majority class.

The Insurance Company Dataset

The dataset used in the CoIL 2000 Challenge contains information on customers of an insurance company. The target class is whether the customer would be interested in buying a caravan insurance policy.

Arrhythmia

Arrhythmia Data Set is to distinguish between the presence and absence of cardiac arrhythmia and classify it in one of the 16 groups. The minority class for this dissertation assigned as 06.

Solar flare

Solar Flare Data Set counts the number of solar flares of a particular class that occur in 24 hours. The target minority class is selected as M (moderate class) = 0 for this research.

Wine quality

The goal of the Wine Quality Data Set is to model wine quality based on physicochemical tests(Cortez, Cerdeira, Almeida, Matos, & Reis, 1998). In this research, the wine of quality <= 4 assigned as the minority class.

Webpage

Webpage Dataset is for a text classification task. The texts are one-hot encoded. Feature spaces are all padded to 300 for texts with features less than this number.

Table 3.1: Data Description

No.	Dataset Name	Minority class	No. of features	No. of instances	Imbalance Ratio
1	ecoli	imU	7	326	8.6:1
2	opticial digits	8	64	5,620	9.1:1
3	satimage	4	36	6,435	9.3:1
4	pen digits	5	16	10,992	9.4:1
5	abalone	7	10	4,177	9.7:1
6	sick euthyroid	sick euthyroid	42	3,163	9.8:1
7	spectrometer	>=44	93	531	11:1

Continued on next page

Table 3.1 – continued from previous page

No.	Dataset Name	Minority class	No. of features	No. of instances	Imbalance Ratio
8	Car Evaluation	Good, Very good	21	1,728	12:1
9	ISOLET	A, B	617	7797	12:1
10	us crime	>0.65	100	1,994	12:1
11	yeast	ml8	103	2,407	13:1
12	scene	>one label	294	2,407	13:1
13	libras move	1	90	360	14:1
14	thyroid sick	sick	52	3,772	15:1
15	coil 2000	minority	85	9,822	16:1
16	arrhythmia	06	278	452	17:1
17	solar flare	M->0	32	1,389	19:1
18	oil	minority	49	937	22:1
19	Car Evaluation	Very good	21	1,728	26:1
20	wine quality	<=4	11	4,898	26:1
21	letter img	Z	16	20,000	26:1
22	yeast	Me2	8	1,484	28:1
23	webpage	minority	300	34,780	33:1
24	ozone level	minority	72	2,536	34:1
25	mammography	minority	6	11,183	42:1
26	protein homo	minority	74	145,751	11:1
27	abalone	19	10	4,177	130:1
28	haberman	minority	3	306	2.78:1
29	glass	5	9	214	15.5:1
30	glass	6	9	214	22.78:1
31	glass	6	9	214	6.4:1
32	heart	minority	13	303	1.19:1

Continued on next page

Table 3.1 – continued from previous page

No.	Dataset Name	Minority class	No. of features	No. of instances	Imbalance Ratio
33	ionosphere	b	34	351	1.8:1
34	pima	minority	8	768	1.86:1
35	Breast Cancer	Malignant	30	569	1.7:1

3.3 Research Methodology

First, we reassign each dataset, among the 35 test datasets, to 2 classes classification problem. We assign one of the minority classes to the minority class for testing, and assign the rest classes as the majority class altogether for dataset contains more than two classes. Then Features such as 'ID' would be dropped as it would not help classification. After that, we split the dataset into a training set and test set before resampling. If the instance number of the dataset is above 10,000, the train-test partition rate of the dataset is 7:3; otherwise 10-fold cross-validation is applied. After that, this research applies 11 minority class oversampling techniques, including Random Oversampling, SMOTE, Borderline-SMOTE1, Borderline-SMOTE2, SMOTE-ENN, SMOTE-TomekLinks, Agglomerative Hierarchical Clustering Based Oversampling, Safe-Level SMOTE, Clustering Based Oversampling, and Majority Weighted Oversampling Technique. For each dataset, decision tree classifier, k-nearest neighbour classifier, and Support Vector Machine classifier are for modelling. The number of nearest neighbours of k-NN is 5.

The metrics of performance measurement of classifiers are Classification accuracy, AUC value, F1 measures, and G-mean measures, whose definitions are in Section 3.4.

After acquiring the result performances of each case, this research conducts statistic test to inspect whether classifiers trained from datasets resampled by different methods perform differently in statistic level. Section 3.4 covers standard statistic tests include t-test and ANOVA in more detail.

This research aims to find out if classifiers trained from datasets resampled by different SMOTE extensions would perform differently, in a statistic level. It also aims to find out any SMOTE extension would bring better results, under the circumstances of some specific data distribution patterns.

3.4 Performance Evaluation

Measure metrics applied in this research are classification accuracy, AUC value, F1 measures, and G-mean. The definitions of them are in this section.

3.4.1 Confusion Matrix

The most intuitive and direct way to evaluate the performance of a classifier over test datasets is by the confusion matrix.

As illustrated in Figure 3.2, confusion matrix contains four situations: True Positive (TP), False Negative (FN), Falso Positive (FP), and True Negative (TN).

Multiple metrics can be calculated, based on the result of the confusion matrix.

Classification Accuracy

Classification Accuracy is defined as:

$$\mathrm{ACC} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{P} + \mathrm{N}} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}}$$

It shows how many instances are correctly classified by the classifier. However, it is not a very convincing measure method in the case of imbalanced learning, as mentioned in Section 2.1. If the test set contains only 10% of 'No' instances, a classifier only classifies instance to 'Yes' can get an accuracy of 90%, since the dataset is extremely imbalanced.

Thus Classification accuracy is only used as a reference of measurement in this research.

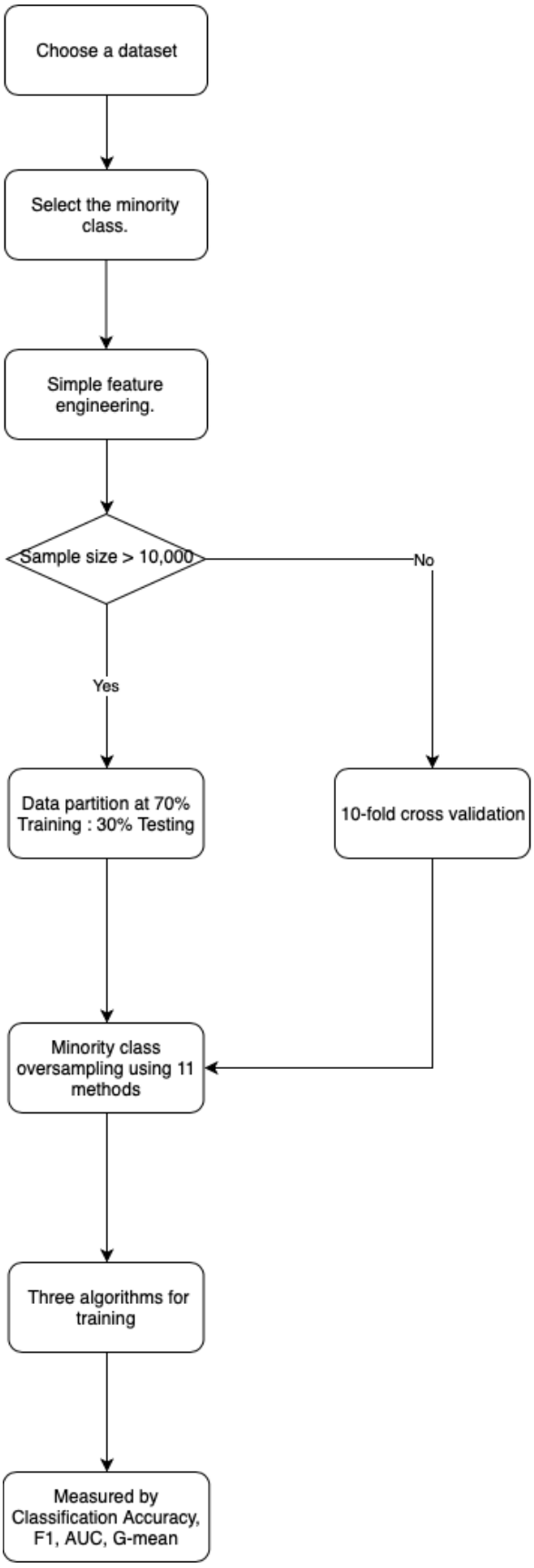


Figure 3.1: Experiment Flow Chart

n=165	Predicted: NO	Predicted: YES	
Actual: NO	TN = 50	FP = 10	60
Actual: YES	FN = 5	TP = 100	105
	55	110	

Figure 3.2: Confusion Matrix Illustration

F1 measures

To introduce F1 measures, we should compute two other values as the prerequisite. They are recall (also called sensitivity, true positive rate (TPR)):

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{P}} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} = 1 - \mathrm{FNR}$$

and precision (also called positive predictive value (PPV)):

$$\mathrm{PPV} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} = 1 - \mathrm{FDR}$$

Precision is the chance of correctly classifying when the classifier classifies an instance of Positive. The recall is the percentage of correctly classified positive instances among all of them. After calculating the values of precision and recall, F1 measures is the harmonic mean of recall and precision:

$$\mathrm{F}_1 = 2 \cdot \frac{\mathrm{PPV} \cdot \mathrm{TPR}}{\mathrm{PPV} + \mathrm{TPR}} = \frac{2\mathrm{TP}}{2\mathrm{TP} + \mathrm{FP} + \mathrm{FN}}$$

So that only when both recall and precision achieve high values, F1 score can reach

a high value.

Geometric Mean

Similarly, we can calculate the geometric mean (g) of recall and precision, by the formula:

$$\text{g} = \sqrt{\text{Precision} \times \text{Recall}}$$

After introducing the precision and the recall, the formula becomes:

$$g = \frac{TP}{\sqrt{(TP + FP)(TP + FN)}}$$

G-mean is a measurement similar to the F1 score. In this research, the main criteria for comparison are the AUC, and the F1 score, the rest two ways of measuring are metrics for referencing.

Area Under The Curve (AUC)

Area Under The Curve (AUC) value, refers to the area under the Receiver Operating Characteristics Curve, which is with True Positive Rate (TPR) against the False Positive Rate (FPR). Usually, TPR is on y-axis and FPR is on the x-axis.

False Positive Rate is defined by:

$$\text{FPR} = 1 - \text{Specificity}$$

Where

$$\text{Specificity} = \frac{\text{TN}}{\text{TN} + FP}$$

Thus

$$\text{FPR} = \frac{\text{FP}}{\text{TN} + \text{FP}}$$

AUC is the area under the ROC curve, as the Figure 3.3(*Understanding AUC - ROC Curve*, n.d.) illustrates.

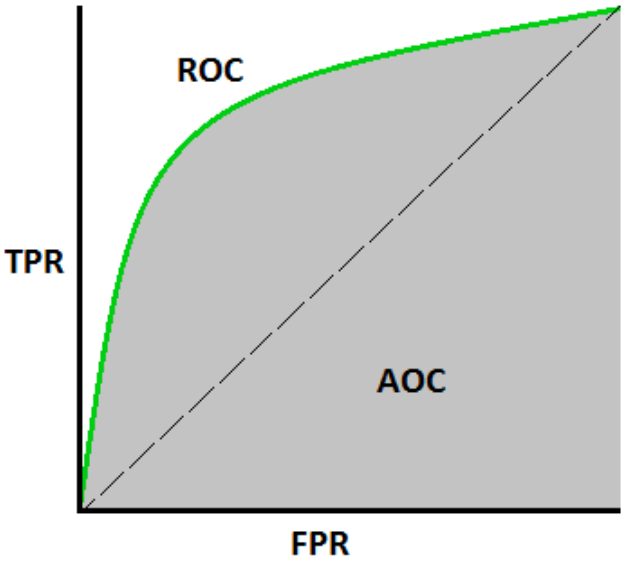


Figure 3.3: ROC Curve Illustration (Understanding AUC-ROC Curve)

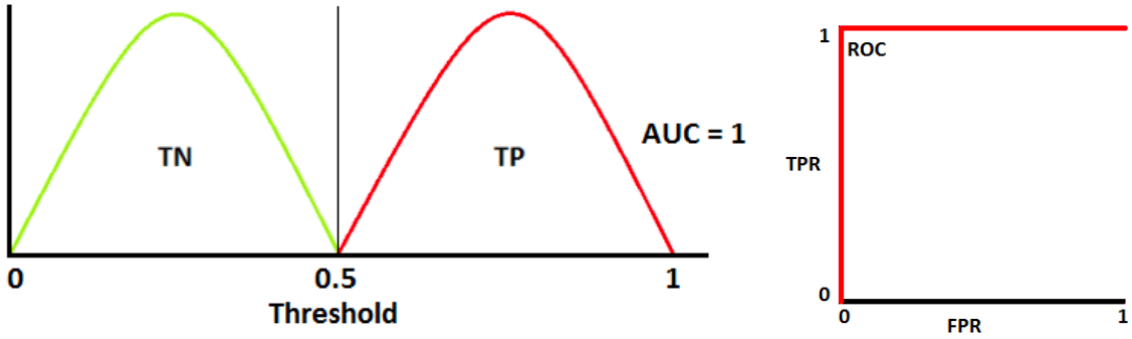


Figure 3.4: When AUC = 1 (Understanding AUC-ROC Curve)

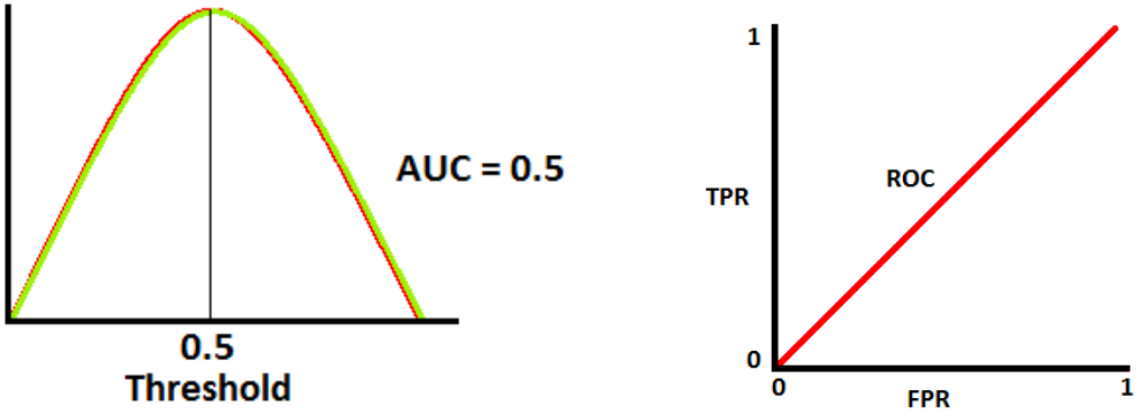


Figure 3.5: When AUC = 0.5 (Understanding AUC-ROC Curve)

The ROC Curve can tell the performance of the classifier to distinguish the two classes. Ideally, an AUC value close to 1 means the classifier can separate the two classes (As Figure 3.4 shows). An AUC value of 0.5 means the classifier has no capability of separating classes (As Figure 3.5 shows). The higher the AUC, the better the classifier is.

3.4.2 Statistic Tests

Statistic tests are also applied in this research to prove or reject our hypothesis. Different statistical tests are on different occasions. For this research, statistical tests conducted between three or more groups of digits (3 machine learning algorithms, ten over-sampling methods, and 35 imbalanced learning problems). ANOVA is for these occasions, where it is to compare the means of a condition between 2+ groups.

The concept of the t-test is the prerequisite to introducing ANOVA. T-test compares the means of two groups of samples. It applies to occasions where two groups of samples compared to inspect their differences. The paired-sample T-test is for two dependent groups of samples. For two groups of independent samples, the applied one is independent sample t-test.

Similar to t-test, ANOVA compares the group means between multiple groups ($>$ 2). To conduct t-test and ANOVA, the samples for testing should meet the three assumptions. There are three assumptions of ANOVA. Independent assumption: the elements of one sample are not related to those of the other sample. Normality assumption: the distribution of samples is normal, so means can be used as a method of measuring central tendency. Equal variance assumption: the population variance for each group is equal.

ANOVA test would give two values: F-ratio and p-value, where F-ratio shows the magnitude of difference and p-value shows whether the difference is statistically significant.

Another issue of ANOVA is that it is an omnibus test, that is to say ANOVA can tell you if there is an effect in the samples but not where. Thus we need to conduct posthoc testing to identify the level of effect and where the effect is evident. The

post-doc testing for this research is Tukey's HSD.

Before conducting ANOVA test, we need to conduct bartlett's test to check for homoscedasticity for the variable of interest across groups. If the p-value $> .05$, we can argue that the variants are homogeneous and the preliminaries for ANOVA test do not meet.

For groups of data can not meet the assumption of ANOVA test, Kruskal-Wallis test(Kruskal & Wallis, 1952) is the non-parametric counterpart of the one-way independent ANOVA.

Chapter 4

Results, evaluation and discussion

4.1 Experiment Result

Crude Experiment Results are in the Table A.2, the Table A.3, and the Table A.4. Those are the experiment results of 35 datasets (include the same dataset of different target classes) resampled by 11 different oversampling techniques, trained by decision tree classifiers, five nearest neighbours classifiers, and support vector machines. This chapter is about processing and discussing the initial experiment results, aim to figure out how different minority class oversampling methods can help to produce better results in imbalanced learning tasks. Especially, how different minority class oversampling methods (SMOTE extensions specifically) coordinate different distribution patterns of the dataset to produce synthetic samples in the way as optimized as possible.

4.2 Result Discussion

4.2.1 Comparing the Performances of Different Algorithms on Imbalanced Learning Tasks

The first part is to inspect whether different machine learning algorithms (for this research, tested algorithms are decision tree, five nearest neighbours, and support

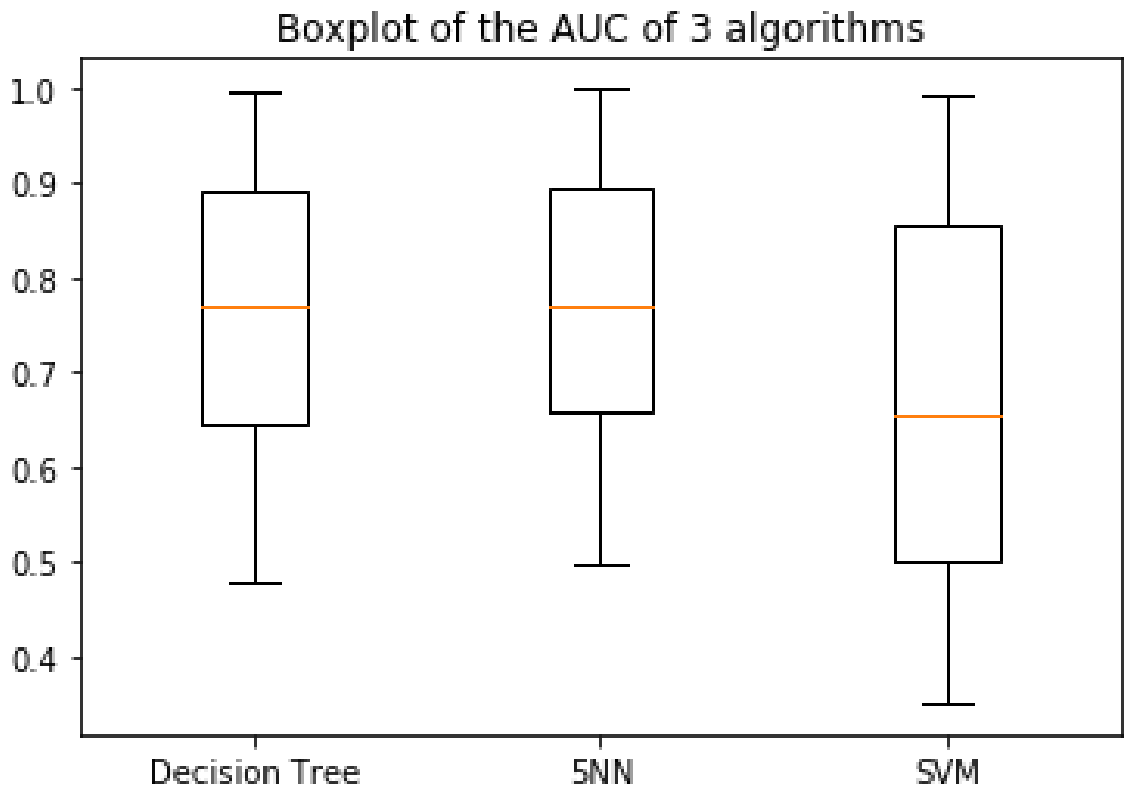


Figure 4.1: AUC of the Three Machine Learning Algorithms

vector machine) perform statistical differently on the selected imbalanced learning problems. Though the training datasets are over-sampled, different algorithms could still present different abilities of learning.

Measured by AUC

The boxplot of the three algorithms' performances, measured by AUC is of Figure 4.1.

It shows the average AUC of Decision Tree and the k-NN (k = 5) is close to each other and both higher than it is of SVM. There are also more cases for SVM that AUC = 0.5, which means the classifier is not separating the two classes at all. From the boxplot, t decision tree and hekNN are more appropriate for resampled imbalanced learning problems, compare to SVM.

Bartlett's test indicates the variables are not homogeneous at the $p < .05$ level (Stats=21.451, p=8.085e-06). Therefore, ANOVA is feasible for groups of data. The

```
F = 31.133665607937434
p-value = 7.399680880805996e-14
Tukey HSD:     Multiple Comparison of Means - Tukey HSD, FWER=0.05
=============================================================
    group1    group2 meandiff p-adj   lower   upper  reject
-------------------------------------------------------------
Decision tree    KNN   0.0105 0.6336 -0.0171   0.038  False
Decision tree    SVM  -0.0752  0.001  -0.103 -0.0475   True
          KNN    SVM  -0.0857  0.001 -0.1134 -0.0579   True
-------------------------------------------------------------
```

Figure 4.2: ANOVA test of 3 Algorithms Measured by AUC

test result of the ANOVA test and posthoc comparisons using Tukey HSD is in Figure 4.2

A one-way between-groups analysis of variance conducted to explore the impact of machine learning algorithms on the performances of classifiers on imbalanced learning problems, as measured by AUC. Participants contain three groups according to their applied machine learning algorithms (Group 1: Decision Tree Classifier; Group 2: kNN Classifier where k = 5; Group 3: Support Vector Machine Classifier). There was a statistically significant difference at the $p < .05$ level in AUC for the three groups of algorithms: F = 31.134, p=7.4e-14. Post-hoc comparisons using the Tukey HSD test (alpha = 0.5) indicated that the mean score for Group Decision Tree Classifier was statistically different from Group SVM. Group KNN was statistically different from Group SVM though Group KNN and Group Decision Tree did not differ significantly from each other.

Measured by F1 score

Similar to the performance results measured by AUC, Decision tree and k-NN (k = 5) present high average performances than the F1 scores SVM presents. The F1 scores of Decision Tree are also slightly higher than it of k-NN. For SVM, there are more cases which F1 score is equal to 0 or very close to 0, meaning SVM is only presenting classifiers predict all or most of the samples as the majority class, instead of classifying them.

Bartlett's test over the variables indicates they are not homogeneous at the $p <$

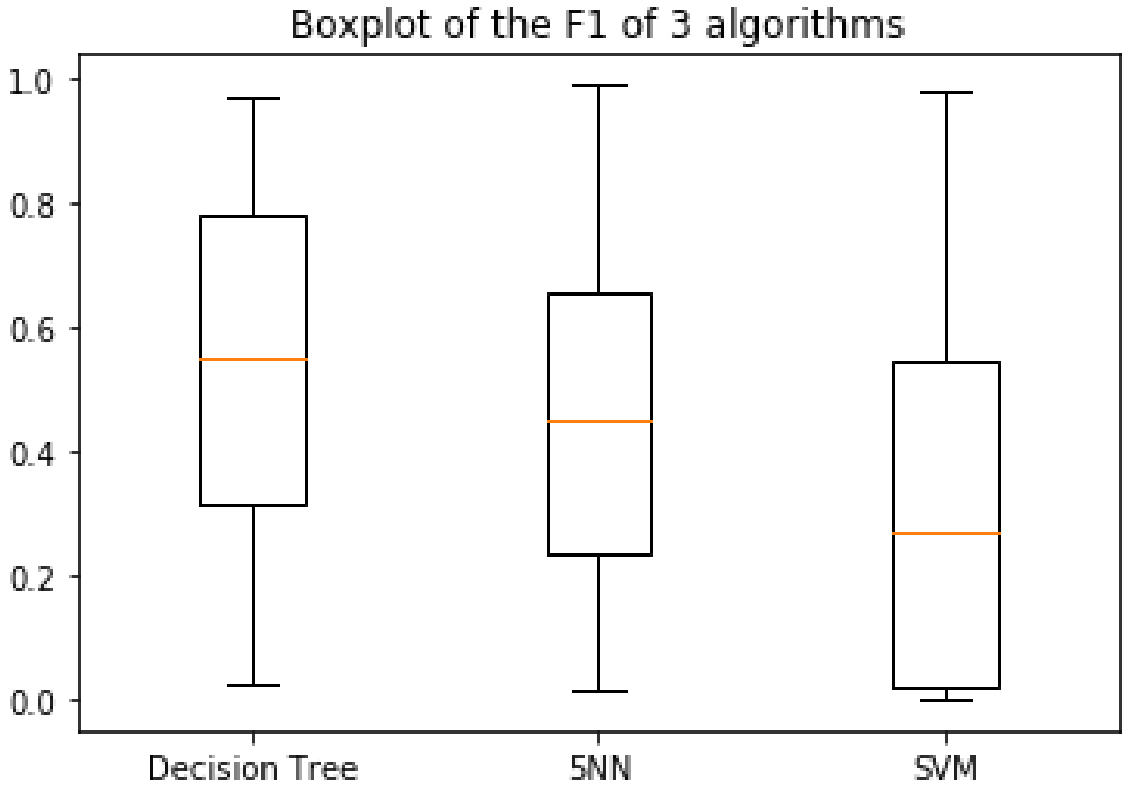


Figure 4.3: F1 score of the Three Machine Learning Algorithms

```
F = 50.0952057931366
p-value = 1.7052049483183237e-21
Tukey HSD:     Multiple Comparison of Means - Tukey HSD, FWER=0.05
=============================================================
    group1    group2 meandiff p-adj   lower   upper  reject
-------------------------------------------------------------
Decision tree    KNN  -0.0634 0.0083 -0.1133 -0.0134   True
Decision tree    SVM  -0.2095  0.001 -0.2598 -0.1592   True
          KNN    SVM  -0.1461  0.001 -0.1964 -0.0958   True
-------------------------------------------------------------
```

Figure 4.4: ANOVA test of 3 Algorithms Measured by F1 score

.05 level (Stats=7.764, p=0.02). The data is feasible for the ANOVA test. The test result of the ANOVA test and posthoc comparisons using Tukey HSD is in Figure 4.4

A one-way between-groups analysis of variance conducted to explore the impact of machine learning algorithms on the performances of classifiers on imbalanced learning problems, as measured by F1 score. Participants contain three groups according to their applied machine learning algorithms (Group 1: Decision Tree Classifier; Group 2: kNN Classifier where k = 5; Group 3: Support Vector Machine Classifier). There was a statistically significant difference at the $p < .05$ level in F1 scores for the three groups of algorithms: F = 50.095, p=1.7e-21. Post-hoc comparisons using the Tukey HSD test (alpha = 0.5) indicated that the mean scores for the 3 Groups are all statistically differ from each other.

From the ANOVA test of the performances, we can conclude the decision tree and the k-NN are more appropriate for imbalanced learning tasks when the classifiers learn from resampled training data. On the other hand, SVM is often unable to present ideal results. Its performances are statistically lower than those of kNN and decision tree, either measured by AUC or by F1 scores.

4.2.2 Comparing the Performances of Different Families of Oversampling Methods on Imbalanced Learning Tasks

This research classifies the ten minority class over-sampling methods tested in this research into five categories.

- Random Oversampling
- SMOTE
- The combination of over-sampling and under-sampling. Methods of this kind tested in this research are SMOTE-TomekLinks and SMOTE-ENN.
- Range Restricted SMOTE extensions. Methods of this kind tested in this research are Borderline-SMOTE (include Borderline SMOTE1 and Borderline SMOTE2, see Section 2.6.3) and Safe-Level SMOTE.
- Clustering Based SMOTE extensions. Methods of this kind tested in this research are Agglomerative Hierarchical Clustering (AHC), Cluster-Based Oversampling (CBSO), and Major Weight Minority Oversampling Technique (MWMOTE).

Bartlett's test conducted to check for homoscedasticity in the dataset. Then to choose to use the parametric approach (ANOVA test) or the non-parametric approach (Kruskal-Wallis test) to compare whether classifiers trained from data over-sampled by them present statistically different results.

Measured by AUC

Figure 4.5 shows the boxplot of the five oversampling methods families. The distribution of their performances shows similar patterns as we could not find any method presents significantly higher or lower results.

Bartlett's test conducted indicates there is homoscedasticity in the groups (Stats=0.526, p=0.971). This research conducts a Kruskal-Wallis test as a non-parametric approach to detect the differences in their distributions.

Kruskal-Wallis test was to explore the impact of the oversampling method, as measured by AUC. For the five groups of oversampling methods. We could not find a statistically significant difference at the $p < .05$ level in AUC for the five groups: Stats=6.066, p=.194.

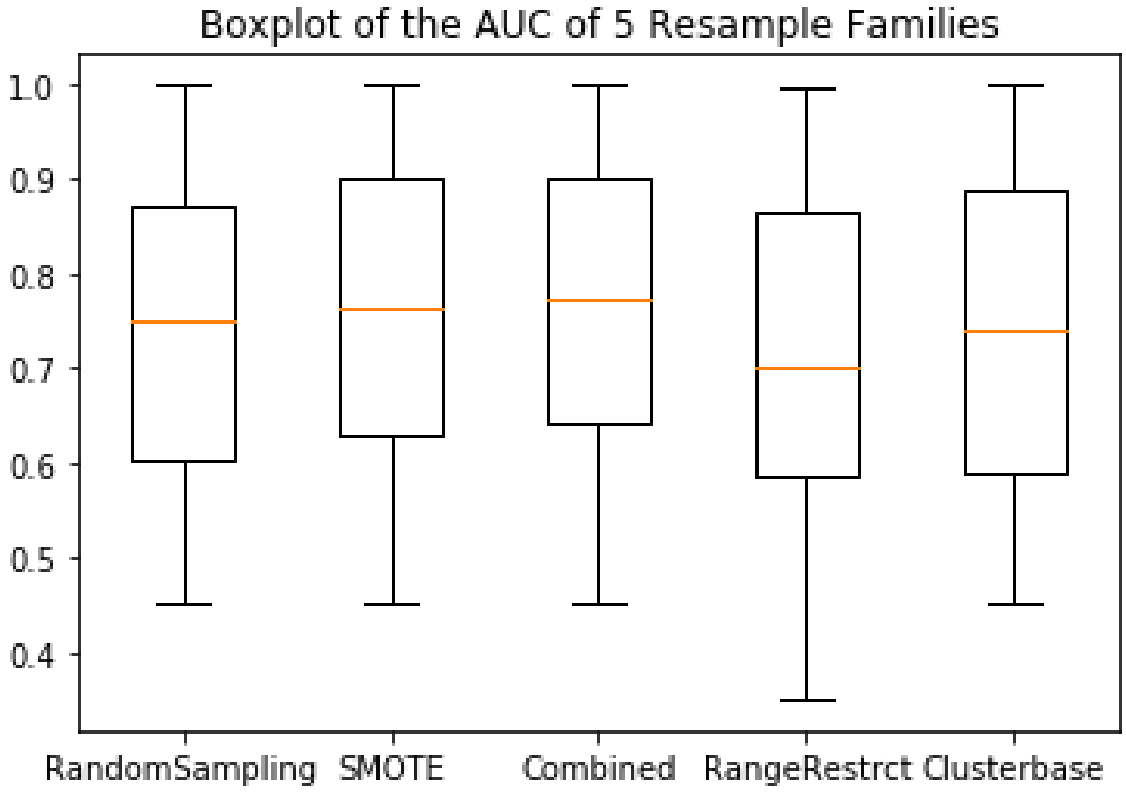


Figure 4.5: AUC Boxplot of 5 Families of Oversampling

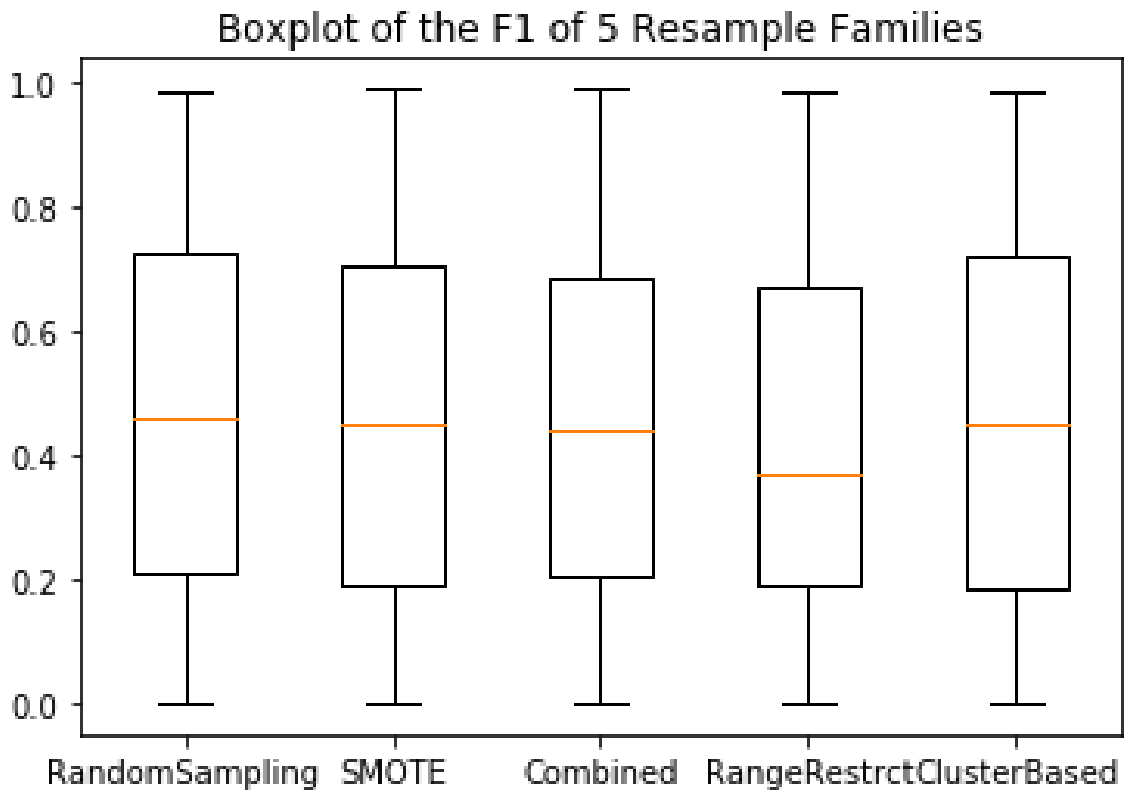


Figure 4.6: F1 Boxplot of 5 Families of Oversampling

Measured by F1 score

Similar to the results of AUC, the performances of different families of oversampling methods, when measured by F1 scores, also could not present any method significantly higher or lower than the others.

Barlett's test indicates the data is suitable for the Kruskal-Wallis test(Stats=0.358, p=0.986)

The result of Kruskal-Wallis test also validates that as the performances of the five families of oversampling methods, as measured by F1 scores do not show a statistically significant difference at the $p < .05$ level in F1 scores for the five groups: Stats=1.743, p=0.783.

As presented in the test results, the complexity of oversampling methods does not contribute to the overall performances of the trained classifier. More straightforward methods such as random oversampling and traditional SMOTE is also possible to

present results as good as or even better than those by methods of more complex such as MWMOTE and CBSO.

In conclusion, we could not find any tested family of the oversampling method is capable of presenting performances statistical significantly better than the others. Thus we do not have the confidence to say which family of the oversampling method is always better despite the kind of imbalanced problems or the distribution of samples.

4.2.3 Comparing the Performances of Oversampling Methods within the Family of Combination Method

Some are the combination of different resampling techniques, as the undersampling process is to reduce noises.

Methods of this kind applied in this research are SMOTE-TomekLinks and SMOTE-ENN, the detail of which are in Section 2.4.

The random oversampling method and the traditional SMOTE is the baseline methods. It is supposed that the tested methods of this section could present better performances than the baseline methods.

It seems SMOTE-ENN and SMOTE-Tomeklinks present slightly higher AUCs than the baseline methods from the boxplot Figure 4.7. Moreover, the average AUC of SMOTE-ENN (AUC = 0.767) is slightly higher than that of SMOTE-TomekLinks (AUC = 0.751). However, the result of ANOVA test shows that the difference is not statistically significant. Bartlett's test shows that they are suitable for the Kruskal-Wallis test(Stats=0.452,p=0.929). A Kruskal-Wallis test conducted to explore the difference in performances of SMOTE-ENN and SMOTE-TomekLinks, whose result shows there was no statistically significant difference at the $p < .05$ level in AUCs of SMOTE-ENN and SMOTE-TomekLinks: Stats=2.098, p=0.552.

The performances measured by F1 score also shows similar patterns, as Figure 4.8 depicts.

Kruskal-Wallis test over the F1 scores of SMOTE-ENN, SMOTE-TomekLinks, SMOTE, and Random Oversampling indicates there was no statistically significant

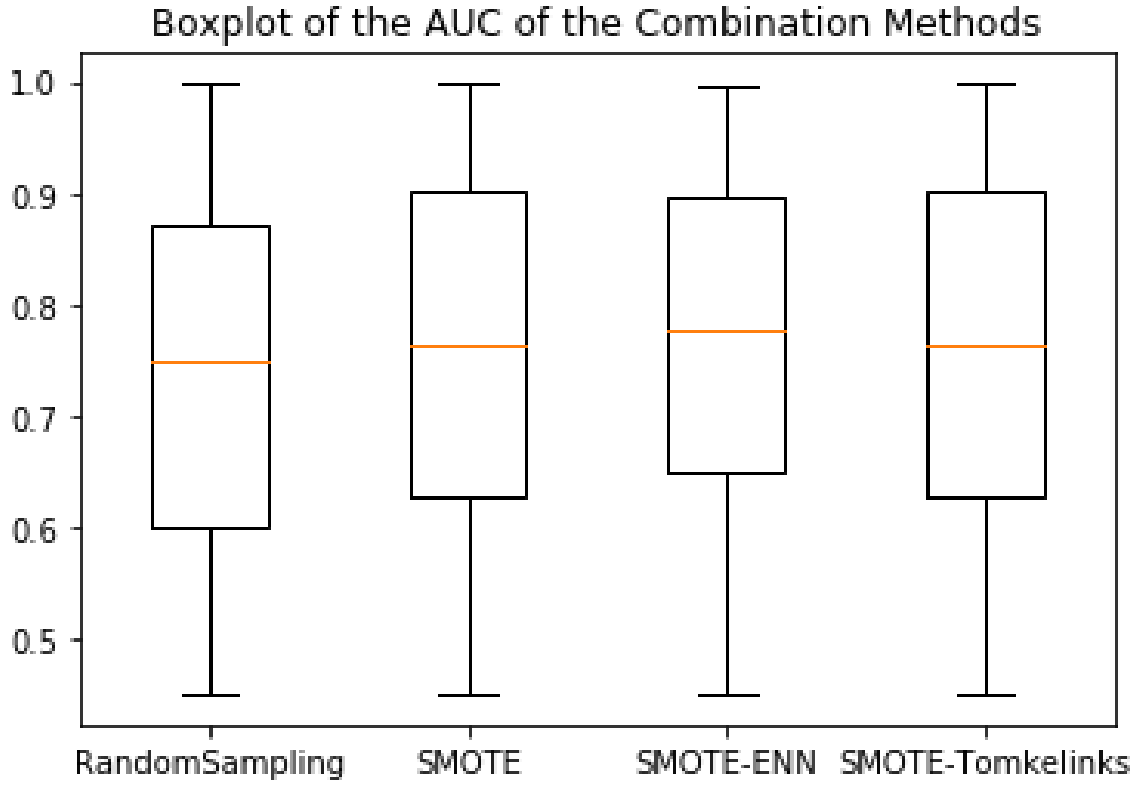


Figure 4.7: AUC Boxplot of Combination of Oversampling and Undersampling

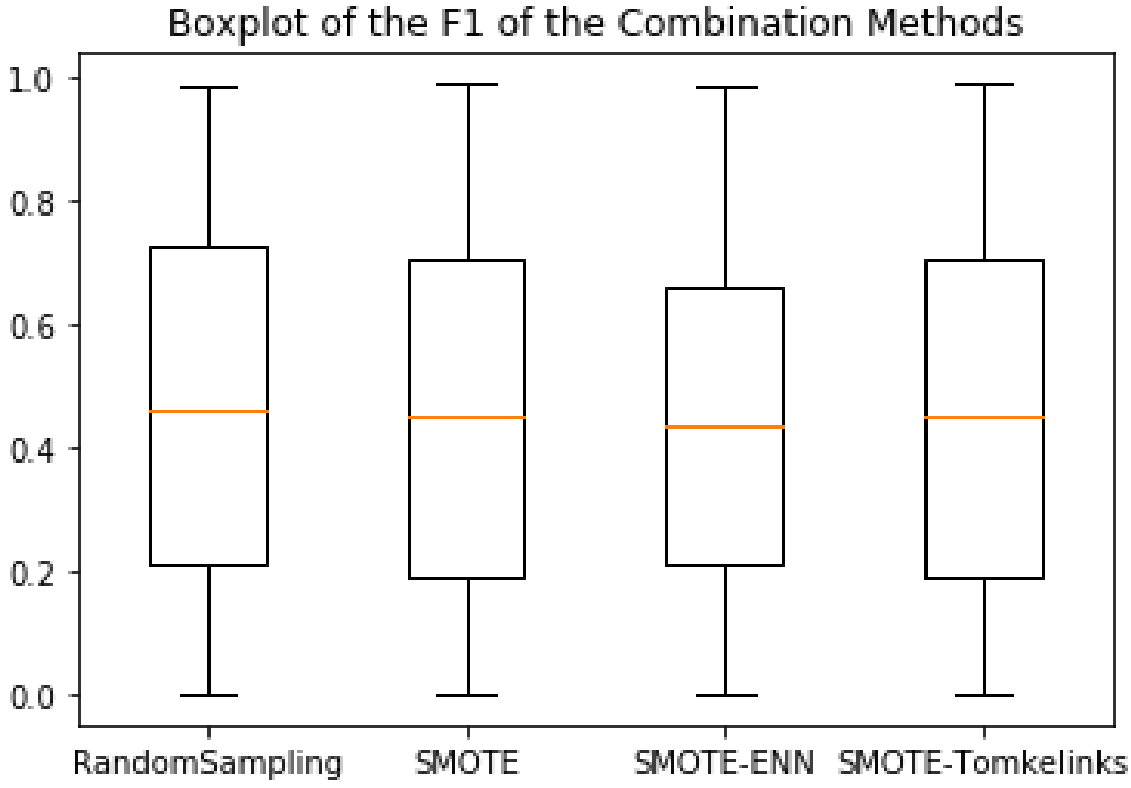


Figure 4.8: F1 Boxplot of Combination of Oversampling and Undersampling

difference at the $p < .05$ level between them: F=0.063, p=0.996. (Bartlett's test: Stats=0.454, p=0.929)

Thus, we could not prove that SMOTE-ENN could always present better results than SMOTE-TomekLinks over any imbalanced problems, or the opposite. Similarly, we could not prove that the combination of multiple resampling technique is always better than Random Oversamplnig and traditional SMOTE.

4.2.4 Comparing the Performances of Oversampling Methods within the Family of Range Restricted SMOTE Extensions

Range Restricted SMOTE extension, covered in the Section 2.6.3, chooses specific subgroups of samples for synthetic sample generation. Normally SMOTE extensions applied range-restricted method prefer to choose those samples close to decision boundaries, which could bring more information in the classifier training process.

The boxplot Figure 4.9 shows the results of 3 tested range-restricted extensions of SMOTE and the two baseline methods, measured by AUC. It shows there is not any method is capable of presenting any significantly better results. Safe Level SMOTE, which is a method extended from Borderline SMOTE, fail to present results as good as baseline methods, surprisingly.

Bartlett's test over the five methods indicates that the Kruskal-Wallis test is more suitable for this occasion (Stats=0.071, p=0.999). The Kruskal-Wallis test shows there is a statistically significant difference at the $p < .05$ level in the five groups of AUCs: Stats=9.623, p=0.047. Post-hoc comparisons using the Tukey HSD test indicates that the mean score of SMOTE (M=0.751, SD=0.158) and Safe Level SMOTE (M=0.689, SD=0.158) is statistically different. No more statistically difference is found in the comparisons between the other methods when measured by AUC.

On the other hand, the performances of the Range Restricted Oversampling methods, when measured by F1 score, show slightly different results.

As Figure 4.11 shows no methods could significantly outperform the rest, and

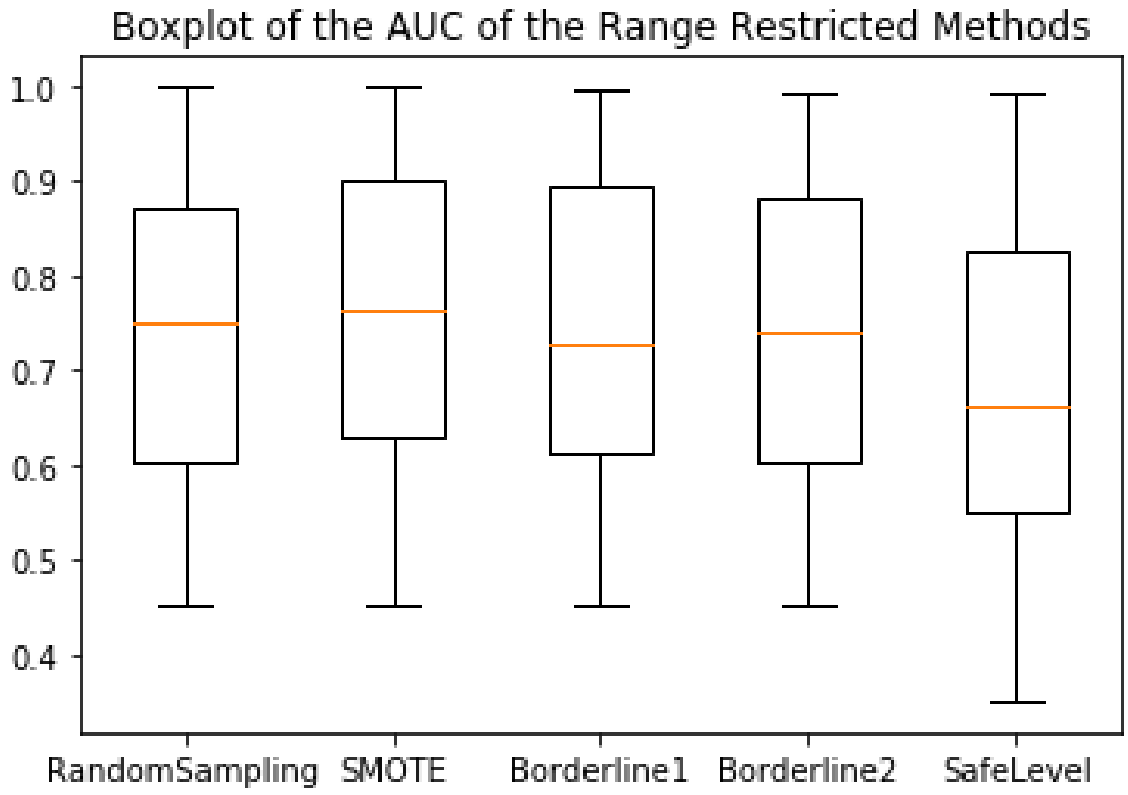


Figure 4.9: Boxplot of AUC of Different Range Restricted Oversampling Methods

Multiple Comparison of Means - Tukey HSD, FWER=0.05

group1	group2	meandiff	p-adj	lower	upper	reject
BorderlineSMOE1	BorderlineSMOE2	-0.0013	0.9	-0.0616	0.059	False
BorderlineSMOE1	SMOTE	0.007	0.9	-0.0533	0.0673	False
BorderlineSMOE1	Safe Level SMOTE	-0.0551	0.0924	-0.1154	0.0053	False
BorderlineSMOE1	random oversampling	-0.009	0.9	-0.0694	0.0513	False
BorderlineSMOE2	SMOTE	0.0083	0.9	-0.052	0.0686	False
BorderlineSMOE2	Safe Level SMOTE	-0.0538	0.1066	-0.1141	0.0066	False
BorderlineSMOE2	random oversampling	-0.0077	0.9	-0.0681	0.0526	False
SMOTE	Safe Level SMOTE	-0.0621	0.0402	-0.1224	-0.0017	True
SMOTE	random oversampling	-0.016	0.9	-0.0764	0.0443	False
Safe Level SMOTE	random oversampling	0.046	0.2265	-0.0143	0.1064	False

Figure 4.10: Tukey HSD test Over the 5 Methods

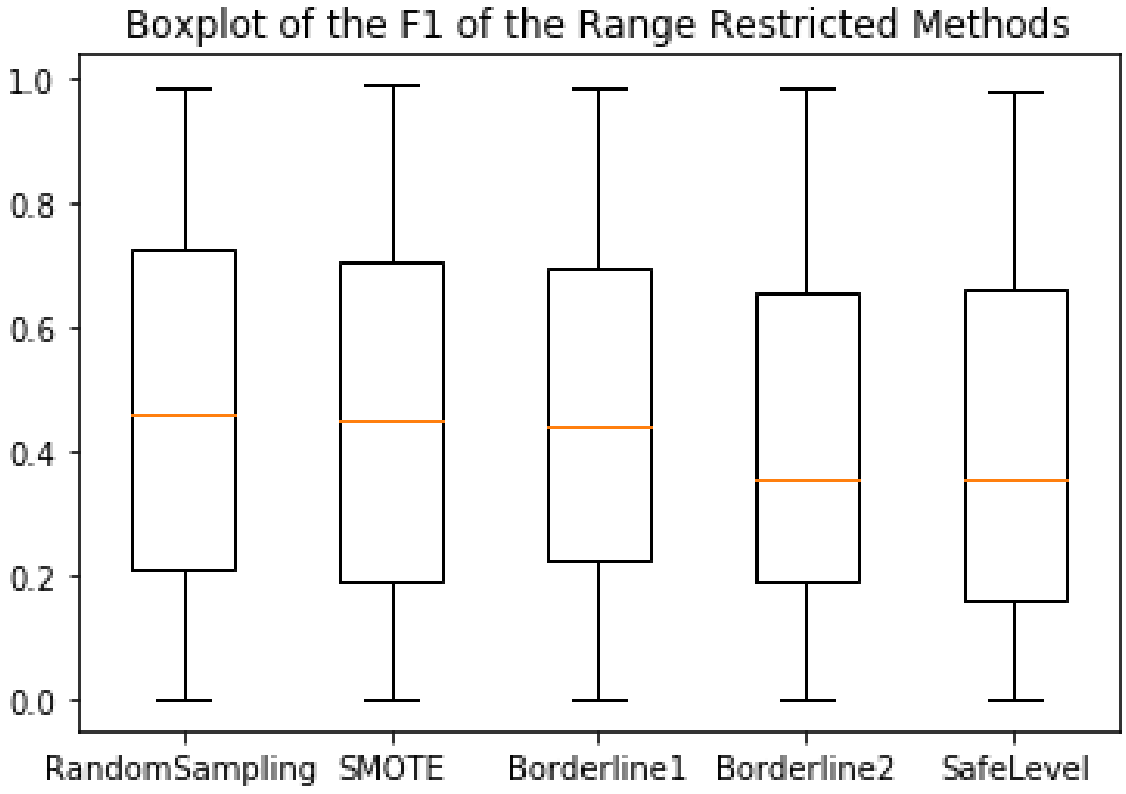


Figure 4.11: Boxplot of F1 of Different Range Restricted Oversampling Methods

Borderline-SMOTE2 drops to the same level of performances as the Safe level SMOTE when measured by F1 score.

Bartlett's test shows that the Kruskal-Wallis test is more feasible (Stats=0.445, p=.979). Kruskal-Wallis test over the F1 scores of the Range Restricted methods shows there is no statistically significant difference at the p <.05 level in the five groups of F1 scores: Stats=2.486, p=0.647.

In summary, we are not confident enough to say SMOTE extensions of range-restricted interpolation method would always present the best result, compare to the baseline methods of traditional SMOTE and random oversampling. Moreover, Safe Level SMOTE fails to prove it could perform reliable results of generating synthetic samples, and thus not recommended for minority class data oversampling.

4.2.5 Comparing the Performances of Oversampling Methods within the Family of Clustering Based SMOTE Extensions

Clustering Based SMOTE, whose definition is covered in Section 2.6.4, as this kind of extensions apply clustering techniques aims to avoid amplifying noises when generating synthetic samples. Methods of this kind applied in this research, which are AHC, CBSO, and MWMOTE, are introduced in Section 2.6.5, Section 2.6.6, and Section 2.6.7, respectively.

Similar to the previous tests, random oversampling and traditional SMOTE is the baseline methods, also involved in performances comparison.

The Figure 4.12 shows results similar to the previous ones: for the clustering based oversampling methods, we can not find any SMOTE extension of this kind is capable of presenting performance significantly better than the others when measured by AUC.

Kruskal-Wallis test over the five methods also indicates that there is no statistically significant difference at the $p < .05$ level in AUCs for the five methods: Stats=5.714, p=0.222. (Bartlett's test result: Stats=2.486, p=0.647).

When measured by F1, the boxplot in Figure 4.13 shows clustering-based SMOTE

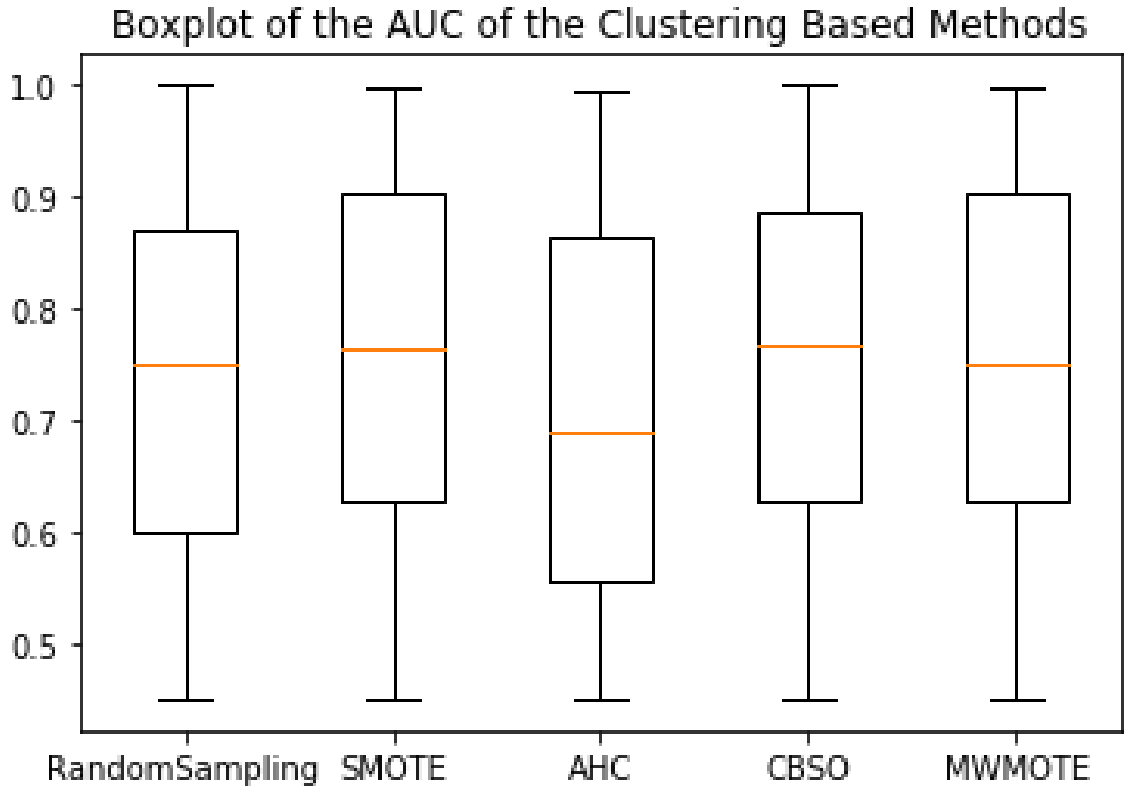


Figure 4.12: Boxplot of AUC of Different Clustering Based Oversampling Methods

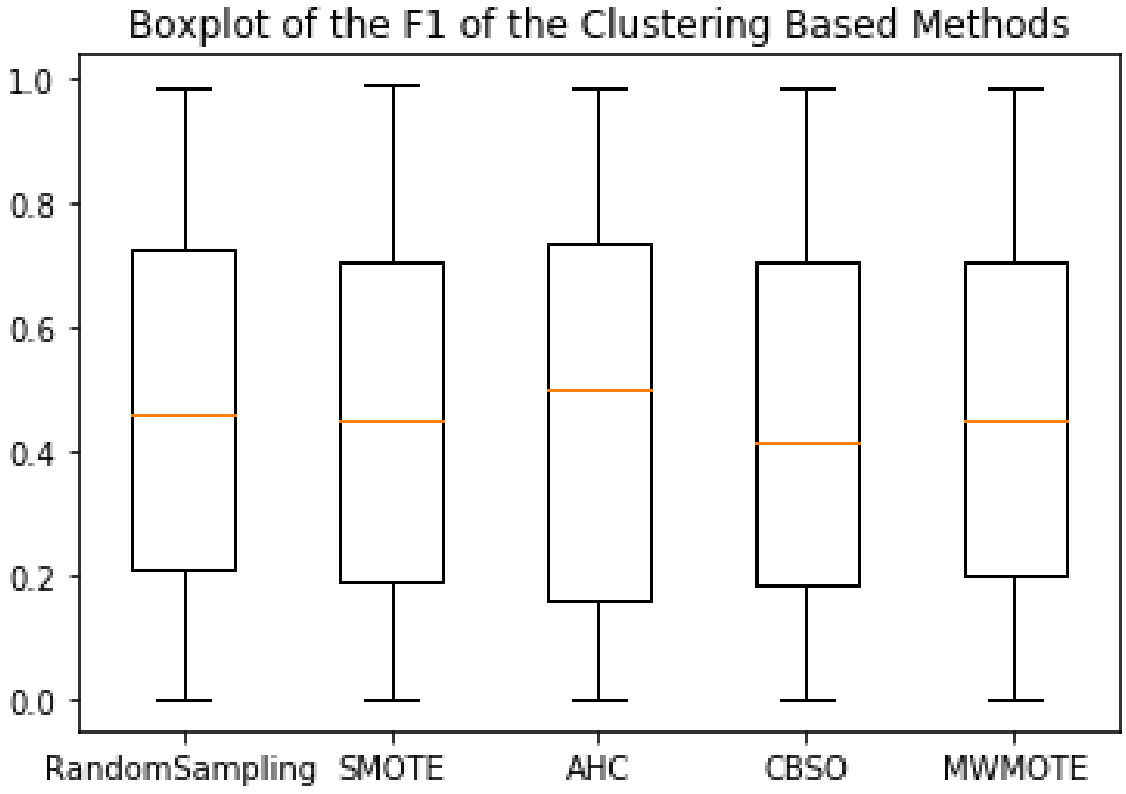


Figure 4.13: Boxplot of F1 of Different Clustering Based Oversampling Methods

extensions could not present significantly better results than it is of the baseline methods, either measured in AUC or an F1 score. The ANOVA test conducted over the F1 scores of the clustering-based methods indicates that there is no statistically significant difference at the $p < .05$ level in F1 scores for the five methods: Stats=0.045, p=0.996. (Bartlett's test result: Stats=0.849, p=0.932).

4.2.6 Statistic Test Conclusion

According to the statistic tests over the processes of imbalanced learning, decision tree and k-NN are more recommended as the machine learning algorithms tested in this research, compare to support vector machine, which presents statistical significantly lower results than the others. However, we do not have strong enough evidence to prove any minority data oversampling method is always the most appropriate for any imbalanced learning problem.

4.3 Other Findings

Though there is not any resampling method is always the best, this research still explores how different resampling methods act differently regards to datasets.

4.3.1 Comparing Performances on Unstructured Data

The 35 datasets selected in this research are mostly structured dataset, except the dataset *'optical digits'* and *'webpage'*. *'Optical digits'* is a dataset that contains handwritten digit images. *'Webpage'* is a dataset contains one-hot encoded texts. Though the experiment only conducted on a small number of unstructured datasets and the result is not statistically convincing. The experiment result is still a reference for choosing data resampling techniques on unstructured data learning tasks.

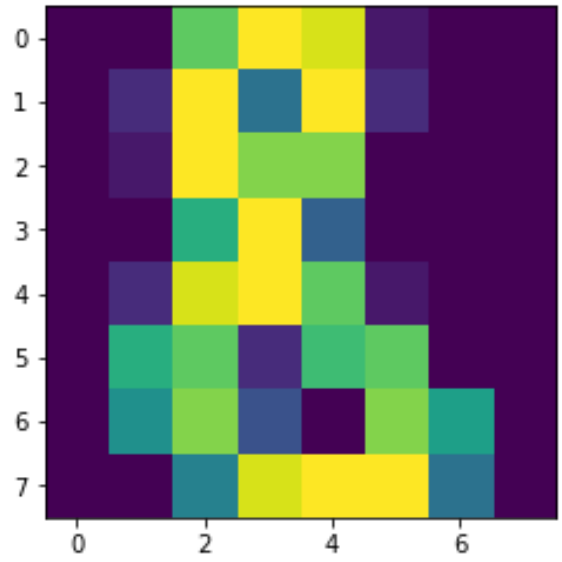


Figure 4.14: An Instance of Optical Digit Dataset

Image Data

The *'optical digits'* dataset contains 8×8 pixels black and white images, the digit eight is the minority class for this research. An Instance of this dataset is in Figure 4.14.

For this dataset, the performances of tested data oversampling methods, on three machine learning algorithms are in Figure 4.15. From the bar chart, we can find that k-NN generally produces the best result among the three algorithms, and SVM produces the worst results. When focuses on the performances of all the data resampling methods on decision tree and support vector machine, the tested families of oversampling methods produce excellent results on all of them. At the same time, range-restricted SMOTE extensions are incapable of producing any good results. From this dataset, we can conclude that traditional SMOTE and clustering-based SMOTE are the most appropriate methods for minority class oversampling, and k-NN is the most recommended algorithm for this kind of tasks. However, this experiment conducted on a low-resolution image dataset. Whether SMOTE and its extensions are still valuable on high-resolution image dataset is worth further researching. Furthermore, the features generated by convolutional layers can be useful for minority class sample generation.

Text Data

The dataset *'webpage'* is an one-hot encoded text dataset. The result performances of tested minority class oversampling methods are in Figure 4.16. The findings from the bar charts are decision tree classifiers generally produce the best results, and clustering-based SMOTE extensions(AHC, CBSO, and MWMOTE) produce the best results, among other methods, when the machine learning algorithm is a decision tree. Moreover, the method which combines different resampling techniques(SMOTE-TomekLinks and SMOTE-ENN), and range-restricted SMOTE extensions (Borderline-SMOTE and Safe-Level SMOTE) could not present good results, no matter what the training algorithm is.

In conclusion, this research can not identify any resampling method is the best for the one-hot encoded text classification task. However, it can recommend decision tree classifier and clustering-based oversampling methods as this combination produces a good result on this selected text classification task.

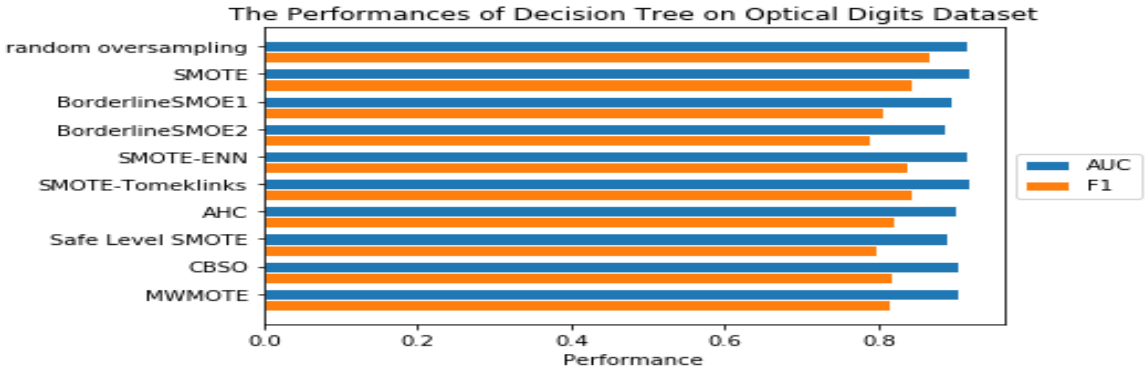
4.4 Strength & Limitations of Findings

4.4.1 Strength of Findings

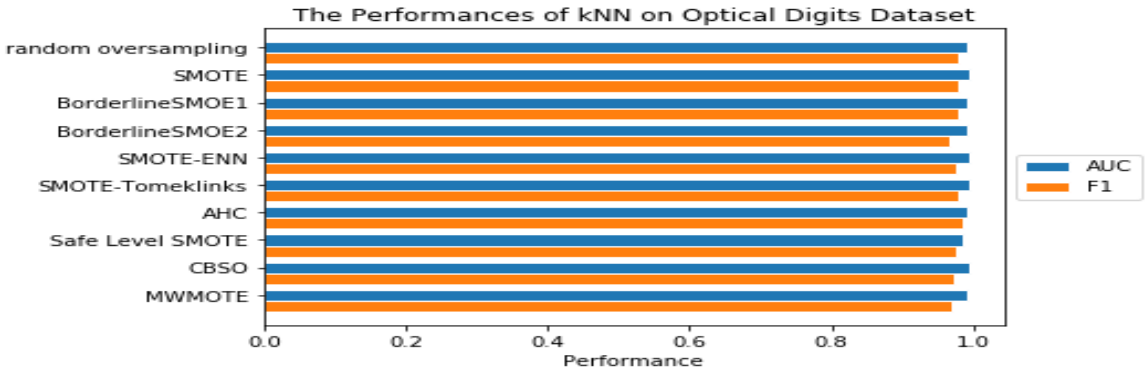
The main strength of the findings of this research is that a minimal number of researches before compared oversampling methods on a large number of datasets and test the statistically significant of their performances. This dissertation tests ten oversampling methods and uses three machine learning algorithms to ensure the research covers enough varieties in real-life application. Moreover, this research experiments on 35 datasets, range from medical datasets to financial and pattern recognition, make sure the variety of the selected datasets and produce a statistically significant result. Another strength of findings for this dissertation is that it discusses oversampling methods could have different performances on different kinds of datasets. The result can be a useful reference for choosing suitable techniques to oversample data of minority class to produce better results in imbalanced learning.

4.4.2 Limitation of Findings

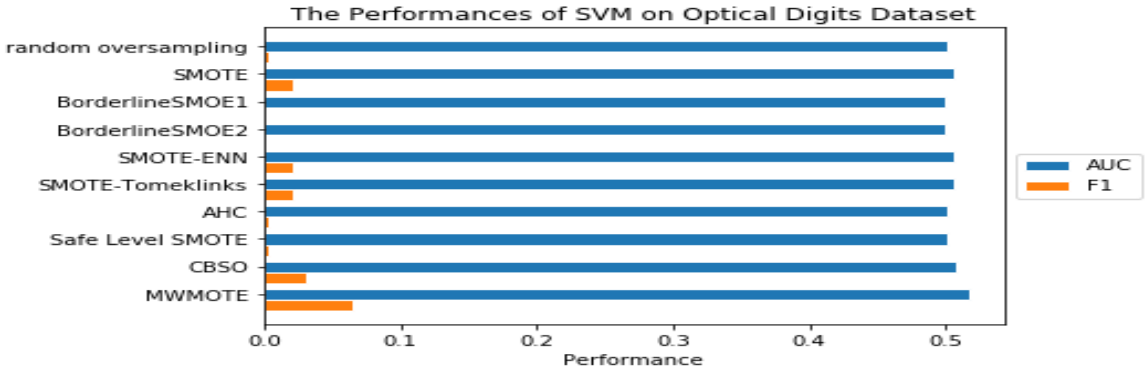
As for the main limitation, this research does not discuss the relations between the data distribution and the oversampling process in depth. This research could not give good recommendations for this task as the result of this dissertation is from a statistic approach.



(a) Performances of Decision Tree on Optical Digits

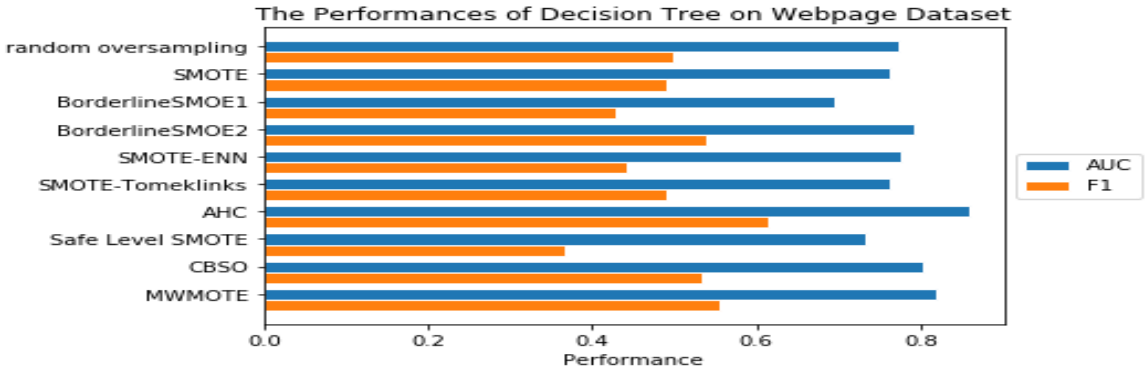


(b) Performances of k-NN on Optical Digits

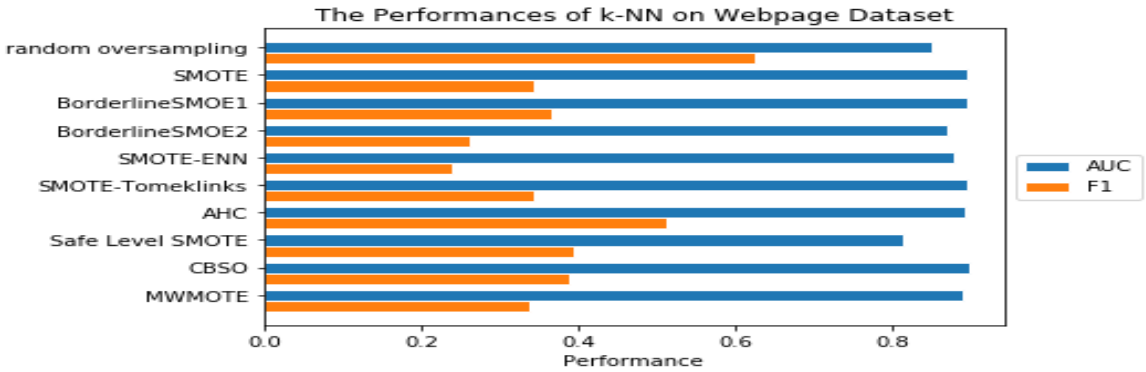


(c) Performances of SVM on Optical Digits

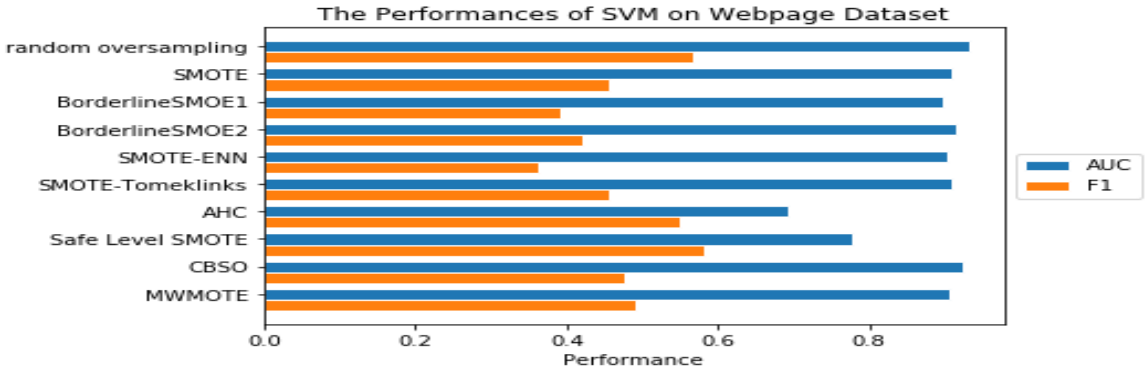
Figure 4.15: Performance Comparison on Dataset Optical Digits



(a) Performances of Decision Tree on Webpage



(b) Performances of k-NN on Webpage



(c) Performances of SVM on Webpage

Figure 4.16: Performance Comparison on Dataset Webpage

Chapter 5

Conclusion

This chapter provides conclusions inferred from this body of work. It briefly touches on the research overview, problem definition, experiment design, evaluation and results as discussed in the previous chapters.

At last, it discusses the contributions and impact of the experiment conducted in this work while also pointing towards any future work and recommendations for further studies in this domain.

5.1 Research Overview

This research tests ten oversampling methods: Random Oversampling, SMOTE, SMOTE-ENN, SMOTE-TomekLinks, Borderline-SMOTE1, Borderline-SMOTE2, Safe-Level SMOTE, AHC, CBSO, and MWMOTE. The ten methods are of five categories based on reviewed literature. They are methods of combination techniques (SMOTE-ENN and SMOTE-TomekLinks), range-restricted SMOTE extensions (Borderline-SMOTE1, Borderline-SMOTE2, and Safe-Level SMOTE), clustering-based SMOTE extensions (AHC, CBSO, and MWMOTE), and two methods as the baseline methods (Random oversampling and SMOTE). Three machine learning algorithms trained by the data oversampled by ten resampling methods applied to 35 datasets. Then this research conduct statistic tests to find the statistically significant relations in the result. The result indicates that the performances of SVM is statistically significantly lower than

the others, and the ten oversampling methods do not show statistically significant differences.

5.2 Problem Definition

The research question is *'Is there a statistically significant difference in the performances of the combination of data resampling method, range-restricted SMOTE extensions, and clustering-based SMOTE extensions, when measured by F1 and AUC?'*

The primary purpose of the research was to establish the validity of the following hypotheses:

Null hypothesis: There is no statistically significant difference in the performances of the combination of data resampling method, range-restricted SMOTE extensions, and clustering-based SMOTE extensions when measured by F1 and AUC.

Alternative hypothesis: There is a statistically significant difference in the performances of the combination of data resampling method, range-restricted SMOTE extensions, and clustering-based SMOTE extensions when measured by F1 and AUC.

The research focused on exploring the three classification algorithms, coordinate with ten oversampling methods, on 35 imbalanced datasets to test their performances on imbalanced learning problem. The performance measurement metrics are AUC, F1, Classification accuracy, and G-mean.

5.3 Design/Experimentation, Evaluation & Results

The design of the experiment was sound and well-grounded as it was on a large number of datasets. There are thirty-five imbalanced datasets selected for testing, with their imbalanced ratio range from 1.2 to 130. This research selects ten oversampling methods and classified them into three categories and two baseline methods based on reviewed literature. Ten oversampling methods and three machine learning algorithms, which are decision tree, k-Nearest Neighbours, and Support Vector Machine are applied. In this way, we conducted $35 \times 3 \times 10 = 1050$ experiments, and the measurements of the

results are classification accuracy, AUC, F1, and G-mean.

Then this research conducted ANOVA test Kruskal-Wallis test to explore the statistical significantly relations in the experiment results. The selection of the test is by whether the distribution of data meets the assumption of ANOVA. Based on the result of statistic tests, the performances of SVM on imbalanced data are statistically significantly lower than those of decision tree and k-Nearest Neighbours. However, this research did not find the tested oversampling methods are performing a statistically significant difference.

5.4 Contributions and impact

The significant contribution of this research is that it compares different oversampling methods in a large number of datasets, which brings a more statistical result of comparing, compared to the literature of other resampling methods.

The innovation of this work is that it thoroughly reviewed and compared state-of-the-art SMOTE extensions. Furthermore, it discusses the performances of different oversampling methods on imbalanced learning tasks on unstructured data.

5.5 Future Work & recommendations

This research looks at the performances of 10 oversampling methods on 35 imbalanced datasets. However, there was only a small amount of works conducted to discuss how and why oversampling methods act differently on imbalanced datasets with different distribution patterns.

Further work could be on exploring the relations between the distribution patterns of imbalance datasets and the data resampling methods, which can provide more specific recommendations on choosing the appropriate data resampling method.

Another area of future work is to research on how to apply SMOTE and its extensions to more complex unstructured data, such as high-resolution image data and long text data. SMOTE on the feature space of complex unstructured data may be

incapable of producing helpful instances, as using pixels or single words for oversampling is too superficial for complex data. Thus, SMOTE combines with Convolutional Neural Networks, and Recurrent Neural Networks is highly potential to produce good results on imbalanced learning tasks of image recognition and text classification.

References

Abdi, L., & Hashemi, S. (2016). To combat multi-class imbalanced problems by means of over-sampling techniques. *IEEE Transactions on Knowledge and Data Engineering*, *28*(1), 238-251. doi: 10.1109/TKDE.2015.2458858

Alejo, R., García, V., & Pacheco, J. (2015). An efficient over-sampling approach based on mean square error back-propagation for dealing with the multi-class imbalance problem. *Neural Processing Letters*, *42*, 603-617. doi: 10.1007/s11063-014-9376-3

Ali, A., Shamsuddin, S. M., & Ralescu, A. (2015). Classification with class imbalance problem: A review. , *7*, 176-204.

Alimoglu, F. (1996). *Combining multiple classifiers for pen-based handwritten digit recognition* (Unpublished master's thesis). Institute of Graduate Studies in Science and Engineering, Bogazici University.

Alimoglu, F., & Alpaydin, E. (1997). Combining multiple representations and classifiers for pen-based handwritten digit recognition. In *Proceedings of the fourth international conference on document analysis and recognition* (Vol. 2, p. 637-640 vol.2). doi: 10.1109/ICDAR.1997.620583

Alpaydin, E., & Kaynak, C. (1998). Cascaded classifiers. *Kybernetika*, *34*, 369–374.

Angiulli, F. (2005). Fast condensed nearest neighbor rule. In (p. 25-32). doi: 10.1145/1102351.1102355

Barua, S., Islam, M., & Murase, K. (2013). Prowsyn: Proximity weighted synthetic oversampling technique for imbalanced data set learning. In (p. 317-328). doi: 10 .1007/978-3-642-37456-2_27

Barua, S., Islam, M. M., & Murase, K. (2011). A novel synthetic minority oversampling technique for imbalanced data set learning. In *Proceedings of the 18th international conference on neural information processing - volume part ii* (pp. 735–744). Berlin, Heidelberg: Springer-Verlag. Retrieved from `http://dx.doi.org/10.1007/978-3-642-24958-7_85` doi: 10.1007/978-3-642-24958-7_85

Barua, S., Islam, M. 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*(2), 405-425. doi: 10.1109/TKDE.2012.232

Batista, G., Bazzan, A., & Monard, M.-C. (2003). Balancing training data for automated annotation of keywords: a case study. In (p. 10-18).

Batista, G., Prati, R., & Monard, M.-C. (2004). A study of the behavior of several methods for balancing machine learning training data. *SIGKDD Explorations*, *6*, 20-29. doi: 10.1145/1007730.1007735

Bunkhumpornpat, C., Sinapiromsaran, K., & Lursinsap, C. (2009). Safe-level-smote: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem. In (Vol. 5476, p. 475-482). doi: 10.1007/978-3-642-01307-2_43

Bunkhumpornpat, C., Sinapiromsaran, K., & Lursinsap, C. (2011). Dbsmote: Density-based synthetic minority over-sampling technique. *Applied Intelligence - APIN*, *36*. doi: 10.1007/s10489-011-0287-y

Bunkhumpornpat, C., & Subpaiboonkit, S. (2013). Safe level graph for synthetic minority over-sampling techniques. In *2013 13th international symposium on communications and information technologies (iscit)* (p. 570-575). doi: 10.1109/ISCIT.2013.6645923

Błaszczyński, J., Deckert, M., Stefanowski, J., & Wilk, S. (2012). Iivotes ensemble for imbalanced data. *Intelligent Data Analysis*, *16*, 777-801. doi: 10.3233/IDA-2012 -0551

Cateni, S., Colla, V., & Vannucci, M. (2011). Novel resampling method for the classification of imbalanced datasets for industrial and other real-world problems. In *2011 11th international conference on intelligent systems design and applications* (p. 402-407). doi: 10.1109/ISDA.2011.6121689

Chan, P. K., & Stolfo, S. J. (1998). Toward scalable learning with non-uniform class and cost distributions: A case study in credit card fraud detection. *KDD*.

Chawla, N., Bowyer, K., Hall, L., & Kegelmeyer, W. (2002). Smote: Synthetic minority over-sampling technique. *J. Artif. Intell. Res. (JAIR)*, *16*, 321-357. doi: 10.1613/jair.953

Chen, S., He, H., & Garcia, E. A. (2010). Ramoboost: Ranked minority oversampling in boosting. *IEEE Transactions on Neural Networks*, *21*(10), 1624-1642. doi: 10 .1109/TNN.2010.2066988

Cohen, G., Hilario, M., Sax, H., Hugonnet, S., & Geissbuhler, A. (2006). Learning from imbalanced data in surveillance of nosocomial infection. *Artificial intelligence in medicine*, *37*, 7-18. doi: 10.1016/j.artmed.2005.03.002

Cortez, P., Cerdeira, A., Almeida, F., Matos, T., & Reis, J. (1998). Modeling wine preferences by data mining from physicochemical properties. *Decision Support Systems*, *47*(4), 547-553.

Dang, X. T., Tran, D. H., Hirose, O., & Satou, K. (2015). Spy: A novel resampling method for improving classification performance in imbalanced data. In *2015 seventh international conference on knowledge and systems engineering (kse)* (p. 280-285). doi: 10.1109/KSE.2015.24

Douzas, G., & Bação, F. (2017). Self-organizing map oversampling (somo) for imbalanced data set learning. *Expert Systems with Applications*, *82*, 40-52. doi: 10.1016/j.eswa.2017.03.073

Elkan, C. (2001). The foundations of cost-sensitive learning. In *Proceedings of the 17th international joint conference on artificial intelligence - volume 2* (pp. 973–978). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. Retrieved from `http://dl.acm.org/citation.cfm?id=1642194.1642224`

Elrahman, S. M. A., & Abraham, A. (2014). A review of class imbalance problem..

Estabrooks, A., Jo, D. T., & Japkowicz, N. (2004). A multiple resampling method for learning from imbalanced data sets. *Computational Intelligence*, *20*, 18-36. doi: 10.1111/j.0824-7935.2004.t01-1-00228.x

An experiment with the edited nearest-neighbor rule. (1976). *IEEE Transactions on Systems, Man, and Cybernetics*, *SMC-6*(6), 448-452. doi: 10.1109/TSMC.1976.4309523

Fernández, A., Garcia, S., Herrera, F., & Chawla, N. (2018). Smote for learning from imbalanced data: Progress and challenges, marking the 15-year anniversary. *Journal of Artificial Intelligence Research*, *61*, 863-905. doi: 10.1613/jair.1.11192

Gao, M., Hong, X., Chen, S., Harris, C., & Khalaf, E. (2014). Pdfos: Pdf estimation based over-sampling for imbalanced two-class problems. *Neurocomputing*, *138*, 248–259. doi: 10.1016/j.neucom.2014.02.006

Gazzah, S., & ESSOUKRI BEN AMARA, N. (2008). New oversampling approaches based on polynomial fitting for imbalanced data sets. In (p. 677-684). doi: 10.1109/DAS.2008.74

Gazzah, S., Hechkel, A., & Essoukri Ben Amara, N. (2015). A hybrid sampling method for imbalanced data. In *2015 ieee 12th international multi-conference on systems, signals devices (ssd15)* (p. 1-6). doi: 10.1109/SSD.2015.7348093

Gong, C., & Gu, L. (2016). A novel smote-based classification approach to online data imbalance problem. *Mathematical Problems in Engineering*, *2016*, 1-14. doi: 10.1155/2016/5685970

Gu, Q., Cai, Z., & Zhu, L. (2009). Classification of imbalanced data sets by using the hybrid re-sampling algorithm based on isomap. In (Vol. 5821, p. 287-296). doi: 10.1007/978-3-642-04843-2_31

Haibo He, Yang Bai, Garcia, E. A., & Shutao Li. (2008). Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In *2008 ieee international joint conference on neural networks (ieee world congress on computational intelligence)* (p. 1322-1328). doi: 10.1109/IJCNN.2008.4633969

Han, H., Wang, W.-Y., & Mao, B.-H. (2005). Borderline-smote: A new over-sampling method in imbalanced data sets learning. In (Vol. 3644, p. 878-887). doi: 10.1007/11538059_91

Hart, P. (1968). The condensed nearest neighbor rule (corresp.). *IEEE Transactions on Information Theory*, *14*(3), 515-516. doi: 10.1109/TIT.1968.1054155

He, H., & Shen, X. (2007). A ranked subspace learning method for gene expression data classification. In (Vol. 1, p. 358-364).

Horton, P., & Nakai, K. (1996). A probabilistic classification system for predicting the cellular localization sites of proteins. In *Proceedings of the fourth international conference on intelligent systems for molecular biology* (pp. 109–115). AAAI Press. Retrieved from `http://dl.acm.org/citation.cfm?id=645631.662879`

Koto, F. (2014). Smote-out, smote-cosine, and selected-smote: An enhancement strategy to handle imbalance in data level.. doi: 10.13140/2.1.3533.0887

Kruskal, W. H., & Wallis, W. A. (1952). Use of ranks in one-criterion variance analysis. *Journal of the American Statistical Association*, *47*(260), 583-621. Retrieved from `https://www.tandfonline.com/doi/abs/10.1080/01621459.1952.10483441` doi: 10.1080/01621459.1952.10483441

Kubat, M. (2000). Addressing the curse of imbalanced training sets: One-sided selection. *Fourteenth International Conference on Machine Learning*.

Kubat, M., Holte, R. C., & Matwin, S. (1998, 01). Machine learning for the detection of oil spills in satellite radar images. *Machine Learning*, *30*(2), 195–215. Retrieved from `https://doi.org/10.1023/A:1007452223027` doi: 10.1023/A:1007452223027

Kukar, M., & Kononenko, I. (1998). Cost-sensitive learning with neural networks. In *Proceedings of the 13th european conference on artificial intelligence (ecai-98* (pp. 445–449). John Wiley Sons.

Last, F., Douzas, G., & Bação, F. (2017). Oversampling for imbalanced learning based on k-means and smote. *ArXiv*, *abs/1711.00837*.

Laurikkala, J. (2001). Improving identification of difficult small classes by balancing class distribution. In (p. 63-66). doi: 10.1007/3-540-48229-6_9

Lemaître, G., Nogueira, F., & Aridas, C. K. (2017). Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *Journal of Machine Learning Research*, *18*(17), 1-5. Retrieved from `http://jmlr.org/papers/v18/16-365.html`

Leonard, D. (2019). Week 10: Clustering.

Liang, T., Xu, X., & Xiao, P. (2017). A new image classification method based on modified condensed nearest neighbor and convolutional neural networks. *Pattern Recognition Letters*. doi: 10.1016/j.patrec.2017.05.019

Ma, L., & FAN, S. (2017). Cure-smote algorithm and hybrid algorithm for feature selection and parameter optimization based on random forests. *BMC Bioinformatics*, *18*. doi: 10.1186/s12859-017-1578-z

Maciejewski, T., & Stefanowski, J. (2011). Local neighbourhood extension of smote for mining imbalanced data. In *2011 ieee symposium on computational intelligence and data mining (cidm)* (p. 104-111). doi: 10.1109/CIDM.2011.5949434

Maheshwari, S., Jain, D. R., & Jadon, D. R. S. (2017). A review on class imbalance problem : Analysis and potential solutions..

Maloof, M. (2003). Learning when data sets are imbalanced and when costs are unequal and unknown. *Analysis*, *21*.

Mazurowskia, M. A., Habasa, P. A., Zuradaa, J. M., Lob, J. Y., Bakerb, J. A., & Tourassi, G. D. (2008). Training neural network classifiers for medical decision making: The effects of imbalanced datasets on classification performance. *Neural Networks*, *21*, 427–436. doi: doi.org/10.1016/j.neunet.2007.12.031

Mease, D., Wyner, A., & Buja, A. (2007). Boosted classification trees and class probability / quantile estimation. *Journal of Machine Learning Research*, *8*, 409-439.

Moradi, S., & Rafiei, F. M. (2019). A dynamic credit risk assessment model with data mining techniques: evidence from iranian banks. *Financial Innovation*, *5*, 1-27. doi: https://doi.org/10.1186/s40854-019-0121-9

Nekooeimehr, I., & Lai-Yuen, S. (2015). Adaptive semi-unsupervised weighted over-sampling (a-suwo) for imbalanced datasets. *Expert Systems with Applications*, *46*. doi: 10.1016/j.eswa.2015.10.031

Pearson, R. K., Gonye, G. E., & Schwaber, J. S. (2003). Imbalanced clustering of microarray time-series. *Workshop on Learning from Imbalanced Dataset II, ICML.*

Prati, R., Batista, G., & Monard, M.-C. (2004). Learning with class skews and small disjuncts. In (p. 296-306). doi: 10.1007/978-3-540-28645-5_30

Pérez-Ortiz, M., Gutiérrez, P. A., & Martínez, C. (2013). Borderline kernel based over-sampling. In (Vol. 8073, p. 472-481). doi: 10.1007/978-3-642-40846-5_47

Qorry Meidianingsih, Erfiani, & Bagus Sartono. (2017). The study of safe-level smote method in unbalanced data classification. *Proceedings of the 2017 International Journal of Scientific Engineering Research, IJSER 2017*, *8*, 1167-1171.

Rivera, W. (2017). Noise reduction a priori synthetic over-sampling for class imbalanced data sets. *Information Sciences*, *408*, 146-161. doi: 10.1016/j.ins.2017.04.046

Rong, T., Gong, H., & Ng, W. (2014). Stochastic sensitivity oversampling technique for imbalanced data. In (Vol. 481, p. 161-171). doi: 10.1007/978-3-662-45652-1_18

Sandhan, T., & Choi, J. Y. (2014). Handling imbalanced datasets by partially guided hybrid sampling for pattern recognition. In *2014 22nd international conference on pattern recognition* (p. 1449-1453). doi: 10.1109/ICPR.2014.258

Siddappa, N., & Kampalappa, T. (2019). Adaptive condensed nearest neighbor for imbalance data classification..

Sun, A., peng Lim, E., & Liu, Y. (2009). On strategies for imbalanced text classification using svm: A comparative study. *Decision Support System*, *48*, 191-201. doi: doi:10.1016/j.dss.2009.07.011

Sun, Y., Kamel, M. S., & Wang, Y. (2006). Boosting for learning multiple classes with imbalanced class distribution. In *Sixth international conference on data mining (icdm'06)* (p. 592-602). doi: 10.1109/ICDM.2006.29

Sánchez, A., Morales, E., & Gonzalez, J. (2013). Synthetic oversampling of instances using clustering. *International Journal of Artificial Intelligence Tools*, *22*. doi: 10 .1142/S0218213013500085

Torres, F., Carrasco-Ochoa, J., & Martínez-Trinidad, J. F. (2016). Smote-d a deterministic version of smote. In (p. 177-188). doi: 10.1007/978-3-319-39393-3_18

Two modifications of cnn. (1976). *IEEE Transactions on Systems, Man, and Cybernetics*, *SMC-6*(11), 769-772. doi: 10.1109/TSMC.1976.4309452

Understanding auc - roc curve. (n.d.). `https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5`. (Accessed: 2019-12-01)

Wang, J., Yao, Y., Zhou, H., Leng, M., & Chen, X. (2013). A new over-sampling technique based on svm for imbalanced diseases data. In (p. 1224-1228). doi: 10.1109/MEC.2013.6885254

Wang, J., Yun, B., Huang, P., & Liu, Y.-A. (2013). Applying threshold smote algoritwith attribute bagging to imbalanced datasets. In (Vol. 8171, p. 221-228). doi: 10.1007/978-3-642-41299-8_21

Weiss, G., & Provost, F. (2001). The effect of class distribution on classifier learning: An empirical study. *Tech Rep*.

Wilson, D. L. (1972). Asymptotic properties of nearest neighbor rules using edited data. *IEEE Transactions on Systems, Man, and Cybernetics*, *SMC-2*(3), 408-421. doi: 10.1109/TSMC.1972.4309137

Xie, Z., Jiang, L., Ye, T., & li, X. (2015). A synthetic minority oversampling method based on local densities in low-dimensional space for imbalanced learning. In (Vol. 9050). doi: 10.1007/978-3-319-18123-3_1

Xu, L., Krzyzak, A., & Suen, C. (1992). Methods of combining multiple classifiers and their applications to handwriting recognition. *IEEE Transactions on Systems Man and Cybernetics Part B (Cybernetics)*, *22*, 418 - 435. doi: 10.1109/21.155943

Zhang, J., & Mani, I. (2003). KNN Approach to Unbalanced Data Distributions: A Case Study Involving Information Extraction. In *Proceedings of the ICML'2003 Workshop on Learning from Imbalanced Datasets.*

Zhi-Hua Zhou, & Xu-Ying Liu. (2006). Training cost-sensitive neural networks with methods addressing the class imbalance problem. *IEEE Transactions on Knowledge and Data Engineering*, *18*(1), 63-77. doi: 10.1109/TKDE.2006.17

Appendix A

Additional content

Table A.1: Selected Datasets from the Reviewed Literatures

Ref.	Algorithm	Datasets	Measure Metrics
(Chawla et al., 2002)	SMOTE	Pima, Phoneme, Adult, E-state, Satimage, Forest Cover, Oil, Mammography, Can	AUC, Acc
(Cohen et al., 2006)	AHC-SMOTE	nosocomially infected patients dataset	AUC, Sensitivity, Specificity, CWA, Acc
(Barua et al., 2011)	CBSO	Abalone, Vehicle, Glass, Wine, Texture, Pima, Ionosphere, Spambase	Acc, F1, G-mean
(Douzas & Bação, 2017)	SOMO	Breast, Ecoli, Glass, Haberman, Heart, Eucalyptus, Iris, Libra, Liver, Pima, Segment, vehicle, Wine	AUC, F1, G-mean

Continued on next page

Table A.1 – continued from previous page

Ref.	Algorithm	Datasets	Measure Metrics
(Last et al., 2017)	K-Means SMOTE	Breast, Ecoli, Glass, Haberman, Heart, Iris, Libra, Liver, Pima, Segement, Vehicle, Wine, thyroid, vowel, yeast	AUC, G-mean, F1
(Xie et al., 2015)	MOT2LD	Statlandsat, Yeast, Ecoli, PageBlocks, BreastCancer, Glass, Vehicle, Libra, Abalone, Vowel, Pima, Ionosphere, Segment, BreastTissue, Wine	G-mean, F1
(Barua et al., 2013)	ProWSyn	PageBlocks, Abalone, CTG, Segment, Libra, Yeast, Robot, Vehicle, Breast-tissue, Pima	F1, G-mean, AUC, StDAUC
(Sánchez et al., 2013)	SOI-CJ	Nominal, Abalone, Balloon, BreastCancer, Bupa, Car, CardioVasc, Cinco, Coil, CPU, Diabetes, Ecoli	Recall, Precision, F1

Continued on next page

Table A.1 – continued from previous page

Ref.	Algorithm	Datasets	Measure Metrics
(Barua et al., 2014)	MWMOTE	Pageblocks, Abalone, CTG, OCR, Statlandsat, Semeion, Segment, Libra, Yeast, Robot, Ecoli, Glass, Vehicle, Wine, Texture, Phoneme, Satimage, Breast-tissue, Breast-Cancer, Pima	F1, G-mean, AUC, StDAUC
(Gong & Gu, 2016)	DGSMOTE	Simulation Dataset	AUC, Acc, G-mean
(Nekooeimehr & Lai-Yuen, 2015)	A-SUWO	Vehicle, Ecoli, Pima, Balance, Liver, Wine, Breast-tissue, Libra, LEV, Iris, Heart, Glass, Haberman, Eucalyptus, Heating, Segment	F1, G-mean, AUC
(Ma & FAN, 2017)	CURE-SMOTE	Circle, Blood-transfusion, Haberman, Breast-Cancer, SPECT	F1, G-mean, AUC, OOB error

Continued on next page

Table A.1 – continued from previous page

Ref.	Algorithm	Datasets	Measure Metrics
(Abdi & Hashemi, 2016)	MDO	Breast-tissue, Hayes-roth, Wine, Autos, Glass, New-thyroid, Voice, User Knowledge-Modeling (UKM), Cleveland, Vertebral-column, Ecoli, Pageblocks, Balance, Vehicle, Contraceptive, Yeast, Wine, Abalone, Letter	MAUC, G-mean, Precision, Recall, F1
(Cateni et al., 2011)	SUNDO	Simulation Datasets, Industrial Datasets	TP, TN, FP, FN, Acc
(Gao et al., 2014)	PDFOS	Pima, Haberman, Glass, ADI, Satimage, Yeast	AUC, G-mean, F1
(Rong et al., 2014)	SSO	Ionos, Primary_tumor, Ionosphere, German, Horse, Pima	F1, G-mean
(Sandhan & Choi, 2014)	G-SMOTE	SCOP, Satimage, HypoThy, EuThy	AUC
(Bunkhumpornpat et al., 2011)	DBSMOTE	Pima, Haberman, Glass, Segment, Satimage, Ecoli, Yeast	Precision, Recall, F1, AUC
(Wang, Yao, et al., 2013)	MST-SMOTE	Pima, Haberman, Liver	Sensitivity, Specificity, G-mean, Acc
(Han et al., 2005)	Borderline-SMOTE	Simulation Data, Pima, Satimage, Haberman	F1, TP rate

Continued on next page

Table A.1 – continued from previous page

Ref.	Algorithm	Datasets	Measure Metrics
(Bunkhumpornpat et al., 2009)	Safe-level SMOTE	Satimage, Haberman	Precision, Recall, F1, AUC
(Bunkhumpornpat & Subpaiboonkit, 2013)	SL-Graph-SMOTE	Pima, Haberman, Ecoli, Statlog, Glass, Letter, PageBlocks, Yeast	F1, AUC
(Wang, Yun, et al., 2013)	TSMOTE+AB	Abalone, Ionosphere, Vehicle, Satimage, Pheneme, German, Yeast	F1, G-mean, AUC
(Koto, 2014)	SMOTE-Out, SMOTE-Cosine, Selected-SMOTE	Arrhythmia, Breast-Cancer, Liver, Dematology, Yeast, Fertility, Climate Model Simulation, Glass, Ionosphere, Statlog-Landsat, Credit Card, Car, Hill-valley, Inflammations, Carcinoma tissue, Congressional Voting, Magic Gamma Telescope, Wine	Precision, Recall, F1, Specificity, B-Acc
(Maciejewski & Stefanowski, 2011)	Local-Neighbour SMOTE	Balance, Breast-Cancer, Cleveland, CMC, Ecoli, Flags, German, Haberman, Hepatitis, Pima, Postoperative, Solar, Transfusion, Yeast	F1, G-mean, Sensitivity

Continued on next page

Table A.1 – continued from previous page

Ref.	Algorithm	Datasets	Measure Metrics
(Pérez-Ortiz et al., 2013)	Borderline Kernel Based SMOTE	Liver, Bands, Vehicle, Ecoli, Glass, Yeast, Vowel	Acc, G-mean, Minimum Sensitivity
(Torres et al., 2016)	SMOTE-D	Glass, Ecoli, Breast-Cancer, Pima, Iris, Yeast, Haberman, Vehicle, Glass, Newthyroid, Segment, Page-Blocks	F1
(Gazzah & ESSOUKRI BEN AMARA, 2008)	Polynomial Fitting Oversampling	Handwritten text	AUC, Acc, Precision, Recall, F1, G-mean, TP rate, TN rate
(Gazzah et al., 2015)	HybridMethod	Faces94, SID-Signature	TP rate, TN rate

Table A.2: Experiment Results: Experiments by Decision Tree Classifier

Data set	Metrics	RanOS	SMOTE	Border1	Border2	ENN	Tmlk	AHC	SL	CBSO	MWMOTE
ecoli	Acc	0.92558	0.88708	0.90784	0.90169	0.86916	0.88993	0.91649	0.91355	0.89875	0.91052
	AUC	0.7508	0.7416	0.74611	0.74944	0.78652	0.75155	0.76923	0.73558	0.82111	0.74097
	F1	0.52026	0.47218	0.47812	0.47913	0.48714	0.48693	0.55381	0.4774	0.52956	0.50429
	G-Mean	0.7508	0.7416	0.74611	0.74944	0.78652	0.75155	0.76923	0.73558	0.82111	0.74097
optical digits	Acc	0.97473	0.96922	0.96174	0.95765	0.96762	0.96922	0.96495	0.95943	0.96406	0.96263
	AUC	0.91648	0.91749	0.89454	0.88628	0.91437	0.91749	0.89982	0.89051	0.90362	0.90454
	F1	0.86718	0.84313	0.80538	0.78806	0.83648	0.84313	0.81941	0.79645	0.81866	0.8141
	G-Mean	0.91648	0.91749	0.89454	0.88628	0.91437	0.91749	0.89982	0.89051	0.90362	0.90454
satimage	Acc	0.91484	0.90412	0.90179	0.89309	0.87693	0.90412	0.90567	0.89308	0.90132	0.89791
	AUC	0.74883	0.77322	0.7674	0.78812	0.83406	0.77322	0.76555	0.78412	0.77904	0.77887
	F1	0.55172	0.55271	0.54155	0.54384	0.55154	0.55271	0.54758	0.54003	0.55128	0.54704
	G-Mean	0.74883	0.77322	0.7674	0.78812	0.83406	0.77322	0.76555	0.78412	0.77904	0.77887
pen digits	Acc	0.9909	0.98817	0.98908	0.9809	0.9903	0.98817	0.98969	0.98636	0.98605	0.98696
	AUC	0.96266	0.96396	0.9757	0.9501	0.97075	0.96396	0.96901	0.96576	0.96279	0.97593
	F1	0.95161	0.93839	0.94427	0.90202	0.94953	0.93839	0.94637	0.93002	0.92812	0.93435
	G-Mean	0.96266	0.96396	0.9757	0.9501	0.97075	0.96396	0.96901	0.96576	0.96279	0.97593
abalone	Acc	0.85876	0.82979	0.8396	0.83649	0.78908	0.82835	0.8444	0.83362	0.84583	0.84248

	AUC	0.57768	0.60643	0.60565	0.60351	0.75163	0.60661	0.60384	0.6067	0.62791	0.62662
	F1	0.23245	0.263	0.26945	0.26613	0.38389	0.26266	0.26699	0.26547	0.30151	0.2972
	G-Mean	0.57768	0.60643	0.60565	0.60351	0.75163	0.60661	0.60384	0.6067	0.62791	0.62662
sick euthyroid	Acc	0.96933	0.9687	0.96459	0.95542	0.96269	0.96744	0.9608	0.87797	0.9649	0.9589
	AUC	0.90338	0.91746	0.92033	0.9048	0.94082	0.91825	0.8917	0.8648	0.9004	0.90564
	F1	0.82784	0.82893	0.81208	0.77	0.81181	0.8235	0.7868	0.55626	0.80743	0.78245
	G-Mean	0.90338	0.91746	0.92033	0.9048	0.94082	0.91825	0.8917	0.8648	0.9004	0.90564
spectrometer	Acc	0.94542	0.9435	0.93788	0.9058	0.9435	0.9435	0.95856	0.95674	0.94347	0.92278
	AUC	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan
	F1	0.57457	0.64913	0.54332	0.53189	0.63036	0.65865	0.66498	0.68821	0.61622	0.55739
	G-Mean	0.7901	0.84303	0.76115	0.81495	0.84155	0.85209	0.84898	0.85962	0.81134	0.81477
car eval 34	Acc	0.98554	0.98728	0.98844	0.9867	0.98496	0.98728	0.98322	0.98553	0.98322	0.98785
	AUC	0.94334	0.93727	0.94499	0.93335	0.98116	0.93727	0.93511	0.95398	0.94226	0.96722
	F1	0.90303	0.90258	0.91404	0.89934	0.90913	0.90258	0.8894	0.90814	0.8935	0.92411
	G-Mean	0.94334	0.93727	0.94499	0.93335	0.98116	0.93727	0.93511	0.95398	0.94226	0.96722
isolet	Acc	0.95101	0.9487	0.93523	0.92869	0.93203	0.9487	0.93882	0.93549	0.94216	0.94485
	AUC	0.80806	0.84687	0.79378	0.80814	0.90215	0.84687	0.80437	0.81855	0.8289	0.84351
	F1	0.66546	0.68253	0.59641	0.58738	0.66108	0.68253	0.61738	0.61771	0.64581	0.66568
	G-Mean	0.80806	0.84687	0.79378	0.80814	0.90215	0.84687	0.80437	0.81855	0.8289	0.84351
us crime	Acc	0.90822	0.89068	0.88068	0.87512	0.84957	0.89068	0.8957	0.88617	0.88566	0.89569

	AUC	0.64116	0.70664	0.65674	0.70188	0.77006	0.70664	0.67465	0.67529	0.72513	0.73663
	F1	0.34137	0.39517	0.33349	0.36558	0.39862	0.39517	0.37134	0.34785	0.40767	0.43705
	G-Mean	0.64116	0.70664	0.65674	0.70188	0.77006	0.70664	0.67465	0.67529	0.72513	0.73663
yeast	Acc	0.85352	0.78566	0.81711	0.7799	0.6103	0.78566	0.82417	0.81298	0.74391	0.78239
	AUC	0.49716	0.50793	0.54897	0.5134	0.54812	0.50793	0.5257	0.53711	0.47764	0.51013
	F1	0.07135	0.11301	0.15829	0.11231	0.15163	0.11301	0.12102	0.14211	0.08332	0.11499
	G-Mean	0.49716	0.50793	0.54897	0.5134	0.54812	0.50793	0.5257	0.53711	0.47764	0.51013
ml8	Acc	0.87704	0.83173	0.84174	0.8193	0.75654	0.83131	0.86167	0.81926	0.81017	0.81263
	AUC	0.57416	0.5999	0.5782	0.57334	0.65546	0.59605	0.61024	0.54877	0.58321	0.59432
	F1	0.20204	0.22377	0.20391	0.18563	0.24312	0.2205	0.25151	0.1625	0.19424	0.20645
	G-Mean	0.57416	0.5999	0.5782	0.57334	0.65546	0.59605	0.61024	0.54877	0.58321	0.59432
scene	Acc	0.96111	0.95833	0.975	0.95556	0.95278	0.95833	0.95278	0.925	0.96389	0.96389
	AUC	0.78422	0.80624	0.88717	0.90044	0.86025	0.80624	0.81722	0.76778	0.86604	0.82442
	F1	0.58833	0.61	0.815	0.70333	0.65952	0.61	0.60333	0.48048	0.74667	0.63405
	G-Mean	0.78422	0.80624	0.88717	0.90044	0.86025	0.80624	0.81722	0.76778	0.86604	0.82442
libras	Acc	0.98595	0.98754	0.98436	0.97799	0.98224	0.98595	0.98436	0.94009	0.98356	0.97667
	AUC	0.9367	0.95555	0.9389	0.92194	0.95599	0.9437	0.93492	0.88025	0.93378	0.93491
	F1	0.88142	0.89723	0.87011	0.82498	0.86306	0.8808	0.87681	0.6217	0.86721	0.82324
	G-Mean	0.9367	0.95555	0.9389	0.92194	0.95599	0.9437	0.93492	0.88025	0.93378	0.93491
move	Acc	0.88495	0.89371	0.89076	0.88903	0.84443	0.89503	0.89412	0.86846	0.89208	0.88974
	AUC										
	F1										
	G-Mean										
thyroid	Acc										
	AUC										
	F1										
	G-Mean										
sick	Acc										
	AUC										
	F1										
	G-Mean										
coil 2000	Acc										

	AUC	0.54133	0.55194	0.5429	0.53634	0.58706	0.54832	0.54036	0.54933	0.55261	0.55149
	F1	0.13365	0.15524	0.13888	0.1267	0.18371	0.14934	0.13454	0.14468	0.15642	0.15264
	G-Mean	0.54133	0.55194	0.5429	0.53634	0.58706	0.54832	0.54036	0.54933	0.55261	0.55149
arrhythmia	Acc	0.97116	0.97121	0.96667	0.91155	0.96014	0.97121	0.96899	0.87401	0.97126	0.96019
	AUC	0.9283	0.92835	0.89964	0.78718	0.91609	0.92835	0.86819	0.6733	0.88438	0.9047
	F1	0.74952	0.74619	0.66762	0.34984	0.69722	0.74619	0.6677	0.23189	0.69381	0.68667
	G-Mean	0.9283	0.92835	0.89964	0.78718	0.91609	0.92835	0.86819	0.6733	0.88438	0.9047
solar flare m0	Acc	0.85456	0.91431	0.91143	0.91935	0.88121	0.91431	0.93373	0.9028	0.9078	0.90853
	AUC	0.60765	0.5022	0.51824	0.51685	0.64107	0.5022	0.54272	0.5633	0.51491	0.56389
	F1	0.19217	0.05247	0.08228	0.07485	0.23225	0.05247	0.13833	0.15134	0.08223	0.15951
	G-Mean	0.60765	0.5022	0.51824	0.51685	0.64107	0.5022	0.54272	0.5633	0.51491	0.56389
oil	Acc	0.93698	0.93379	0.92848	0.92852	0.91565	0.9349	0.93919	0.92211	0.93594	0.93488
	AUC	0.5886	0.6738	0.65385	0.67772	0.73311	0.68795	0.74487	0.70852	0.70155	0.63684
	F1	0.23477	0.32684	0.29914	0.33361	0.34647	0.34088	0.40976	0.32871	0.36726	0.26775
	G-Mean	0.5886	0.6738	0.65385	0.67772	0.73311	0.68795	0.74487	0.70852	0.70155	0.63684
car eval 4	Acc	0.99768	0.99827	0.99827	0.99711	0.99422	0.99827	0.99305	0.99306	0.99537	0.99768
	AUC	0.9829	0.98846	0.98846	0.97982	0.99703	0.98846	0.94625	0.93883	0.97229	0.9829
	F1	0.96219	0.96807	0.96807	0.9514	0.89981	0.96807	0.92221	0.90095	0.95273	0.96219
	G-Mean	0.9829	0.98846	0.98846	0.97982	0.99703	0.98846	0.94625	0.93883	0.97229	0.9829
wine quality	Acc	0.95468	0.92445	0.93651	0.9069	0.89097	0.91955	0.95284	0.9367	0.91507	0.88627

	AUC	0.64001	0.68878	0.67857	0.69224	0.68697	0.68688	0.69867	0.68814	0.68719	0.69388
	F1	0.31835	0.29744	0.31147	0.26579	0.23745	0.28197	0.39539	0.32687	0.27396	0.23729
	G-Mean	0.64001	0.68878	0.67857	0.69224	0.68697	0.68688	0.69867	0.68814	0.68719	0.69388
letter	Acc	0.9945	0.99267	0.99217	0.99283	0.9925	0.99267	0.99367	0.99367	0.99483	0.99333
	AUC	0.95645	0.95098	0.97333	0.95785	0.95315	0.95098	0.94924	0.96506	0.96793	0.95811
	F1	0.92199	0.89767	0.89625	0.90115	0.89607	0.89767	0.90995	0.91284	0.92807	0.90741
	G-Mean	0.95645	0.95098	0.97333	0.95785	0.95315	0.95098	0.94924	0.96506	0.96793	0.95811
img	Acc	0.95348	0.93193	0.94205	0.91645	0.9016	0.93191	0.94942	0.9454	0.92046	0.92385
	AUC	0.70184	0.72072	0.69606	0.69119	0.75538	0.7139	0.69578	0.67492	0.68903	0.7113
	F1	0.38221	0.32566	0.32002	0.2658	0.28998	0.32466	0.36364	0.32308	0.26171	0.31136
	G-Mean	0.70184	0.72072	0.69606	0.69119	0.75538	0.7139	0.69578	0.67492	0.68903	0.7113
yeast	Acc	0.96722	0.9677	0.96914	0.97048	0.95773	0.9677	0.97355	0.95035	0.96847	0.96971
	AUC	0.77287	0.7618	0.69461	0.79234	0.77608	0.7618	0.85861	0.73184	0.80263	0.81782
	F1	0.49853	0.4917	0.42908	0.5403	0.44248	0.4917	0.6156	0.36675	0.53333	0.55493
	G-Mean	0.77287	0.7618	0.69461	0.79234	0.77608	0.7618	0.85861	0.73184	0.80263	0.81782
me2	Acc	0.95781	0.92114	0.9373	0.90457	0.90695	0.92153	0.94204	0.94124	0.9231	0.92112
	AUC	0.61669	0.59202	0.61341	0.60942	0.56877	0.60451	0.64604	0.60975	0.60108	0.6004
	F1	0.24514	0.13806	0.17188	0.14444	0.1216	0.14561	0.24177	0.17152	0.16158	0.14862
	G-Mean	0.61669	0.59202	0.61341	0.60942	0.56877	0.60451	0.64604	0.60975	0.60108	0.6004
webpage	Acc	0.98331	0.9696	0.97884	0.94158	0.96423	0.96632	0.9845	0.97526	0.96781	0.95499
	AUC										
	F1										
	G-Mean										
ozone	Acc										
	AUC										
	F1										
	G-Mean										
level	Acc										
	AUC										
	F1										
	G-Mean										
mammography	Acc										

	AUC	0.81315	0.83916	0.85049	0.85786	0.90246	0.83748	0.85999	0.77601	0.88448	0.90434
	F1	0.62667	0.50485	0.59887	0.36774	0.5082	0.47926	0.675	0.50299	0.52212	0.45487
	G-Mean	0.81315	0.83916	0.85049	0.85786	0.90246	0.83748	0.85999	0.77601	0.88448	0.90434
protein homo	Acc	0.99474	0.98658	0.99227	0.98589	0.98701	0.98658	0.9944	0.99364	0.98257	0.98157
	AUC	0.85241	0.89028	0.87148	0.8791	0.89186	0.89028	0.89016	0.85321	0.88826	0.89995
	F1	0.69251	0.497	0.61851	0.47756	0.50609	0.497	0.70085	0.65163	0.43219	0.42593
	G-Mean	0.85241	0.89028	0.87148	0.8791	0.89186	0.89028	0.89016	0.85321	0.88826	0.89995
abalone 19	Acc	0.98731	0.95211	0.97486	0.97127	0.92434	0.95164	0.97822	0.97774	0.97414	0.97846
	AUC	0.51401	0.54056	0.52845	0.52665	0.64312	0.5333	0.54669	0.51753	0.51976	0.54264
	F1	0.05	0.04028	0.0381	0.02538	0.06703	0.02911	0.05317	0.02222	0.02967	0.06
	G-Mean	0.51401	0.54056	0.52845	0.52665	0.64312	0.5333	0.54669	0.51753	0.51976	0.54264
haberman	Acc	0.63688	0.59817	0.65032	0.57194	0.61774	0.62731	0.67011	0.58194	0.6672	0.60505
	AUC	0.54477	0.50731	0.59411	0.51573	0.61276	0.55953	0.60523	0.50441	0.60749	0.55295
	F1	0.31685	0.29824	0.38957	0.30099	0.42881	0.35349	0.42191	0.29838	0.40481	0.34644
	G-Mean	0.54477	0.50731	0.59411	0.51573	0.61276	0.55953	0.60523	0.50441	0.60749	0.55295
glass 5	Acc	0.95346	0.94416	0.97186	0.95823	0.92489	0.94416	0.95368	0.95368	0.95823	0.94848
	AUC	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan
	F1	0.53333	0.47333	0.56333	0.50333	0.50667	0.47333	0.5	0.53333	0.58333	0.48333
	G-Mean	0.66224	0.63211	0.66974	0.66237	0.71696	0.63211	0.63986	0.66236	0.68711	0.61224
glass 6	Acc	0.99069	0.97706	0.97706	0.9671	0.9816	0.97706	0.97229	0.97229	0.99069	0.98593

	AUC	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan
	F1	0.6	0.54	0.54	0.34667	0.56667	0.54	0.44	0.44	0.6	0.5
	G-Mean	0.69773	0.66176	0.66176	0.57606	0.67843	0.66176	0.61176	0.61176	0.69773	0.64773
glass 7	Acc	0.94805	0.94329	0.94329	0.95758	0.93874	0.94329	0.95736	0.95281	0.94827	0.94805
	AUC	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan
	F1	0.72389	0.73556	0.72127	0.76198	0.69294	0.73556	0.77627	0.77111	0.74071	0.7296
	G-Mean	0.79924	0.83856	0.81495	0.81765	0.80418	0.83856	0.83674	0.84369	0.80866	0.80897
heart	Acc	0.76247	0.74323	0.73946	0.73591	0.68323	0.74344	0.77559	0.7657	0.76548	0.7557
	AUC	0.7557	0.73334	0.7305	0.73147	0.68672	0.73644	0.76863	0.75664	0.76066	0.75046
	F1	0.73074	0.70643	0.69352	0.70207	0.6653	0.70577	0.7428	0.71809	0.73476	0.72245
	G-Mean	0.7557	0.73334	0.7305	0.73147	0.68672	0.73644	0.76863	0.75664	0.76066	0.75046
ionosphere	Acc	0.8777	0.86627	0.86056	0.85778	0.90603	0.86627	0.86619	0.89746	0.87159	0.89167
	AUC	0.86847	0.86686	0.85646	0.85856	0.89464	0.86582	0.85359	0.89068	0.86354	0.88041
	F1	0.82661	0.81908	0.80518	0.80948	0.86433	0.81819	0.80567	0.8542	0.8193	0.83994
	G-Mean	0.86847	0.86686	0.85646	0.85856	0.89464	0.86582	0.85359	0.89068	0.86354	0.88041
pima	Acc	0.68758	0.6783	0.69395	0.69412	0.71483	0.68225	0.68626	0.66796	0.69665	0.68237
	AUC	0.65789	0.65428	0.67336	0.6763	0.7269	0.66464	0.66644	0.65791	0.67867	0.66791
	F1	0.55425	0.5559	0.57498	0.58201	0.65033	0.56868	0.57102	0.56407	0.58722	0.57714
	G-Mean	0.65789	0.65428	0.67336	0.6763	0.7269	0.66464	0.66644	0.65791	0.67867	0.66791
Breast	Acc	0.92437	0.92964	0.94019	0.90326	0.93493	0.92265	0.92099	0.93324	0.92447	0.92262

AUC	0.92122	0.92581	0.93857	0.91085	0.93544	0.92165	0.91732	0.92971	0.92269	0.92428
F1	0.89799	0.90233	0.92068	0.87504	0.90909	0.89764	0.89178	0.90978	0.90179	0.89836
G-Mean	0.92122	0.92581	0.93857	0.91085	0.93544	0.92165	0.91732	0.92971	0.92269	0.92428

Table A.3: Experiment Results: Experiments by 5 Nearest Neighbours Classifier

Data set	Metrics	RanOS	SMOTE	Border1	Border2	ENN	Tmlk	AHC	SL	CBSO	MWMOTE
ecoli	Acc	0.88111	0.87825	0.87807	0.85134	0.86025	0.87237	0.93146	0.9344	0.87522	0.87228
	AUC	0.85227	0.86542	0.84241	0.82769	0.86204	0.86202	0.85817	0.77911	0.87022	0.84688
	F1	0.5331	0.55384	0.53113	0.48294	0.52244	0.5456	0.66026	0.58342	0.55103	0.52484
	G-Mean	0.85227	0.86542	0.84241	0.82769	0.86204	0.86202	0.85817	0.77911	0.87022	0.84688
optical digits	Acc	0.99609	0.99573	0.99573	0.99306	0.99502	0.99573	0.9968	0.99537	0.99431	0.99395
	AUC	0.98982	0.99286	0.99196	0.99048	0.99246	0.99286	0.99101	0.9839	0.99299	0.99112
	F1	0.97989	0.97819	0.97773	0.96499	0.97483	0.97819	0.98331	0.97617	0.97164	0.96944
	G-Mean	0.98982	0.99286	0.99196	0.99048	0.99246	0.99286	0.99101	0.9839	0.99299	0.99112
satimage	Acc	0.91267	0.88112	0.87444	0.84429	0.84243	0.88112	0.90583	0.87397	0.87242	0.87351
	AUC	0.90485	0.90892	0.90551	0.8928	0.89456	0.90892	0.88893	0.85558	0.90242	0.89715
	F1	0.66597	0.60647	0.59366	0.54395	0.54165	0.60647	0.64161	0.56219	0.58838	0.58781
	G-Mean	0.90485	0.90892	0.90551	0.8928	0.89456	0.90892	0.88893	0.85558	0.90242	0.89715
pen digits	Acc	0.99757	0.99788	0.99727	0.99666	0.99697	0.99788	0.99757	0.99636	0.99727	0.99727
	AUC	0.99444	0.99602	0.99287	0.99254	0.99551	0.99602	0.99444	0.99237	0.99428	0.99568
	F1	0.98746	0.98905	0.98587	0.98279	0.98442	0.98905	0.98746	0.98125	0.98592	0.98596
	G-Mean	0.99444	0.99602	0.99287	0.99254	0.99551	0.99602	0.99444	0.99237	0.99428	0.99568
abalone	Acc	0.7953	0.77376	0.79651	0.80058	0.73066	0.77232	0.8475	0.81876	0.78285	0.81854

	AUC	0.68921	0.72203	0.72543	0.72449	0.7714	0.72116	0.6598	0.6224	0.74486	0.74143
	F1	0.33706	0.35099	0.36859	0.36926	0.36244	0.3495	0.34215	0.28157	0.37392	0.39676
	G-Mean	0.68921	0.72203	0.72543	0.72449	0.7714	0.72116	0.6598	0.6224	0.74486	0.74143
sick euthyroid	Acc	0.77869	0.75182	0.79292	0.74171	0.68353	0.74961	0.86216	0.73822	0.75687	0.71831
	AUC	0.66448	0.66067	0.66553	0.68727	0.68965	0.65809	0.61557	0.6829	0.69026	0.68773
	F1	0.29991	0.28402	0.30679	0.30211	0.28164	0.28119	0.29135	0.29567	0.31201	0.29341
	G-Mean	0.66448	0.66067	0.66553	0.68727	0.68965	0.65809	0.61557	0.6829	0.69026	0.68773
spectrometer	Acc	0.96611	0.96607	0.96796	0.95479	0.96607	0.96607	0.97173	0.95856	0.96607	0.96419
	AUC	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan
	F1	0.74019	0.74382	0.75088	0.66923	0.75271	0.74382	0.76142	0.70751	0.72109	0.72036
	G-Mean	0.86342	0.87071	0.86136	0.87366	0.87081	0.87071	0.85835	0.83753	0.87975	0.89292
car eval 34	Acc	0.92532	0.89989	0.91146	0.80324	0.88948	0.89989	0.94966	0.94557	0.78125	0.91437
	AUC	0.7615	0.91398	0.89214	0.89315	0.9191	0.91398	0.97006	0.65217	0.8814	0.95365
	F1	0.53674	0.5822	0.59285	0.43418	0.5643	0.5822	0.74725	0.45068	0.40856	0.63794
	G-Mean	0.7615	0.91398	0.89214	0.89315	0.9191	0.91398	0.97006	0.65217	0.8814	0.95365
isolet	Acc	0.94972	0.90483	0.90983	0.87931	0.87893	0.90483	0.90753	0.95255	0.90381	0.90291
	AUC	0.95895	0.94514	0.94484	0.92911	0.93443	0.94514	0.94432	0.92961	0.94458	0.94509
	F1	0.74745	0.61554	0.62651	0.55647	0.55901	0.61554	0.62113	0.7448	0.61292	0.61115
	G-Mean	0.95895	0.94514	0.94484	0.92911	0.93443	0.94514	0.94432	0.92961	0.94458	0.94509
us crime	Acc	0.88919	0.84806	0.87814	0.86059	0.80796	0.84806	0.89772	0.87963	0.85159	0.8561

	AUC	0.75211	0.80359	0.82413	0.81508	0.83564	0.80359	0.74291	0.6989	0.80931	0.80503
	F1	0.43737	0.42202	0.47792	0.44657	0.40102	0.42202	0.44717	0.37371	0.4317	0.43467
	G-Mean	0.75211	0.80359	0.82413	0.81508	0.83564	0.80359	0.74291	0.6989	0.80931	0.80503
yeast	Acc	0.77907	0.44561	0.51347	0.40133	0.33183	0.44561	0.38686	0.72899	0.3517	0.36244
	AUC	0.54429	0.5887	0.59458	0.57832	0.57993	0.5887	0.57592	0.52752	0.56852	0.56326
ml8	F1	0.1539	0.16462	0.16946	0.15976	0.15893	0.16462	0.1588	0.13328	0.15574	0.15446
	G-Mean	0.54429	0.5887	0.59458	0.57832	0.57993	0.5887	0.57592	0.52752	0.56852	0.56326
scene	Acc	0.8255	0.62986	0.69341	0.68342	0.52683	0.62986	0.69009	0.83798	0.64108	0.6336
	AUC	0.63181	0.68789	0.68385	0.7	0.6767	0.68789	0.71562	0.59502	0.70349	0.69249
	F1	0.25517	0.23201	0.2473	0.25198	0.20947	0.23201	0.26128	0.2186	0.24096	0.23476
	G-Mean	0.63181	0.68789	0.68385	0.7	0.6767	0.68789	0.71562	0.59502	0.70349	0.69249
libras	Acc	0.96389	0.95556	0.95556	0.95	0.95	0.95556	0.96667	0.94444	0.95833	0.95556
	AUC	0.91996	0.92649	0.91551	0.92355	0.94704	0.92649	0.87438	0.9315	0.94051	0.91546
move	F1	0.77619	0.75119	0.74643	0.72405	0.74286	0.75119	0.75619	0.71175	0.77976	0.74167
	G-Mean	0.91996	0.92649	0.91551	0.92355	0.94704	0.92649	0.87438	0.9315	0.94051	0.91546
thyroid	Acc	0.85524	0.80593	0.85047	0.80434	0.75954	0.80753	0.90615	0.85896	0.7948	0.78897
	AUC	0.68669	0.70699	0.70342	0.73197	0.71421	0.70581	0.608	0.61887	0.71885	0.75799
sick	F1	0.29074	0.26917	0.30016	0.28521	0.2504	0.26931	0.25252	0.22943	0.27112	0.29213
	G-Mean	0.68669	0.70699	0.70342	0.73197	0.71421	0.70581	0.608	0.61887	0.71885	0.75799
coil 2000	Acc	0.8144	0.78599	0.81613	0.81338	0.71411	0.78528	0.89289	0.85482	0.7974	0.803

	AUC	0.58201	0.58716	0.58963	0.58644	0.60762	0.58596	0.55641	0.54895	0.58805	0.58009
	F1	0.17019	0.16718	0.17646	0.17277	0.16871	0.16609	0.16168	0.14225	0.17097	0.16557
	G-Mean	0.58201	0.58716	0.58963	0.58644	0.60762	0.58596	0.55641	0.54895	0.58805	0.58009
arrhythmia	Acc	0.8428	0.71014	0.74097	0.72324	0.60618	0.71014	0.84527	0.8471	0.67923	0.71889
	AUC	0.63081	0.65115	0.66504	0.65591	0.67683	0.65115	0.68449	0.61295	0.62373	0.63375
	F1	0.19556	0.17218	0.18826	0.18256	0.16832	0.17218	0.20716	0.20231	0.15373	0.16339
	G-Mean	0.63081	0.65115	0.66504	0.65591	0.67683	0.65115	0.68449	0.61295	0.62373	0.63375
solar flare m0	Acc	0.91863	0.85094	0.85814	0.84158	0.80199	0.85094	0.88912	0.93376	0.87184	0.83582
	AUC	0.6027	0.64767	0.65819	0.6755	0.69314	0.64767	0.57352	0.58573	0.6371	0.67595
	F1	0.21973	0.19117	0.20262	0.20611	0.20792	0.19117	0.15526	0.19266	0.18729	0.21047
	G-Mean	0.6027	0.64767	0.65819	0.6755	0.69314	0.64767	0.57352	0.58573	0.6371	0.67595
oil	Acc	0.89112	0.82282	0.85479	0.8399	0.77694	0.81321	0.93704	0.90286	0.82811	0.83885
	AUC	0.62091	0.71288	0.67748	0.70521	0.66753	0.70785	0.5516	0.61094	0.68013	0.71082
	F1	0.20352	0.23933	0.24685	0.23676	0.18938	0.22801	0.15214	0.2129	0.21235	0.24883
	G-Mean	0.62091	0.71288	0.67748	0.70521	0.66753	0.70785	0.5516	0.61094	0.68013	0.71082
car eval 4	Acc	0.96296	0.91898	0.93056	0.85357	0.91841	0.91898	0.97337	0.97221	0.88484	0.94736
	AUC	0.78216	0.94397	0.94994	0.92393	0.94367	0.94397	0.97624	0.64353	0.94021	0.96202
	F1	0.51028	0.45973	0.49379	0.33156	0.458	0.45973	0.71729	0.41773	0.38765	0.56805
	G-Mean	0.78216	0.94397	0.94994	0.92393	0.94367	0.94397	0.97624	0.64353	0.94021	0.96202
wine quality	Acc	0.91017	0.82585	0.88261	0.80585	0.78114	0.82462	0.94835	0.92446	0.80421	0.75337

	AUC	0.6539	0.68441	0.69747	0.70008	0.69393	0.68377	0.57995	0.57779	0.6919	0.70051
	F1	0.22859	0.18008	0.23303	0.17728	0.16524	0.17891	0.20293	0.15616	0.17304	0.16062
	G-Mean	0.6539	0.68441	0.69747	0.70008	0.69393	0.68377	0.57995	0.57779	0.6919	0.70051
letter	Acc	0.99667	0.99583	0.99617	0.988	0.9945	0.99583	0.99817	0.99683	0.99733	0.99183
	AUC	0.99827	0.99784	0.99575	0.99378	0.99715	0.99784	0.99453	0.99158	0.99862	0.99577
	F1	0.95516	0.94457	0.94855	0.85542	0.9281	0.94457	0.9746	0.95672	0.9638	0.89684
	G-Mean	0.99827	0.99784	0.99575	0.99378	0.99715	0.99784	0.99453	0.99158	0.99862	0.99577
img	Acc	0.92922	0.89958	0.93124	0.89421	0.87869	0.89891	0.95821	0.94067	0.89218	0.89216
	AUC	0.7386	0.76673	0.77632	0.7572	0.80621	0.76638	0.68026	0.66162	0.75615	0.76969
	F1	0.34188	0.3009	0.37038	0.2762	0.28974	0.29972	0.38783	0.2714	0.27219	0.28244
	G-Mean	0.7386	0.76673	0.77632	0.7572	0.80621	0.76638	0.68026	0.66162	0.75615	0.76969
me2	Acc	0.97518	0.90243	0.91288	0.85605	0.83113	0.90243	0.95505	0.94	0.92103	0.90167
	AUC	0.84975	0.89478	0.89531	0.86929	0.87748	0.89478	0.89114	0.81223	0.89627	0.88792
	F1	0.62409	0.34323	0.36655	0.26083	0.23986	0.34323	0.51298	0.39341	0.38782	0.33806
	G-Mean	0.84975	0.89478	0.89531	0.86929	0.87748	0.89478	0.89114	0.81223	0.89627	0.88792
webpage	Acc	0.90221	0.80717	0.86791	0.75395	0.75434	0.80717	0.94203	0.96016	0.78982	0.78903
	AUC	0.52624	0.58401	0.56935	0.55898	0.64916	0.58401	0.50644	0.50412	0.68749	0.63027
	F1	0.06677	0.09667	0.09823	0.07919	0.11472	0.09667	0.04683	0.02857	0.12782	0.10761
	G-Mean	0.52624	0.58401	0.56935	0.55898	0.64916	0.58401	0.50644	0.50412	0.68749	0.63027
ozone level	Acc	0.97854	0.96155	0.97615	0.92459	0.95142	0.96155	0.98718	0.98152	0.95112	0.95201
mammography											

	AUC	0.88996	0.9209	0.90195	0.8954	0.93553	0.9209	0.86796	0.81223	0.94199	0.92263
	F1	0.62105	0.50193	0.60396	0.33596	0.45485	0.50193	0.71895	0.60256	0.45695	0.45051
	G-Mean	0.88996	0.9209	0.90195	0.8954	0.93553	0.9209	0.86796	0.81223	0.94199	0.92263
protein homo	Acc	0.9882	0.95831	0.98086	0.92716	0.95035	0.95831	0.9933	0.99181	0.93612	0.92439
	AUC	0.76512	0.80424	0.77632	0.79259	0.81241	0.80424	0.72977	0.68432	0.82962	0.83996
	F1	0.43297	0.20636	0.332	0.13097	0.18475	0.20636	0.53566	0.43175	0.15899	0.14308
	G-Mean	0.76512	0.80424	0.77632	0.79259	0.81241	0.80424	0.72977	0.68432	0.82962	0.83996
abalone 19	Acc	0.97247	0.89706	0.96241	0.9526	0.8645	0.89682	0.99019	0.983	0.93991	0.95811
	AUC	0.49701	0.66556	0.54576	0.55319	0.71295	0.66544	0.50594	0.50518	0.59144	0.53239
	F1	0.01538	0.05767	0.05536	0.04794	0.05717	0.05756	0.025	0.02857	0.06056	0.036
	G-Mean	0.49701	0.66556	0.54576	0.55319	0.71295	0.66544	0.50594	0.50518	0.59144	0.53239
haberman	Acc	0.60419	0.62086	0.60796	0.58473	0.63387	0.6143	0.64344	0.61108	0.62742	0.61409
	AUC	0.58715	0.58359	0.55412	0.53822	0.63511	0.57804	0.58214	0.52446	0.605	0.57578
	F1	0.41411	0.41337	0.38622	0.35717	0.45711	0.40432	0.38152	0.31632	0.43764	0.39925
	G-Mean	0.58715	0.58359	0.55412	0.53822	0.63511	0.57804	0.58214	0.52446	0.605	0.57578
glass 5	Acc	0.9671	0.96255	0.96255	0.95346	0.93918	0.96255	0.96255	0.9632	0.96255	0.9671
	AUC	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan
	F1	0.65667	0.62333	0.62333	0.60667	0.56667	0.62333	0.59667	0.46667	0.62333	0.65667
	G-Mean	0.80986	0.80747	0.80747	0.82484	0.79521	0.80747	0.71224	0.5975	0.80747	0.80986
glass 6	Acc	0.9539	0.94913	0.94437	0.94892	0.93485	0.94913	0.9632	0.9632	0.93983	0.9539

	AUC	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan
	F1	0.46667	0.53571	0.43571	0.51905	0.50905	0.53571	0.40667	0.34	0.41667	0.46667	
	G-Mean	0.66423	0.77362	0.72362	0.7234	0.81624	0.77362	0.60711	0.55949	0.70696	0.66423	
glass 7	Acc	0.93485	0.96255	0.93009	0.93506	0.9487	0.96255	0.9487	0.93918	0.93939	0.93939	
	AUC	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	
	F1	0.74278	0.79849	0.7027	0.71714	0.75087	0.79849	0.73222	0.70841	0.74278	0.74278	
	G-Mean	0.87483	0.9001	0.83516	0.85227	0.86032	0.9001	0.84593	0.83094	0.8771	0.8771	
heart	Acc	0.61355	0.65667	0.60398	0.62333	0.63677	0.66323	0.63376	0.65323	0.64032	0.63011	
	AUC	0.61125	0.6551	0.60176	0.62466	0.63757	0.66286	0.64448	0.65045	0.638	0.62835	
	F1	0.58006	0.62334	0.58029	0.60486	0.60775	0.6187	0.6522	0.62253	0.6116	0.60132	
	G-Mean	0.61125	0.6551	0.60176	0.62466	0.63757	0.66286	0.64448	0.65045	0.638	0.62835	
ionosphere	Acc	0.86619	0.8804	0.88611	0.88897	0.87468	0.8804	0.88611	0.87175	0.88603	0.88619	
	AUC	0.82646	0.84543	0.85232	0.8545	0.83282	0.84543	0.85392	0.83382	0.85119	0.85369	
	F1	0.77766	0.80436	0.81461	0.81614	0.7905	0.80436	0.81595	0.78947	0.80919	0.81596	
	G-Mean	0.82646	0.84543	0.85232	0.8545	0.83282	0.84543	0.85392	0.83382	0.85119	0.85369	
pima	Acc	0.65369	0.65883	0.65231	0.63922	0.67061	0.6549	0.66535	0.66006	0.67319	0.67442	
	AUC	0.64791	0.66032	0.66528	0.64506	0.70276	0.6585	0.67601	0.66104	0.68	0.6769	
	F1	0.55887	0.57551	0.58591	0.56272	0.62943	0.5747	0.59495	0.57608	0.59944	0.59495	
	G-Mean	0.64791	0.66032	0.66528	0.64506	0.70276	0.6585	0.67601	0.66104	0.68	0.6769	
Breast	Acc	0.93315	0.92967	0.9209	0.89458	0.9349	0.92967	0.91739	0.92967	0.92086	0.92086	

AUC	0.93205	0.92878	0.9265	0.90429	0.92965	0.92878	0.92021	0.92961	0.9259	0.92645
F1	0.90937	0.90544	0.89745	0.86931	0.91146	0.90544	0.89122	0.90461	0.899	0.89667
G-Mean	0.93205	0.92878	0.9265	0.90429	0.92965	0.92878	0.92021	0.92961	0.9259	0.92645

Table A.4: Experiment Results: Experiments by Support Vector Machine Classifier

Data set	Metrics	RanOS	SMOTE	Border1	Border2	ENN	Tmlk	AHC	SL	CBSO	MWMOTE
ecoli	Acc	0.81569	0.82451	0.8098	0.78583	0.80971	0.82157	0.88993	0.8959	0.80071	0.82451
	AUC	0.86698	0.87234	0.86425	0.85082	0.86392	0.87055	0.5588	0.5	0.85901	0.87234
	F1	0.48306	0.49364	0.47469	0.44642	0.47781	0.48972	0.13	0.0	0.46762	0.49364
	G-Mean	0.86698	0.87234	0.86425	0.85082	0.86392	0.87055	0.5588	0.5	0.85901	0.87234
optical digits	Acc	0.9016	0.90249	0.90142	0.90142	0.90249	0.90249	0.9016	0.9016	0.90302	0.9048
	AUC	0.501	0.50535	0.5	0.5	0.50535	0.50535	0.501	0.50089	0.50805	0.51694
	F1	0.00392	0.02078	0.0	0.0	0.02078	0.02078	0.00392	0.00351	0.03118	0.06456
	G-Mean	0.501	0.50535	0.5	0.5	0.50535	0.50535	0.501	0.50089	0.50805	0.51694
satimage	Acc	0.90272	0.90272	0.90272	0.90272	0.90272	0.90272	0.90272	0.90272	0.90272	0.90272
	AUC	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
	F1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	G-Mean	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
pen digits	Acc	0.90358	0.90358	0.90358	0.90358	0.90358	0.90358	0.90358	0.90358	0.90358	0.90358
	AUC	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
	F1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	G-Mean	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
abalone	Acc	0.66268	0.66364	0.67633	0.67992	0.63204	0.66388	0.9064	0.82334	0.65095	0.71679

	AUC	0.76112	0.77363	0.76379	0.75248	0.76621	0.77376	0.5	0.59107	0.77258	0.7497
	F1	0.3273	0.33486	0.33353	0.32858	0.32087	0.33504	0.0	0.20412	0.3302	0.34226
	G-Mean	0.76112	0.77363	0.76379	0.75248	0.76621	0.77376	0.5	0.59107	0.77258	0.7497
sick	Acc	0.87987	0.84984	0.87513	0.8315	0.80747	0.85205	0.90327	0.82897	0.85395	0.80873
	AUC	0.62761	0.63239	0.63874	0.68642	0.63516	0.63358	0.58109	0.6462	0.68116	0.7088
	F1	0.31818	0.30328	0.33177	0.35158	0.28391	0.30622	0.25387	0.30493	0.36579	0.35452
	G-Mean	0.62761	0.63239	0.63874	0.68642	0.63516	0.63358	0.58109	0.6462	0.68116	0.7088
euthyroid	Acc	0.9153	0.9153	0.9153	0.9153	0.50168	0.9153	0.9153	0.9153	0.9153	0.9153
	AUC	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan
	F1	0.0	0.0	0.0	0.0	0.07789	0.0	0.0	0.0	0.0	0.0
	G-Mean	0.45	0.45	0.45	0.45	0.45	0.45	0.45	0.45	0.45	0.45
spectrometer	Acc	0.95486	0.9618	0.96065	0.9537	0.94502	0.9618	0.97626	0.96466	0.96469	0.9618
	AUC	0.97543	0.97922	0.97861	0.97484	0.97006	0.97922	0.9485	0.78559	0.98078	0.97922
	F1	0.77246	0.79995	0.79534	0.76888	0.73881	0.79995	0.86373	0.71398	0.81278	0.79995
	G-Mean	0.97543	0.97922	0.97861	0.97484	0.97006	0.97922	0.9485	0.78559	0.98078	0.97922
car eval 34	Acc	0.96614	0.97461	0.97191	0.96794	0.94639	0.97461	0.98205	0.96435	0.97191	0.97576
	AUC	0.97382	0.96902	0.96139	0.96164	0.96397	0.96902	0.94015	0.91369	0.97249	0.96883
	F1	0.81597	0.85245	0.83769	0.81964	0.73731	0.85245	0.8825	0.78496	0.84132	0.85875
	G-Mean	0.97382	0.96902	0.96139	0.96164	0.96397	0.96902	0.94015	0.91369	0.97249	0.96883
isolet	Acc	0.84957	0.85358	0.8621	0.83954	0.81748	0.85358	0.93981	0.9288	0.84557	0.8571
us crime											

	AUC	0.85315	0.84486	0.8594	0.86454	0.8595	0.84486	0.70462	0.72938	0.85447	0.84554
	F1	0.45662	0.45709	0.4801	0.45234	0.42506	0.45709	0.50162	0.50373	0.4527	0.46262
	G-Mean	0.85315	0.84486	0.8594	0.86454	0.8595	0.84486	0.70462	0.72938	0.85447	0.84554
yeast	Acc	0.5325	0.52214	0.56436	0.54202	0.13444	0.52214	0.92635	0.92635	0.51427	0.50145
	AUC	0.58596	0.57974	0.56889	0.56733	0.53285	0.57974	0.5	0.5	0.58461	0.57923
	F1	0.16726	0.16336	0.16003	0.15832	0.14514	0.16336	0.0	0.0	0.1652	0.16243
	G-Mean	0.58596	0.57974	0.56889	0.56733	0.53285	0.57974	0.5	0.5	0.58461	0.57923
ml8	Acc	0.73951	0.7175	0.76362	0.73079	0.52058	0.7175	0.92647	0.87328	0.72582	0.70963
	AUC	0.70093	0.70366	0.69094	0.70385	0.69298	0.70366	0.5	0.52443	0.71243	0.70638
	F1	0.26832	0.26183	0.27161	0.26671	0.21557	0.26183	0.0	0.1031	0.26963	0.26054
	G-Mean	0.70093	0.70366	0.69094	0.70385	0.69298	0.70366	0.5	0.52443	0.71243	0.70638
scene	Acc	0.95556	0.95278	0.92778	0.90278	0.94722	0.95278	0.93333	0.95	0.91667	0.95556
	AUC	0.88502	0.88351	0.88538	0.87209	0.85704	0.88351	0.5	0.8819	0.80447	0.90017
	F1	0.71095	0.70381	0.62905	0.57902	0.65714	0.70381	0.0	0.72492	0.54494	0.73381
	G-Mean	0.88502	0.88351	0.88538	0.87209	0.85704	0.88351	0.5	0.8819	0.80447	0.90017
libras	Acc	0.91383	0.89846	0.91251	0.86611	0.85631	0.89687	0.9316	0.90509	0.88069	0.86558
	AUC	0.61337	0.66532	0.64809	0.70136	0.67803	0.66212	0.55841	0.57309	0.68553	0.74679
	F1	0.2667	0.31805	0.31905	0.31602	0.28358	0.3122	0.1895	0.19596	0.31886	0.3551
	G-Mean	0.61337	0.66532	0.64809	0.70136	0.67803	0.66212	0.55841	0.57309	0.68553	0.74679
move	Acc	0.83374	0.84423	0.86571	0.86897	0.80401	0.84423	0.93474	0.88027	0.84179	0.87284
	AUC	0.88502	0.88351	0.88538	0.87209	0.85704	0.88351	0.5	0.8819	0.80447	0.90017
	F1	0.71095	0.70381	0.62905	0.57902	0.65714	0.70381	0.0	0.72492	0.54494	0.73381
	G-Mean	0.88502	0.88351	0.88538	0.87209	0.85704	0.88351	0.5	0.8819	0.80447	0.90017
thyroid	Acc	0.91383	0.89846	0.91251	0.86611	0.85631	0.89687	0.9316	0.90509	0.88069	0.86558
	AUC	0.61337	0.66532	0.64809	0.70136	0.67803	0.66212	0.55841	0.57309	0.68553	0.74679
	F1	0.2667	0.31805	0.31905	0.31602	0.28358	0.3122	0.1895	0.19596	0.31886	0.3551
	G-Mean	0.61337	0.66532	0.64809	0.70136	0.67803	0.66212	0.55841	0.57309	0.68553	0.74679
sick	Acc	0.83374	0.84423	0.86571	0.86897	0.80401	0.84423	0.93474	0.88027	0.84179	0.87284
	AUC	0.88502	0.88351	0.88538	0.87209	0.85704	0.88351	0.5	0.8819	0.80447	0.90017
	F1	0.71095	0.70381	0.62905	0.57902	0.65714	0.70381	0.0	0.72492	0.54494	0.73381
	G-Mean	0.88502	0.88351	0.88538	0.87209	0.85704	0.88351	0.5	0.8819	0.80447	0.90017
coil 2000	Acc	0.83374	0.84423	0.86571	0.86897	0.80401	0.84423	0.93474	0.88027	0.84179	0.87284

	AUC	0.59508	0.59093	0.58817	0.58404	0.61174	0.59093	0.51607	0.54978	0.58879	0.57808
	F1	0.18922	0.18874	0.19537	0.19141	0.19311	0.18874	0.06752	0.14774	0.18574	0.18625
	G-Mean	0.59508	0.59093	0.58817	0.58404	0.61174	0.59093	0.51607	0.54978	0.58879	0.57808
arrhythmia	Acc	0.94473	0.94473	0.94473	0.94473	0.94473	0.94473	0.94473	0.94473	0.94473	0.94473
	AUC	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
	F1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	G-Mean	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
solar flare m0	Acc	0.70629	0.73292	0.73867	0.75018	0.7401	0.73292	0.95102	0.90208	0.72715	0.76241
	AUC	0.70984	0.68737	0.65979	0.65647	0.70859	0.68737	0.5	0.65598	0.6964	0.68058
	F1	0.18859	0.18265	0.17282	0.17198	0.19427	0.18265	0.0	0.26496	0.18455	0.18772
	G-Mean	0.70984	0.68737	0.65979	0.65647	0.70859	0.68737	0.5	0.65598	0.6964	0.68058
oil	Acc	0.95628	0.95628	0.95628	0.95628	0.59035	0.95628	0.95628	0.95628	0.95628	0.95628
	AUC	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
	F1	0.0	0.0	0.0	0.0	0.03243	0.0	0.0	0.0	0.0	0.0
	G-Mean	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
car eval 4	Acc	0.98206	0.98206	0.98206	0.96239	0.96702	0.98206	0.9838	0.96238	0.98206	0.98206
	AUC	0.99069	0.99069	0.99069	0.98046	0.98286	0.99069	0.99161	0.5	0.99069	0.99069
	F1	0.79291	0.79291	0.79291	0.64946	0.68349	0.79291	0.80567	0.0	0.79291	0.79291
	G-Mean	0.99069	0.99069	0.99069	0.98046	0.98286	0.99069	0.99161	0.5	0.99069	0.99069
wine quality	Acc	0.951	0.9067	0.93385	0.86566	0.86178	0.9065	0.96182	0.94569	0.86055	0.8134

	AUC	0.58605	0.61265	0.61954	0.6626	0.62944	0.61522	0.54963	0.55512	0.65965	0.69141
	F1	0.21333	0.184	0.23191	0.19175	0.16209	0.18621	0.15729	0.14327	0.18592	0.17989
	G-Mean	0.58605	0.61265	0.61954	0.6626	0.62944	0.61522	0.54963	0.55512	0.65965	0.69141
letter	Acc	0.9985	0.99833	0.99833	0.99783	0.99767	0.99833	0.9985	0.997	0.9985	0.99867
	AUC	0.98113	0.97879	0.97879	0.98757	0.9807	0.97879	0.98113	0.97583	0.98113	0.99026
	F1	0.97852	0.97608	0.97608	0.9697	0.96698	0.97608	0.97852	0.95755	0.97852	0.98122
	G-Mean	0.98113	0.97879	0.97879	0.98757	0.9807	0.97879	0.98113	0.97583	0.98113	0.99026
img	Acc	0.85443	0.8659	0.88275	0.86254	0.85579	0.8659	0.9656	0.92925	0.84904	0.86388
	AUC	0.85237	0.83525	0.82764	0.8471	0.84632	0.83525	0.5	0.66337	0.8496	0.841
	F1	0.27539	0.28345	0.30411	0.28613	0.27297	0.28345	0.0	0.22886	0.26814	0.28383
	G-Mean	0.85237	0.83525	0.82764	0.8471	0.84632	0.83525	0.5	0.66337	0.8496	0.841
yeast	Acc	0.96071	0.94029	0.92266	0.92898	0.90857	0.94029	0.98169	0.97661	0.94326	0.94892
	AUC	0.93125	0.9078	0.89711	0.91492	0.90441	0.9078	0.69299	0.77609	0.92389	0.90577
	F1	0.56842	0.45684	0.39278	0.42155	0.36145	0.45684	0.54846	0.58076	0.47795	0.4919
	G-Mean	0.93125	0.9078	0.89711	0.91492	0.90441	0.9078	0.69299	0.77609	0.92389	0.90577
me2	Acc	0.97121	0.97082	0.97082	0.97082	0.97082	0.97082	0.97121	0.97121	0.97121	0.97121
	AUC	0.5	0.4998	0.4998	0.4998	0.4998	0.4998	0.5	0.5	0.50434	0.51664
	F1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.01667	0.05667
	G-Mean	0.5	0.4998	0.4998	0.4998	0.4998	0.4998	0.5	0.5	0.50434	0.51664
webpage	Acc	0.94545	0.94754	0.9541	0.89896	0.94098	0.94754	0.99046	0.98689	0.93681	0.94933
	AUC	0.85237	0.83525	0.82764	0.8471	0.84632	0.83525	0.5	0.66337	0.8496	0.841
	F1	0.27539	0.28345	0.30411	0.28613	0.27297	0.28345	0.0	0.22886	0.26814	0.28383
	G-Mean	0.85237	0.83525	0.82764	0.8471	0.84632	0.83525	0.5	0.66337	0.8496	0.841
ozone	Acc	0.96071	0.94029	0.92266	0.92898	0.90857	0.94029	0.98169	0.97661	0.94326	0.94892
	AUC	0.93125	0.9078	0.89711	0.91492	0.90441	0.9078	0.69299	0.77609	0.92389	0.90577
	F1	0.56842	0.45684	0.39278	0.42155	0.36145	0.45684	0.54846	0.58076	0.47795	0.4919
	G-Mean	0.93125	0.9078	0.89711	0.91492	0.90441	0.9078	0.69299	0.77609	0.92389	0.90577
level	Acc	0.97121	0.97082	0.97082	0.97082	0.97082	0.97082	0.97121	0.97121	0.97121	0.97121
	AUC	0.5	0.4998	0.4998	0.4998	0.4998	0.4998	0.5	0.5	0.50434	0.51664
	F1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.01667	0.05667
	G-Mean	0.5	0.4998	0.4998	0.4998	0.4998	0.4998	0.5	0.5	0.50434	0.51664
mammography	Acc	0.94545	0.94754	0.9541	0.89896	0.94098	0.94754	0.99046	0.98689	0.93681	0.94933

	AUC	0.93909	0.94016	0.91709	0.90871	0.94341	0.94016	0.83662	0.82158	0.92807	0.94768
	F1	0.42991	0.43949	0.45775	0.28632	0.4142	0.43949	0.75758	0.68571	0.3908	0.45161
	G-Mean	0.93909	0.94016	0.91709	0.90871	0.94341	0.94016	0.83662	0.82158	0.92807	0.94768
abalone 19	Acc	0.72947	0.73377	0.81637	0.84103	0.70193	0.73377	0.99234	0.99234	0.73378	0.86571
	AUC	0.7901	0.77574	0.69368	0.68123	0.77625	0.77574	0.5	0.5	0.79229	0.5804
	F1	0.04316	0.04278	0.04104	0.04441	0.0399	0.04278	0.0	0.0	0.04394	0.0347
	G-Mean	0.7901	0.77574	0.69368	0.68123	0.77625	0.77574	0.5	0.5	0.79229	0.5804
haberman	Acc	0.65022	0.65344	0.64075	0.58129	0.5557	0.65022	0.66667	0.6471	0.62441	0.61731
	AUC	0.48283	0.5251	0.50838	0.51176	0.59006	0.52331	0.4918	0.4895	0.48525	0.52878
	F1	0.16585	0.29101	0.26732	0.30513	0.42634	0.29101	0.16144	0.19964	0.22356	0.29859
	G-Mean	0.48283	0.5251	0.50838	0.51176	0.59006	0.52331	0.4918	0.4895	0.48525	0.52878
glass 5	Acc	0.97641	0.97619	0.97619	0.97165	0.95779	0.97619	0.97641	0.97208	0.97641	0.97641
	AUC	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan
	F1	0.69667	0.69667	0.69667	0.67667	0.60667	0.69667	0.68333	0.58333	0.69667	0.69667
	G-Mean	0.81499	0.81486	0.81486	0.81236	0.80497	0.81486	0.76999	0.67237	0.81499	0.81499
glass 6	Acc	0.95844	0.95844	0.95844	0.94913	0.9539	0.95844	0.98139	0.95801	0.95368	0.95844
	AUC	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan
	F1	0.44	0.44	0.44	0.34	0.44	0.44	0.45	0.0	0.34	0.44
	G-Mean	0.65211	0.65211	0.65211	0.59984	0.64984	0.65211	0.56667	0.35	0.60211	0.65211
glass 7	Acc	0.97641	0.97165	0.95346	0.95779	0.96688	0.97165	0.97186	0.9671	0.97165	0.97165

	AUC	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan
	F1	0.82643	0.8096	0.7396	0.75127	0.79532	0.8096	0.7931	0.7731	0.80643	0.80643	0.80643
	G-Mean	0.84806	0.85515	0.84898	0.82452	0.84265	0.85515	0.81333	0.79667	0.84542	0.84542	0.84542
heart	Acc	0.55097	0.5543	0.5543	0.55108	0.53817	0.55763	0.46215	0.57075	0.55763	0.55763	0.57409
	AUC	0.50648	0.51126	0.51035	0.50863	0.51329	0.51359	0.50541	0.51882	0.51483	0.51483	0.53468
	F1	0.03611	0.08497	0.09249	0.08387	0.28489	0.07247	0.62671	0.18393	0.096	0.096	0.17065
	G-Mean	0.50648	0.51126	0.51035	0.50863	0.51329	0.51359	0.50541	0.51882	0.51483	0.51483	0.53468
ionosphere	Acc	0.94024	0.92603	0.91754	0.91754	0.90889	0.92603	0.92317	0.92889	0.92317	0.92317	0.92603
	AUC	0.92203	0.90564	0.90019	0.90035	0.88309	0.90564	0.90602	0.90679	0.90545	0.90545	0.91044
	F1	0.90642	0.88355	0.87391	0.87441	0.85535	0.88355	0.88045	0.88711	0.8811	0.8811	0.88679
	G-Mean	0.92203	0.90564	0.90019	0.90035	0.88309	0.90564	0.90602	0.90679	0.90545	0.90545	0.91044
pima	Acc	0.65096	0.65487	0.65357	0.65227	0.43865	0.65487	0.39067	0.65227	0.65227	0.65227	0.65747
	AUC	0.5	0.50549	0.50479	0.50368	0.5074	0.50549	0.51813	0.50274	0.50399	0.50399	0.51203
	F1	0.0	0.02118	0.02198	0.02143	0.42517	0.02118	0.47226	0.01429	0.02298	0.02298	0.05677
	G-Mean	0.5	0.50549	0.50479	0.50368	0.5074	0.50549	0.51813	0.50274	0.50399	0.50399	0.51203
Breast	Acc	0.62747	0.62923	0.62747	0.62221	0.62747	0.62747	0.45865	0.62747	0.62393	0.62393	0.66075
	AUC	0.5	0.50185	0.5	0.49669	0.5	0.5	0.5692	0.5	0.49874	0.49874	0.73263
	F1	0.0	0.00714	0.0	0.01459	0.0	0.0	0.5749	0.0	0.01429	0.01429	0.68217
	G-Mean	0.5	0.50185	0.5	0.49669	0.5	0.5	0.5692	0.5	0.49874	0.49874	0.73263