UNIVERSITY OF CAPE TOWN

Modelling Highly Imbalanced Credit Card Fraud Detection Data using Statistical Learning

Student:
Revesa Moodley
MDLREV004

Supervisor:
Mr Stefan S Britz

Minor dissertation for the degree MSc in Data Science

at the

DEPARTMENT OF STATISTICAL SCIENCES

July 24, 2023

Declaration of Authorship

I, Revesa S MOODLEY, declare that this thesis titled, "Modelling Highly Imbalanced Credit Card Fraud Detection Data using Statistical Learning" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date: 10/02/2023

UNIVERSITY OF CAPE TOWN

Abstract

Science Faculty
Department of Statistical Sciences

MSc in Data Science

Modelling Highly Imbalanced Credit Card Fraud Detection Data using Statistical Learning

by Revesa S MOODLEY

Credit card fraud is a major concern for businesses worldwide, yielding losses of up to $67 billion per year in major banks and institutions. Machine learning techniques used to detect fraudulent transactions face several challenges when dealing with highly imbalanced data, which is often the case with fraud detection. Whilst different sampling techniques are generally used to reduce the imbalance, minimal studies have focussed on the effect the level imbalance has on the predictive capabilities of various statistical learning techniques. This study investigates the effect of three factors on model performance: 1) sampling technique, 2) supervised learning method, and 3) prevalence rate, also know as imbalance ratio (IR), which refers to the proportion of majority class samples compared to that of the minority class.

Three sampling techniques are utilised in the study: Random Oversampling (ROS), Synthetic Minority Oversampling Technique (SMOTE), and Random Undersampling (RUS). These methods are used to create varying levels of imbalance in the datatset, at the prevalence rates of 0.2%, 1%, 10%, 20%, 30%, 40%, and 50%. Six supervised learning models are then used to identify fraudulent transactions: Logistic Regression (LR), C4.5 Decision Trees (DT), Random Forests (RF), XGBoost, and Neural Network (NN) models. Precision, recall and F2 score are the primary metrics used to assess model performance.

The results suggest that the ROS and SMOTE sampling techniques performed best in terms of F2 score. The best supervised learning models are RF and XGBoost. The tree models were generally well suited to the imbalanced dataset, whilst LR performed the worst, even when applying regularisation. Increasing the prevalence rate surprisingly yielded a decrease in performance. The findings from the experiments can serve as a foundation for selecting the best sampling technique and supervised learning models to utilize with various degrees of dataset imbalance.

Acknowledgements

I would firstly like to give my warmest thanks to my supervisor, Stefan Britz, for his continued support throughout this thesis. His advice and guidance made this study possible. He has instilled a sense of displine in me which has helped keep me on track with my goals throughout the writing process. My sincere appreciation must also be given to INSETA for providing financial assistance for my MSc studies.

To my family and friends: I will forever be grateful for the motivation, kindness and love you have shown me throughout my degree. Thank you for your constant check ins and the enthusiasm shown towards my study. I am blessed to have you in my life.

To my biggest cheerleaders, my parents, Sada and Kisty, and my sister, Kim: Thank you for being my support system throughout my studies. You have motivated me and provided much needed food, love and laughter in times of need. I love you all.

Lastly, I want to express my gratitude to God, without whom this would not have been possible. Thank You for giving me the knowledge, strength and guidance to complete this thesis.

Contents

List of Figures

List of Tables

List of Abbreviations

AUC	**A**rea **U**nder **C**urve
DT	**D**ecision **T**ree
EDA	**E**xploratory **D**ata **A**nalysis
FR	**F**alse **P**ositive
IR	**I**mbalance **R**atio
LASSO	**L**east **A**bsolute **S**hrinkage and **S**election **O**perator
LR	**L**ogistic **R**egression
NN	**N**eural **N**etwork
RF	**R**andom **F**orest
ROC	**R**eceiving **O**perating **C**haracteristic
ROS	**R**andom **O**versampling
RUS	**R**andom **U**ndersampling
SMOTE	**S**ynthetic **M**inority **O**versamping **TE**chnique
TP	**T**rue **P**ositive
XGBoost	e**X**treme **G**radient **Boost**ing

To my grandmother, who never got to see me through this adventure. I hope you're smiling from the skies above...

Chapter 1

Introduction

The use of credit cards for transacting has been on the increase over the past decade. This is as a result of the drive to introduce more convenient payment methods as well as increasing digitisation, keeping in line with the fourth industrial revolution. Although there are multiple advantages to online and digital methods of transacting, one major drawback is the potential for fraudulent activity. Fraud can be defined as "any activity that relies on deception in order to achieve a gain" (Kamusweke et al., 2019). It has been a major cause of concern as it results in damage to institutions' reputation as well as monetary losses up to $ 67 billion per year in major banks and institutions (Makki et al., 2019). Therefore, it is imperative to introduce fraud prevention and detection methodologies into these organisations. These methods have been known to be notoriously difficult to develop as the patterns to recognise fraud change frequently. In addition, the complexity associated with these patterns can cause fraud investigations to extend over a long period of time, which is not ideal in an environment which requires quick action to apprehend criminal activity (Kamusweke et al., 2019).

Over the years, there have been various solutions implemented to address the concerns faced with fraud, including building of machine learning models. Previous literature has pointed out a major attribute of fraudulent datasets, this being the highly imbalanced nature of the data. Imbalanced datasets contain a small proportion of observations in the class of interest. This introduces inaccuracy into the classification problem. Multiple methods have been investigated to handle the imbalance, such as oversampling and undersampling, which improves the performance of models used to detect fraud. However, not much emphasis has been placed on furthering research into the effect of levels of imbalance in a dataset on model performance. This area will be the primary focus of this study, using a credit card transactions dataset as the basis of the investigation.

1.1 Fraud

There are multiple types of fraud depending on the industry in which it occurs. One of the most common occurrences of fraud is in the financial industry. Financial fraud refers to any fraudulent activity within the following groups: 1) bank Fraud, which encompasses fraud resulting from credit cards, mortgage and money laundering; 2) insurance Fraud, dealing mainly with healthcare and vehicle insurance; 3) corporate Fraud, mainly focusing on

financial statement fraud; and 4) cryptocurrency fraud.

This study focuses on credit card fraud data. This type of fraud is a worldwide issue and has caused major losses in the financial industry. Chaudhary et al. (2012) state that Ukraine has the highest percentage of fraudulent transactions, with a surprising 19% of all credit card transactions being fraudulent. From this information, one can understand the importance of identifying these transactions and preventing any future fraudulent activity from occurring. This type of fraud could be orchestrated in one of the following ways, as indicated by Jain et al. (2019):

1. **Lost or stolen cards**: When the primary cardholder misplaces their card, it may land in the hands of fraudsters. Even though there are security protocols put into place, such as PINs, internet transactions can be simple enough to complete without knowledge of the account holder.

2. **Producing fake or counterfeit cards**: This is typically conducted through the skimming process. Here, a conterfeit card is produced which contains all the information from the original card. This fake card is operational and may be used in transactions for purchases.

3. **Cloning the original merchant site**: A type of fraud where the customer is persuaded into using a fake website which the scammer has designed. This website closely resembles the original merchant website. The fake website generally entices the customer to purchase products or services. Once the transaction has been completed, the cardholder's information is gathered and can be used for future activities.

4. **Application fraud**: If fraudsters gain access to the customer's account details and supporting documents, an application for a credit card can be made. The majority of instances also involves identity theft.

5. **Card not present**: A customer's credit card can be used without it being physically present if a fraudster has knowledge of the card information, such as account number, expiry date, and card verification code (CVC).

Detection of fraud remains a complex task. It is a combination of various factors, such as: behavioural traits of a customer, their spending habits, taking into account any previously committed frauds, and observing their spending patterns (Jain et al., 2019). This translates into multiple parameters which need to be considered when determining whether a transaction is fraudulent or not. Therefore, fraud detection mechanisms need to be able to process large amounts of parameters in datasets.

Traditional fraud detection methods tend to be costly and time-consuming. The introduction of machine learning techniques have addressed a various drawbacks arising from traditional methods. Fraud detection systems should be designed to meet the following requirements: 1) accurate identification of fraudulent transactions, where false positives should be at a minimum; 2) genuine transactions should not be classified as fraudulent (false negatives at a minimum); and 3) since detecting fraud is time sensitive, the system should be able to detect transactions as close to real time as possible (Jain et al., 2019).

The nature of fraudulent data also should be taken into account when designing models. The data has been known to be imbalanced, whereby most of the transactions are classified as genuine, with only a small amount being fraudulent. The topic of imbalance will be discussed briefly in Section 1.2.

1.2 Imbalanced datasets

Imbalance, in the context of a classification problem, relates to a scenario where the majority of observations belong to one class, which affects the detection of the minority class in the dataset. There have been many studies conducted to address this issue. The methods developed differ based on the complexity of the problem at hand. The commonly used solutions include a data-level approach (e.g. oversampling or undersampling), algorithm-level approach (e.g. cost-sensitive learning) or one-class classification methods (e.g. training only on minority class) (Makki et al., 2019). These methods will be discussed in greater detail in the subsequent chapters of the dissertation, specifically in Chapter 2. For this specific investigation, a data-level approach was taken to create various levels of imbalance within the dataset.

1.3 Research aims and objectives

There have been various studies conducted over the recent years delving into the detection of fraud using machine learning capabilities, with the majority of them focusing on determining which algorithms result in the highest performance using the fraud dataset on hand. The imbalanced nature of fraud datasets is an important factor to consider when building models for detection. Previous literature has catered for this in various ways, such as implementing sampling techniques. However, limited focus has been placed on the effect that the level of imbalance in the re-sampled dataset has on the performance of models.

The main aim of this study is to investigate the effect various levels of imbalance has on the performance of various supervised learning models. Different sampling techniques were also explored to determine the highest performing method based on a credit card fraud dataset. Logistic regression, C4.5 decision tree, random forest, XGBoost and neural network models were included in the study, which were chosen based on their popularity in previous fraud detection studies.

A secondary aim is understanding the effect of threshold values when assigning class labels during the modelling process. As a default, the value of 0.5 was used. This meant that if the probability of the class prediction was greater than 0.5, it would be labelled as a fraudulent case and vice versa for non-fraudulent cases. The investigation was completed on the logistic regression model with different threshold values. A comparison of performances between each of the values was conducted to form a conclusion.

In order to meet the above objectives, the following steps were taken:

1. Data preparation

2. Exploratory data analysis
3. Creating datasets with various levels of imbalance using random oversampling (ROS), synthetic minority oversampling technique (SMOTE) and random undersampling (RUS)
4. Applying logistic regression (with regularisation), C4.5 decision tree, random forest, XGBoost and neural network algorithms to datasets
5. Tuning hyperparameters during model fitting
6. Fitting models to various datasets produced from sampling methods
7. Evaluating and comparing performance of algorithms using the primary metrics of precision, recall and F2 score. Area under the curve (AUC) was also investigated as an additional metric

The outcome of this study would allow better understanding into the best performing algorithms to implement on imbalanced datasets, based on the degree of imbalance.

1.4 Format of thesis

This thesis has six chapters and two appendices. These chapters contain an introduction, explaining the contents of the chapter, a body and a conclusion, which summarises the main points of the chapter.

Chapter 1 consists of an overview of the topic of research investigated in this thesis. The chapter provides a background to the study, specifically in a fraud context, as well as the goals and objectives of the investigation.

Chapter 2 provides an insight into imbalanced datasets and strategies to overcome the imbalance problem. Previous studies completed on fraud detection are mentioned in this chapter.

Chapter 3 gives the reader information on the data used, including the source of the data, data characteristics, and important findings during exploratory data analysis.

Chapters 4 focuses on the methodology followed to meet the research objectives. Information around the sampling techniques, supervised learning models and prevalence rate percentages will be explored in this chapter.

Chapter 5 discusses the results obtained from the study. A comparison between the models as well as the imbalance percentages will be shown, together with key findings from the experiments. The focus will be placed on the performance of the models and comparing each to determine the best model to use for imbalanced datasets.

Lastly, the study's main findings are summarised in Chapter 6. These results are presented in relation to the research questions and objectives. Additionally, suggestions are given on how to enhance the study for potential future studies. Concluding remarks are added to complete the investigation.

Chapter 2

Review on Handling of Imbalanced Data

2.1 Introduction

In recent years, the issue of class imbalance in real-world problems has drawn considerable attention. In the case of binary classification, a dataset is referred to as imbalanced when one of the classes is significantly underrepresented in comparison to the other class. Classifiers in this situation may perform well on the majority class but perform poorly on the minority class. Generally, the minority class is the one of interest and is often assumed to be anomalies or noise, thereby resulting in significant misclassifications. This increases the need for methodologies to improve detection capabilities when dealing with such cases.

Due to the impact resulting from imbalanced datasets in machine learning, there has been a significant amount of research done on the subject. Visa and Ralescu (2005) have explained the reasoning behind poor performance of classification algorithms, which can be summarised into the following three broad categories:

1. Classifiers are focused on maximising the overall accuracy of the predictions. With imbalanced datasets, the minority class contributes an insignificant amount to this evaluation metric.
2. Classifiers implicitly presume that data are distributed equally across all classes.
3. Classifiers assume that misclassifying errors resulting from each class have an equal cost.

In order to combat these issues, two solutions have been developed, referred to either as an internal strategy or an external strategy (Ramyachitra and Manikandan, 2014). The internal strategy, also known as the algorithmic level approach, develops new algorithms or modifies the current ones taking into account the class imbalance problem. This approach is algorithm specific, which is the major disadvantage, requiring more to effort to be conducted at the model building stages of the pipeline. The external strategy, as known as the data level approach, introduces a pre-processing step into the pipeline in order to reduce the impact resulting from the class imbalance. It allows for more flexibility as it is not classifier specific. This research will focus on this strategy, where the data will be pre-processed and investigated to determine its effect on the performance of the models.

In most cases, accurately predicting the minority class is a higher priority, therefore requiring an approach which considers the difficulties that come with modelling using imbalanced datasets. In this study, it is clear to see how this concept is applied into the practical application of credit card fraud detection. In this instance, the timeous detection of fraudulent activity could assist in highlighting criminals and enforce justice, as well as preventing criminal activity in future. If a transaction is labelled as a non-fraudulent transaction but is actually fraud, the financial repercussions could be severe.

The topic of imbalanced datasets spans across many industries and applications such as cyber security, which focuses on prevention of cyber-crime and malicious activity; rare disease detection and social network monitoring (Krawczyk, 2016).

This chapter is organised in the following manner: Section 2.2 discusses the characteristics of imbalanced datasets, focusing on the parameters of small disjuncts, lack of density, class overlapping and distribution of data. Section 2.3 explains the challenges imbalanced datasets bring into data modelling and machine learning, looking at the underlying mechanisms of existing algorithms and downfalls of them in relation to the imbalance problem. Section 2.4 shows an overview of the solutions to tackling imbalanced datasets, broken down into data-level, algorithmic, and cost-sensitive learning methods. Section 2.5 describes the metrics that should be adopted when evaluating the performance of algorithms using imbalanced data. Lastly, Section 2.6 gives a summary of significant studies conducted over the past decade to combat multiple imbalanced dataset real-world problems.

2.2 Characteristics of imbalanced datasets

Class imbalance, as mentioned above, occurs when there are significantly less observations in one class over the other. This can manifest in two ways: 1) when the data follow the characteristics of imbalance naturally, such as in datasets relating to rare disease detection, or 2) when the data are not naturally imbalanced, but rather involves high cost implications, confidentiality issues or there is a large amount of work required to retrieve the data (Ali et al., 2015). An example of such a dataset could be in the event of failure of a space-shuttle. The complications with class imbalance comes in the form of four factors: small disjuncts, lack of density, class overlapping, and distribution of data. These concepts are explained in the below subsections.

2.2.1 Small disjuncts

Identifying whether a dataset has a distinct imbalance between classes is a task which can be completed with ease by comparing the ratio between the minority and majority classes. However, determining if there is an imbalance within a class is not as obvious to detect. This phenomenon is referred to as small disjuncts, where the class consists of several sub-groups with differing amounts of observations (Japkowicz, 2001; Prati et al., 2004; Weiss, 2010). Figure 2.1 shows an example of a small disjuncts in a dataset in a classification problem. Figure 2.1a shows the imbalance between classes, while 2.1b shows the small disjunct phenomenon in a class imbalance context.

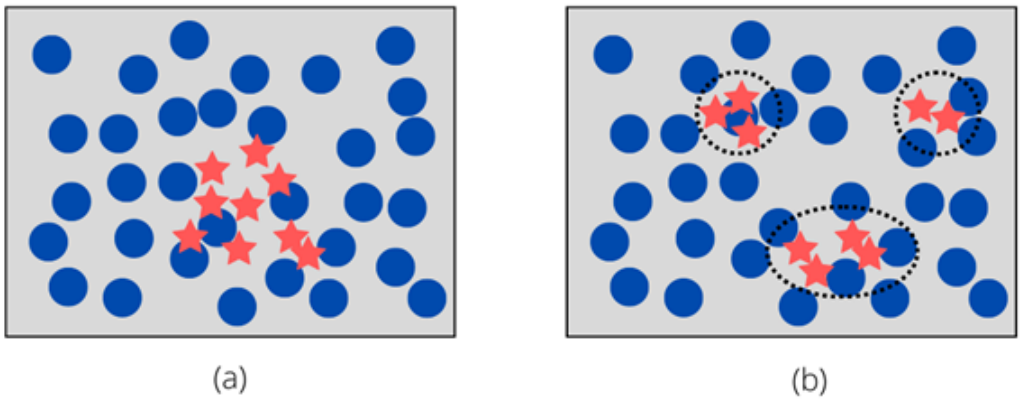

FIGURE 2.1: a) Depiction of between-class imbalance; b) Depiction of small disjuncts

This characteristic becomes prominent when applying classification algorithms, which use the core principle of "divide and conquer", such as decision trees, where the data are partitioned into many groups of data with a few instances of the minority class (Weiss, 2004).

It has been reported that the classification performance is impacted by the following factors: 1) Small disjuncts make it difficult for classifiers to understand concepts from the minority class; and 2) the majority of current approaches primarily focus on finding solutions for the between-class problem rather than the imbalance within classes (Sun et al., 2009).

Even though this topic has not been researched extensively, there have been several studies conducted by Weiss and Provost (2003) and Jo and Japkowicz (2004a) on small disjuncts which have suggested that a guided oversampling technique can be used to improve the representation of the observations within the minority class.

2.2.2 Lack of density

One of the most common challenges when it comes to class imbalance lies in the sample size of the training dataset. This ties back to the concept of "lack of density" which, simply put, means that there is an inadequate number of observations to highlight patterns and make generalisations on the data (Ramyachitra and Manikandan, 2014).

The impact of sample size on the performance of classifiers with imbalanced datasets can be seen in Figure 2.2. In Figure 2.2a, there is a satisfactory number of examples available in both classes for classification. The model predicts an estimated decision boundary (dashed line) fairly similar to that of the true boundary (solid line). This results in more observations being accurately predicted. On the contrary, upon evaluation of Figure 2.2b, there is a small number of observations and a highly imbalanced dataset. When implementing classification techniques to this data, the estimated decision boundary only covers a small area of the true boundary, thus will lead to a high error rate.

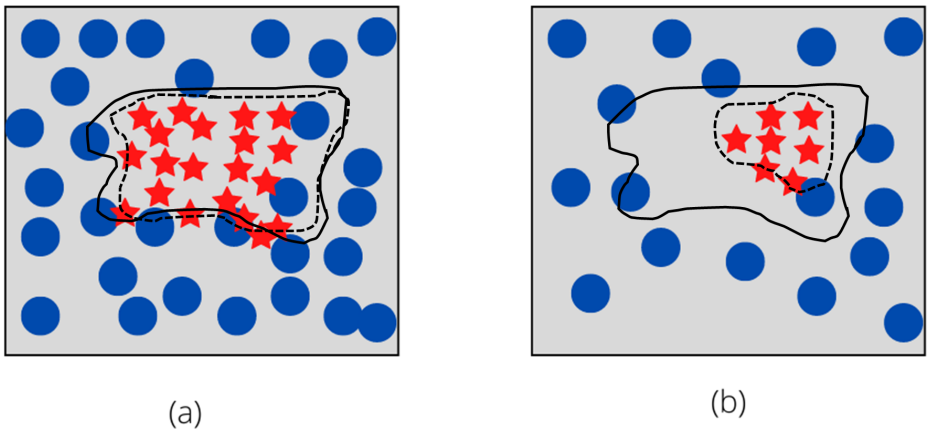

FIGURE 2.2: a) Image showing the impact of a large sample size on predictive capability of a classifier; b) Image showing the impact of a small sample size on predictive capability of a classifier

2.2.3 Class overlap

When considering class imbalance problems, an important factor in the initial investigations would be to understand if the data contains class overlapping. Class overlapping, also known as class complexity or class separability, describes how separable the classes are from one another within a dataset (see Figure 2.3 for depiction). Depending on the level of overlapping between the minority and majority classes, the predictive capabilities for the minority class could be severely affected. It is quite challenging to define rules to differentiate between classes when overlapping is present in each class for a certain feature space, or occasionally even in all feature spaces.

From previous studies, a directly proportional relationship was found between the level of data complexity and the ability of a classifier to make generalisations on the data (Japkowicz and Stephen, 2002). For standard classification algorithms, the result of this discovery brings to light the downfalls of overlapping classes. The focus lies in maximising the overall accuracy which leads to the overlapping region being labelled as the majority class and the minority class being regarded as noise and excluded (Weiss, 2004).

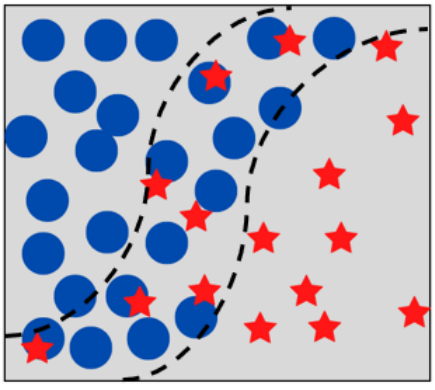

FIGURE 2.3: Image depicting class overlapping

2.2.4 Dataset shift

Dataset shift occurs when the training and test datasets follow different distributions. This is a common issue faced when using classification algorithms and is a consequence of sample selection bias (Fernández et al., 2011). The dataset shift issue generally has little influence on real-world problems with balanced datasets. However, when dealing with imbalanced datasets, this problem becomes particularly significant as there are minimal minority class samples thereby affecting single classification errors.

2.3 Modelling challenges caused by imbalanced datasets

The performance of a model is highly dependent on the characteristics of the dataset. Given that conventional machine learning algorithms assume a balanced dataset, it is expected that there are some downfalls when introducing class imbalance. There are several factors that can contribute to the current classification algorithms' subpar performance on imbalanced datasets. Most of these have been discussed in Section 2.2, however this section will highlight the main reasons for these challenges.

Firstly, one of the major issues is the fact that most standard algorithms are accuracy driven. This means that they strive to reduce the overall error, to which the minority class makes up a very small amount. Therefore, the choice of an accuracy metric to determine performance would lead to misleading information. Another factor which causes significant concern is the assumption that the data are distributed equally across all the classes (as seen in Section 2.2.4). Many classification methods provide erroneous model representations in classification tasks because they ignore the underlying distribution of the datasets, causing a decline in model performance. A majority of learning algorithms are built on the assumption that training sets have a well-balanced distribution, which is frequently false, according to studies conducted by Japkowicz (2000) and Japkowicz and Stephen (2002).

Furthermore, although many existing classifiers assume the error costs from different classes are similar, research conducted in applications such as rare disease diagnosis and fraudulent activity detection by Visa and Ralescu (2005) and Sun et al. (2009) prove that the error costs vary significantly. With proper insight on costing as well as class distribution, modelling challenges could be minimised. However, due to the complexity of such tasks, other methodologies have been developed to overcome the issues that imbalanced datasets bring into classification models, which are discussed in the next following section.

2.4 Solutions to the imbalance problem

Previous studies have delved into the imbalance problem and have suggested multiple ways of overcoming the modelling challenges. One such method involves dealing with the data intrinsically by using resampling methods or pre-processing of the data to introduce a more balanced dataset into the classification problem. This is referred to as the data-level solution where only the data are altered and no enhancements to the algorithm itself are made (Patel et al., 2020). The other approach to the imbalance problem is the algorithm-level solution,

which aims to introduce modifications into the classification algorithm itself to bring about in correct classification results (Patel et al., 2020). Research into other techniques have been extensively conducted, which include cost-sensitive approaches and ensemble techniques.

Cost-sensitive learning combines the techniques from both the algorithmic and data-level approaches. It aims to reduce the cost of misclassifying the minority class, which drives the performance of the algorithm (Sun et al., 2007; López et al., 2013). Ensemble techniques could be introduced into imbalance problems in one of two ways, either by altering the ensemble learning algorithm during the data pre-processing steps before classification occurs, or by incorporating a cost-sensitive approach to the ensemble learning process (Ting, 2000; Chawla et al., 2003; Sun et al., 2007; Błaszczyński et al., 2010). The majority of machine learning algorithms perform better when using balanced datasets, therefore it is imperative to consider the characteristics of the dataset prior to modelling. Using this as a basis for this study, this section will focus on data-level approaches to the imbalance problem. The other methodologies will be discussed briefly to provide context to the available solutions currently being used.

2.4.1 Data-level approaches to imbalanced learning

The data-level approach, as described in the above paragraph, targets the imbalance problem by either undersampling or oversampling to reduce the imbalance ratio between the classes. Undersampling refers to the removal of observations within the majority class, either randomly or specific to the areas near the decision boundary, to reduce the discrepancy between the classes (Batista et al., 2004). Oversampling focuses on the minority class where the purpose is to duplicate or create new observations within the class, either randomly or from areas away from the decision boundary (Batista et al., 2004). This will result in a more evenly balanced dataset and improve classifier performance.

2.4.1.1 Undersampling

Undersampling, sometimes referred to as "downsizing", is a useful technique that has been used in the context of the class imbalance problem. In this approach, the classifier is trained using a subset of the majority class. By virtue of the fact that observations from the majority class are ignored, the dataset leans towards a more balanced one while reducing the time taken to complete the training process. Undersampling can be done in a variety of ways, but this subsection will focus on three of the more popular ones: Random Undersampling (RUS), Edited Nearest Neighbours (ENN), and Tomek Links.

Random Undersampling Random undersampling (RUS) is the most popular pre-processing non-heuristic technique, which is generally used as a baseline method to tackling the imbalance problem (Fernández et al., 2018). In this method, data points from the majority class are chosen randomly and are removed from the dataset until an equal ratio between majority and minority classes is reached (Bach et al., 2019). The fundamental disadvantage of undersampling lies in the fact that it removes the potentially helpful information which was included in the eliminated observations.

Edited Nearest Neighbour Edited Nearest Neighbour (ENN) is an undersampling technique which uses the concept of k-Nearest Neighbour to decide whether to discard the data point from the majority class or not. In this particular case, if an example in the majority class is surrounded by predominantly minority class neighbours, the example would be removed from the dataset (Beckmann et al., 2015).

Tomek Link The Tomek Link (T-link) method is an undersampling method using a distance measure as the basis of eliminating data points from a dataset. In T-link, a distance measure is created between two examples of the majority and minority class. This measure is then utilised to eliminate instances of the majority class (Tomek, 1976). To better understand the concept, Fernández et al. (2018) have explained it as follows:

1. Consider two examples: A_i and A_j where A_i can be represented as (x_i, y_i) and A_j can be represented as (x_j, y_j).
2. The distance between A_i and A_j is D which can be represented as $\mathrm{d}(A_i, A_j)$.
3. A pair (A_i, A_j) can be said to have Tomek Link if there is not an example A_l such that $\mathrm{d}(A_i, A_l) < \mathrm{d}(A_i, A_j)$ or $\mathrm{d}(A_j, A_l) < \mathrm{d}(A_i, A_j)$.
4. After locating Tomek Links, the majority class instance is eliminated, while examples from the minority class are retained in the dataset.

If data points form a T-link, they could be said to either be noise or could be on the decision boundary between the two classes and therefore should be removed. The Tomek Link method can be used as an undersampling technique, where the majority class is removed, or in data cleaning where both examples are removed (Bach et al., 2019).

2.4.1.2 Oversampling

This subsection focuses on techniques for oversampling, which operate in the opposite way from undersampling. The purpose of oversampling is to raise the number of observations in the minority class to the same level as the majority class instance count. Oversampling, also known as "upsizing", increases the minority class as a result. This subsection includes two techniques which are frequently used: Random Oversampling (ROS) and Synthetic Minority Oversampling Technique (SMOTE).

Random Oversampling The simplest way to introduce more observations into the minority class is through random oversampling (ROS). This is a non-heuristic technique that balances the class distribution by the random replication of observations in the minority group (Santoso et al., 2017). The unfortunate drawback of this method is that ROS can lead to overfitting problems. It was also highlighted by Visa and Ralescu (2005) that the duplication of observations in the dataset also brings about an increase in learning time. This method is more frequently used when compared to undersampling, as undersampling results in important data being lost due to elimination of observations. To address this issue, Chawla et al. (2002) have proposed an over-sampling approach called Synthetic Minority Oversampling Technique (SMOTE).

SMOTE This method is a more sophisticated way of dealing with imbalanced datasets, which does not simply duplicate existing data points, but rather creates synthetic data points near the existing minority class points using k-Nearest Neighbours (k-NN). The procedure for generating synthetic data points using SMOTE is as follows:

1. A random sample from the minority class is picked.
2. The k-Nearest Neighbours are identified from the sample drawn.
3. One of the neighbours is selected and the vector between this point and the existing data point is identified.
4. The vector is multiplied by a randomly selected number between 0 and 1.
5. Add the calculated vector value to the existing data point to find the synthetic point.

This methodology ensures that the synthesised data points are not the same as the existing data, however still in the general vicinity of an expected observation within the minority class. In contrast to what is normally seen when using ROS, the synthetic points force the classifier to create bigger and more generalised decision areas to produce better performing algorithms (Chawla, 2005).

Numerous changes to the original SMOTE algorithm have been suggested in the literature. Although the SMOTE technique was generalised to support mixed datasets of continuous and categorical characteristics, it does not handle data sets with all categorical features. Synthetic Minority Oversampling Technique Nominal Continuous (SMOTE-NC) and Synthetic Minority Oversampling Technique Nominal (SMOTE-N), both proposed by Chawla et al. (2002), are extensions of SMOTE that can additionally include categorical features. Since the introduction of SMOTE into the sampling techniques, there have been a wide variety of studies conducted to improve and enhance the method. One such proposal includes the borderline-SMOTE1 and borderline-SMOTE2 by Han et al. (2005), which only oversample data points close to the decision boundary. These strategies outperform the original SMOTE technique and ROS in terms of the True Positive (TP) rate and the F-value.

Besides data-level approaches, there are a variety of other methodologies implemented in classification imbalance problems, which will be briefly explained for completeness of information. As mentioned previously, the algorithm-level and hybrid methods are commonly used in real-world applications.

2.4.2 Algorithm-level approaches to imbalanced learning

The focus of this approach is to create or modify current classifiers to minimise their bias towards the majority class. For this approach to be effective, it would require a good understanding into the algorithm architecture and pinpointing reasons for its poor performance towards imbalanced datasets. The most popular branch of this type is the cost-sensitive learning technique (Zhou and Liu, 2010). For this approach, the classifier chosen would be modified to include a penalty for each class being categorised. This will result in higher costs being assigned for misclassified examples in the minority class when compared to the majority class, thereby increasing the importance of the less represented class

(Krawczyk, 2016). The challenge in this comes with application in real-world problems, where the cost matrix is seldom provided.

Another commonly used approach is the one class learning method. In this strategy, the algorithm recognises examples which belong to a certain class while rejecting the rest. In the case of imbalanced data, it would retrieve the examples solely from the minority class while learning and ignores the other examples; the algorithm then follows the same method for the majority class (Ali et al., 2019). In this way, the learning is unbiased with little influence from the nature of the dataset.

Lastly, the ensemble learning method has been a prominent topic within the algorithm-level solution. Here, existing classifiers are combined to create a new classifier which performs better than each individual one (Kittler et al., 1996). Single classifiers cannot resolve the imbalance class problem on their own, therefore the idea is to make use of ensembles of classifiers to improve the accuracy of a single classifier in machine learning (Galar et al., 2012). Various techniques and methodologies can be employed to create ensemble learning algorithms. Boosting, Adaboost and Bagging are three of the most well-known ensemble learning algorithms.

An extension of these approaches comes in the form of a hyrid method. These methods focus on combining the above-mentioned approaches of algorithm and data-level techniques to highlight their strengths and minimize their drawbacks. It is very common to combine data-level solutions with classifier ensembles, which produce reliable and effective classifiers (Wozniak, 2013).

2.5 Evaluation metrics for imbalanced datasets

Due to the nature of imbalanced classification problems, it was noted that the traditional metric of overall accuracy is not an adequate measure of model performance (Wang and Mendel, 1992; Japkowicz and Stephen, 2002). Specific assessment metrics and approached have been proposed for these applications. The confusion matrix and its resulting measurements will be used to provide insights into the classification models. These evaluation metrics are explained in the subsequent sections below.

2.5.1 Confusion matrix

The confusion matrix is a commonly used measure for binary as well as multi-class classification problems. The simplest version of a confusion matrix is for a binary-class problem, where there are four results presented in a table (see Table 2.1). The outputs of the model will be reported as true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN). In our application, the positive refers to the minority class, while the negative refers to the majority class.

		Predicted	
		Positive	Negative
Actual	Positive	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

TABLE 2.1: Binary-class problem confusion matrix (adapted from Ali et al. (2015))

Following the example in Table 2.1, the binary classification would result in predicted values, which would be compared against the actual values to determine the performance of the model. The inputs into the matrix are seen below:

- True Positive (TP): The number of actual positive observations which are correctly predicted as positive observations by the model.
- True Negative (TN): The number of actual negative observations which are correctly predicted as negative observations by the model.
- False Positive (FP): The number of actual negative observations which are incorrectly predicted as positive observations by the model.
- False Negative (FN): The number of actual positive observations which are incorrectly predicted as negative observations by the model.

These values alone will not be sufficient in understanding the performance of a model. Therefore, other metrics have been developed to evaluate the classifier, which have been derived from the results of the confusion matrix.

2.5.2 Accuracy

As discussed earlier, accuracy is the most commonly used metric in classification problems. In terms of the confusion matrix, this value represents the true results (positive and negative) in comparison to the population. The formula for accuracy is:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.1}$$

Generally, for imbalanced datasets, accuracy alone does not provide a holistic representation of the performance of a classifier. Therefore, other metrics are useful in providing more insight into the results.

2.5.3 Precision and Recall

Precision and recall are frequently used classification metrics. Precision, also known as "positive predictive value", demonstrates how well the model predicts positive values. It is the measure of correctness and informs of the number of correct positive classifications in comparison to all the positive values obtained (both true and false).

$$Precision = \frac{TP}{TP + FP} \tag{2.2}$$

Recall, also known as "sensitivity", can be used to evaluate a model's robustness. This measure describes the ability of the model to recognise the positive class as such.

$$Recall = \frac{TP}{TP + FN} \tag{2.3}$$

These two metrics are closely associated and provide useful information. However, for improved model performance, it is imperative to increase the recall rate while still ensuring that the precision is not affected severely.

2.5.4 Geometric mean (G-mean)

The geometric mean (G-mean) is a measure which aims to highlight how well a classifier predicts the positive class in comparison to the negative class (Ali et al., 2015). This is determined through the precision and recall rates. If the value for the G-mean is low, this indicates that the model is biased to one of the classes. The formula to calculate this measure is:

$$\text{G-mean} = \sqrt{Precision \times Recall} \tag{2.4}$$

2.5.5 F-score (F-value)

Recall and precision provide important insights into how a model is performing. However, these measures alone cannot be used to evaluate a model's performance. This is where F-score plays a pivotal role, combining both recall and precision to convey these results in a single metric. In order to reach optimal performance, the recall is improved while maintaining a good precision. The formula for F-score is:

$$\text{F-score} = \frac{\left(1 + \beta^2\right)\left(Precision \times Recall\right)}{\left(\beta^2 \times Precision\right) + Recall} \tag{2.5}$$

In this equation, β refers to the relative importance of precision compared to recall in terms of a weight (Santoso et al., 2017). This weight is commonly set to 1 relating to an equal importance between both measures and is generally referred to as the F1-score. This F-score ranges from 0 to 1, where 0 represents a poor performance and 1 represents the best performance possible.

The F-score was used as an important evaluation metric in this study based on the objectives set out, which is the detection of fraudulent transactions using a credit card dataset. Importance should be placed on optimising the recall as the cost of a false negative is high in this application (or the model predicting a case as non-fraud when it is fraud). Using this knowledge, a β value of 2 was chosen to evaluate the models, resulting in the F2 score. Equation 2.5 then becomes the following:

$$\text{F2 score} = \frac{5 \times (Precision \times Recall)}{(4 \times Precision) + Recall} \tag{2.6}$$

The F2 score was favoured over the F1 score as it gives more weighting towards recall than precision, which is in line with the case study objectives.

2.5.6 Receiver Operating Characteristic (ROC) Curve

In addition to the evaluation metrics discussed above, there is a visual representation of these measures for binary classification problems in the form of a Receiver Operating Characteristic (ROC) curve. The graphical depiction is a probability curve which shows the false positive rate (FPR) plotted on the x-axis and true positive rate (TPR) on the y-axis at various threshold values (Kulkarni et al., 2021). These thresholds range from 0 to 1 and generate corresponding pairs of FP and TP measures. In a perfect classifier, the TPR would be 1 and a FPR would be 0. Therefore, when plotting the curve, a good model should generate points towards the upper left quadrant of the plot, as seen in Figure 2.4. The diagonal dotted line represents the performance of a classifier if it was randomly guessing outcomes (Fernández et al., 2018). If the ROC curve lies below the diagonal line, this would imply that the model is performing worse than random guessing, which is counter-productive in this instance (Kulkarni et al., 2021).

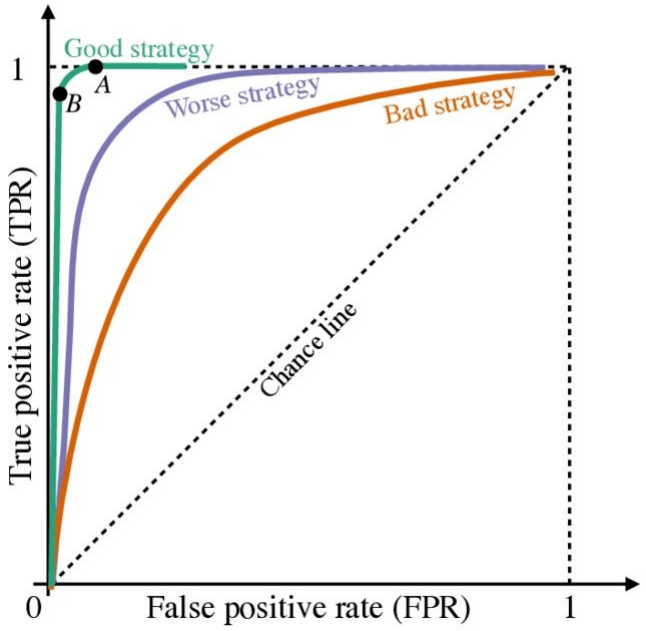

FIGURE 2.4: Example of a ROC curve (Gwirtz et al., 2021)

Together with the ROC curve, the Area Under Curve (AUC) score is commonly calculated, representing the performance of a classifier. This value lies between 0 and 1, where 1 is a perfect model. If a classifier lies below the diagonal line in the lower quadrants, the ROC curve will have an AUC score below 0.5. Likewise, if a classifier lies above the diagonal line in the upper quadrants, the AUC score will be above 0.5.

2.6 Related Works

This section is dedicated to presenting an overview of the most relevant studies which have been published on learning with imbalanced data as well as the pre-processing techniques utilised. Since this research focuses on data-level approaches, studies based on sampling techniques will be discussed here. Since these techniques do not depend on learning algorithms, they are more flexible than the other methods for addressing class imbalance issues. This will form the basis of Section 2.6.1, with the important findings summarised in Table 2.2. Section 2.6.2 will summarise real-world applications of these techniques and provide a brief introduction into the industries generally affected by imbalance problems .

Haixiang et al. (2017) have provided a comprehensive summary of rare event detection and imbalanced classification techniques, and the reader is urged to refer to this publication for more information on the topic.

2.6.1 Data-level approaches to imbalanced learning

A thorough analysis was conducted by Van Hulse et al. (2007) using 35 real-world datasets, 11 classifiers and 7 pre-processing techniques. Their research is the first of its kind to offer a practical approach to building classification algorithms from imbalanced datasets and to provide guidance for future studies of a similar nature. The aim of the study is to analytically assess class imbalance from a broad perspective, comparing classifiers, sampling strategies and performance metrics using a variety of datasets. The datasets utilised in the study ranged from highly imbalanced (1.33%) to slightly imbalanced (35%). The datasets were grouped into four buckets based on the severity of imbalance or the *Imbalance Ratio (IR)*.

$$IR = \frac{Number\ of\ minority\ class\ observations}{Total\ number\ of\ observations} \tag{2.7}$$

RUS, ROS, one-sided selection (OSS), cluster-based oversampling (CBOS), Wilson's editing (WE), SMOTE (SM) and borderline-SMOTE (BSM) were used to adjust the imbalance in the datasets. RUS was completed at 5%, 10%, 25%, 50%, 75% and 90% in relation to the majority class. For ROS, SM and BSM, their rates of oversampling were performed at 50%, 100%, 200%, 300%, 500%, 750% and 1000%. The results from the investigation showed that for data with a high level of imbalance of below 5% to 10%, RUS performed better than other methods, according to the authors. Looking at the results in detail, it was observed that datasets with an IR of less than 5% had RUS as the highest performing method when using AUC and F-score as evaluation metrics. In contrast to other studies, they asserted that the simpler random sampling techniques frequently outperformed more complex resampling techniques such as CBOS and OSS.

Another study by Batista et al. (2012) based their studies on designing a methodology to assess how class imbalance at various degrees affected the performance of selected classifiers. 20 datasets were used in the investigation, with 7 classifiers and 2 sampling techniques utilised. The models included a combination of decision trees, decision rules, neural networks, support vector machine (SVM) and Naïve Bayes algorithms. ROS and SMOTE were chosen to complete the sampling. AUC was used as the primary metric to evaluate the models' performance. A

loss measurement was developed to combine the AUC values at an imbalance state versus at a balanced state, seen below:

$$L = \frac{B - T}{B} \tag{2.8}$$

where B is the AUC measure using the balanced representation of the data (50/50) and
T is AUC measure using the imbalanced representation of the data.

The authors stated that the class imbalance issue affected all examined classification algorithms investigated. It was concluded that when the minority class represents 10% or less of the dataset, there are considerable performance losses due to imbalance. With the exception of SVM, all classifiers exhibit some performance loss for all degrees of imbalance presented in the datasets. In the case of SVM, the performance loss was more prominent in the extremely imbalanced distributions. As the classes get more unbalanced, this loss tends to become more obvious.

Through artificial rebalancing of the data, the performance loss caused by class imbalance was recovered through oversampling techniques. It was noted that random oversampling outperforms SMOTE by a significant margin, except for the Naïve Bayes classifier. It is important to note that SMOTE requires significantly more computational effort than random oversampling. As a result, the performance boost from using a more complex sampling method comes at the expense of more computational time. A range between 50% and 60% of the performance lost could occasionally be recovered using the oversampling techniques that were tested. However, the performance recovery was below 30% for the majority of the tests, which is regarded as a relatively poor recovery rate.

A study conducted by Batista et al. (2004) analysed 10 different sampling methods to understand its ability to overcome the problem of class imbalance. In this, 3 new methods of sampling were proposed and compared to existing techniques. To complete this investigation, the authors chose 13 datasets, which spanned across a variety of different imbalance degrees ranging from 2.6% to 34.7% imbalance rates. The C4.5 learning algorithm was used to complete the model development, for the imbalanced dataset, where the AUC was calculated to determine a baseline for comparisons post application of a sampling technique. The datasets were balanced by addition or removal of observations. The two methods proposed in this particular study were SMOTE + Tomek Links and SMOTE + ENN, which resulted in good AUC performance for the datasets which have a high imbalance degree. Additionally, random oversampling, which often is regarded as an unprofitable method, produced outcomes that were on par with those of the more sophisticated techniques. SMOTE + Tomek Links and SMOTE + ENN may be used as a general recommendation for datasets with a small number of positive cases. The random oversampling method, which is computationally less expensive than its counterparts, might give significant findings for datasets with a greater number of minority class observations.

Japkowicz and Stephen (2002) delved into research which aimed to uncover the nature of imbalanced datasets and how to overcome the challenges relating to it. This was done by comparing a variety of strategies proposed to address the problem and evaluating how the class imbalances affected different classification algorithms. Simulated datasets were

generated to conduct these experiments at various levels of imbalance. A major focus of this study also included the relationship between data characteristics and model performance. The problem of small disjuncts and the size of the training data were explored. The authors concluded that when a dataset is highly imbalanced and contains a small number of observations, the more complex the problem and the greater the effect on the performance of the classifier. Three types of algorithms were investigated; C5.0, multilayer perceptrons (MLP) and support vector machines. C5.0 was shown to be the most sensitive to imbalanced datasets, followed by MLP, and lastly SVM which showed to be the least impacted by the level of imbalance of the dataset. For the C5.0 algorithm, three sampling methods were introduced into the study. It was found that random oversampling resulted in a greater performance when compared to undersampling and cost modification methods.

It should also be noted that in another study by Jo and Japkowicz (2004b), a comprehensive study was done examining the effect that small disjuncts has on the performance of classifiers in the context of imbalanced problems. The outcome of the experiments showed that the loss in performance was mainly due to the small disjuncts problem versus that of a class imbalance problem. In order to address this, the authors suggested a cluster-based oversampling method to overcome both the class imbalance and small disjuncts issue in datasets.

Napierala and Stefanowski (2016) extensively researched the topic of data factors which may negatively influence the learning from imbalanced data. 26 datasets were chosen, using previous studies in the field as a basis for investigation. These represented different levels of imbalance, data sizes, areas of application and included features of a continuous and categorical nature. 6 classification algorithms were employed and 4 different sampling techniques (more information on study design in Table 2.2). In order to evaluate the models, the sensitivity, geometric mean and F-measure metrics were used. In contrast to the previous studies in the field, the AUC was not considered as the classifiers chosen for investigation resulted in deterministic predictions, whereas AUC gives a better reflection of performance for probabilistic or scoring classifiers. The results from the experimentation showed that using IR or size of the dataset is insufficient to account for the variations in performance of the algorithms. In this study, it was found that the datasets with a high IR performed better than those with a lower IR. In terms of data size, the larger the dataset, the more difficult the learning process.

This provides an interesting conclusion, which was also seen in other studies such as Jo and Japkowicz (2004b), that the IR of a dataset is not the main cause of poor classifier performance for imbalanced problems. From the algorithms chosen to investigate, it was observed that the RBF and SVM classifiers perform well on datasets which include borderline examples. For datasets with outliers, PART, J48 and 1NN seemed to perform slightly better, but on a low-level. When examining the outcomes when introducing a sampling technique, it was found that NCR showed a superior result when dealing with borderline examples, while SMOTE and SPIDER did well on outliers. ROS generally was the lowest performing method, except when coupled with RBF. Overall, the sampling techniques improved the recognition of the minority class by 10 to 30%.

Each of these studies made use of a variety of different parameters for experimentation, these

being sampling methods, learning techniques, evaluation metrics, and data characteristics. A summary of these per study mentioned above is given in Table 2.2, where available.

New approaches to sampling have been researched by various authors and are growing rapidly. Due to relevance of these in this dissertation, it will not be discussed in detail.

2.6.2 Application domains affected by imbalanced data

In many real-world applications, the issue of imbalanced data classification becomes a significant concern, which lowers the model's ability to predict accurately. The most common focus area in the field of machine learning and data mining is rare event detection. This refers to events which occur less frequently, however could have a significant impact and lead to adverse effects on society. Rare events occur in a variety of application domains such as financial management, biomedical engineering and information technology (Haixiang et al., 2017). In order to build suitable detection systems for these applications, imbalanced learning strategies would need to be implemented, since these data are typically imbalanced.

In the medical field, the most frequent topic researched involved rare disease diagnosis. The data generated in these domains are typically unbalanced. In certain instances, image data are also used to detect anomalies in biomedical events. One such study by Bae and Yoon (2015) applied learning techniques to locate and estimate the size of polyps in images generated from endoscopy and colonoscopy procedures. In most recent times, the SARS-CoV-2 (commonly referred to as Coronavirus) pandemic has accelerated the need for predictive models to aid in the fight against the virus. Dorn et al. (2021) evaluated complete blood count (CBC) tests to determine if a patient is COVID-19 positive or negative. These medical applications highlighted possible solutions to early detection of rare diseases. This will assist in recommending courses of treatment timeously to avoid any fatalities.

Another field which contains a large amount of imbalanced data is Bioinformatics. These data contain biological information, with majority of the focus placed on genetics and genomics. Studies in this field primarily involve protein detection and gene expression detection. An interesting application was researched by Gao and Siu (2020) which investigated peptide-binding protein prediction. Protein-peptide interaction is one of the basic processes in cells, which facilitates a variety of cellular functions. Therefore, the prediction of these binding residues are essential and have been of interest in the machine learning field in the recent years.

In the information technology (IT) sphere, software defect detection, network intrusion detection and other anomaly detection techniques are typically used in situations when there is imbalanced data present. Due to fast expansion in the IT domain, it is imperative to find abnormal events from information devices and platforms and is essential for company decision-making and strategy development (Haixiang et al., 2017). One such study by Vorobeva (2016) aimed to enhance web-user identification and authentication process in order to improve the level of web security.

Fraud detection is another field which has a large proportion of imbalanced data. This consists of various types such as credit card fraud, insurance fraud, financial statement fraud

Reference	Sampling Technique	Learning Technique	Evaluation Metrics	Data Characteristics	IR
(Van Hulse et al., 2007)	RUS, ROS, One-Sided Selection (OSS), Cluster-Based Oversampling (CBOS), Wilson's Editing (WE), SMOTE and Borderline-SMOTE (BSM)	C4.5D and C4.5N decision tree learners, k-Nearest Neighbours (kNN) classifiers, Naïve Bayes (NB) classifier, Multilayer perceptrons (MLP), Radial basis function networks (RBF), Repeated Incremental Pruning to Produce Error Reduction (RIPPER), RF, SVM and LR	AUC, Kolmogorov-Smirnov statistic (K/S), geometric mean (G-mean), F-score, accuracy and true positive rate (TPR)	35 datasets, of which a majority originate from the UCI repository. The datasets contain both continuous and categorical features and range in size from 214 to 20 000 examples.	Ranging from 1.3% to 35%
(Batista et al., 2012)	ROS and SMOTE	C4.5 decision tree, C4.5 Rules (rule-based decision trees), CN2 and RIPPER, Back-propagation Neural Network, Naïve Bayes and SVMs	AUC	20 datasets from sources such as the UCI repository or data from projects on Statlog	Range from balanced datasets of 50% to highly imbalanced datasets of 1%
(Batista et al., 2004)	ROS, RUS, TL, Condensed Nearest Neighbour Rule (CNN), OSS, CNN + TL, Neighbourhood Cleaning Rule (NCR), SMOTE, SMOTE + TL, SMOTE + ENN	C4.5 decision tree	AUC	13 datasets from UCI repository with both continuous and categorical features. The data size ranges from 90 to 20 000.	Ranged from 2.6% to 34.7%
(Napierala and Stefanowski, 2016)	ROS, SMOTE and SPIDER	C4.5 decision tree, rule induction with PART and RIPPER algorithms, kNN, RBF and SVM	Sensitivity, G-mean and F-score	26 datasets were used in the study, where a majority comes from the UCI repository and some from medical sources., using previous studies in the field as a basis for investigation. These datasets contain both continuous and categorical features.	Ranging from 2.5% to 35.9%

TABLE 2.2: Summary of literature studies

and cyptocurrency fraud to name a few. These were briefly touched on in Chapter 1. Credit card fraud data are used as the focus of this dissertation, however, application of a few of the other types include a study by Kirlidog and Asuk (2012) and Aziz et al. (2022), which focused on health insurance fraud and Ethereum fraud respectively. In the health care insurance industry, predictive models are being extensively researched to detect possible fraudulent claims. Similarly, in the Ethereum fraud study, a variety of models were investigated to predict fraudulent transactions. These studies have advanced the detection of fraud significantly, where previous methods would be time-consuming as well as costly.

There are many other application domains which have been explored previously. These domain categories include chemical engineering, energy management, environmental management, agriculture, and astronomy. These will not be discussed but it is interesting to note that imbalanced datasets are prevalent in a large variety of real-life situations.

2.7 Conclusion

The topic of learning with imbalanced data is one that consists of a variety of factors. This chapter provides an overview of these considerations focusing on the characteristics of the data, solutions to the imbalance problem, evaluation metrics which need to be taken into account, and a summary of previous work that has been conducted. This information was taken into account when formulating the experimental set-up used in this dissertation, which is presented in the following chapter.

Chapter 3

Data

3.1 Introduction

This chapter aims to give an overview of the data used in this study, broken down into 2 subsections. This includes the origins of the data, the characteristics of the data and a summary of important findings from exploratory data analysis. This was performed in order to understand the data patterns as well as to make informed decisions during the modelling stages of the process. These points have been highlighted in the below sections.

3.2 Source of data

The dataset used in this investigation was sourced from Kaggle, which contains credit card transactions made by cardholders in Europe over a two day period in September 2013 (Machine Learning Group, 2018). The dataset contains 284 807 transactions, where 492 are classified as fraudulent (represented by the label 1) and the remainder being non-fraudulent cases (shown as a 0 label).

Figure 3.1 depicts the distribution of the transactions between fraud and non-fraud cases. Here, it can be seen that the data presents as a highly imbalanced dataset, with a prevalence rate of minority class observations being 0.2%.

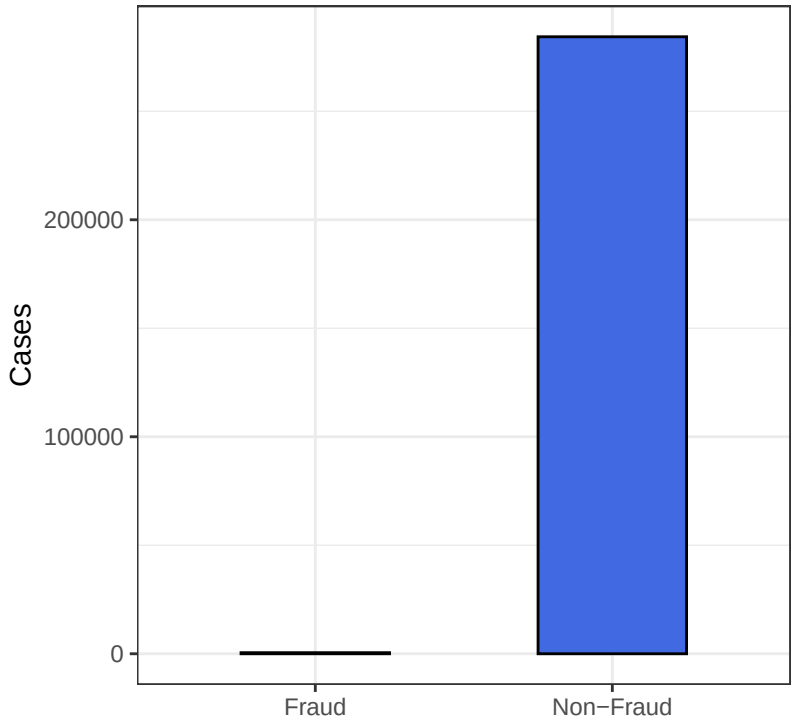

FIGURE 3.1: Distribution of fraud cases in dataset

Due to the sensitivity of the information within the dataset, PCA transformation was used. The resulting data contained only numerical values and did not provide much insight to complete further analysis. Table 3.1 below shows an overview of the data and the features within it. The only features which have not been transformed with PCA are Time, Amount and Class.

Feature	Description
V1, V2, ..., V28	PCA transformed features
Time	Contains the seconds elapsed between each transaction and the first transaction in the dataset
Amount	Amount relating to the transaction in dollars
Class	The response or target variable – 1 in case of fraud, 0 otherwise

TABLE 3.1: Description of features within dataset

PCA was used to transform the original features in the dataset and does not consider the semantic information about the features. Therefore, analysis and interpretation of these components and their corresponding numerical values could not be performed. Focus will be placed mainly on the features which have not been transformed, these being Time and Amount.

3.3 Data characteristics

Table 3.2 shows the features and a summary of the statistics related to each. From this, it can be seen that the Amount feature was measured on a different scale to that of the other features. Therefore, the Amount data were standardised.

Variable	Min	q_1	$\tilde{x}$	$\bar{x}$	q_3	Max	s	IQR	#NA
Time	0.0	54201.5	84692.0	94813.9	139320.5	172792.0	47488.1	85119.0	0
V1	-56.4	-0.9	0.0	0.0	1.3	2.5	2.0	2.2	0
V2	-72.7	-0.6	0.1	0.0	0.8	22.1	1.7	1.4	0
V3	-48.3	-0.9	0.2	0.0	1.0	9.4	1.5	1.9	0
V4	-5.7	-0.8	0.0	0.0	0.7	16.9	1.4	1.6	0
V5	-113.7	-0.7	-0.1	0.0	0.6	34.8	1.4	1.3	0
V6	-26.2	-0.8	-0.3	0.0	0.4	73.3	1.3	1.2	0
V7	-43.6	-0.6	0.0	0.0	0.6	120.6	1.2	1.1	0
V8	-73.2	-0.2	0.0	0.0	0.3	20.0	1.2	0.5	0
V9	-13.4	-0.6	-0.1	0.0	0.6	15.6	1.1	1.2	0
V10	-24.6	-0.5	-0.1	0.0	0.5	23.7	1.1	1.0	0
V11	-4.8	-0.8	0.0	0.0	0.7	12.0	1.0	1.5	0
V12	-18.7	-0.4	0.1	0.0	0.6	7.8	1.0	1.0	0
V13	-5.8	-0.6	0.0	0.0	0.7	7.1	1.0	1.3	0
V14	-19.2	-0.4	0.1	0.0	0.5	10.5	1.0	0.9	0
V15	-4.5	-0.6	0.0	0.0	0.6	8.9	0.9	1.2	0
V16	-14.1	-0.5	0.1	0.0	0.5	17.3	0.9	1.0	0
V17	-25.2	-0.5	-0.1	0.0	0.4	9.3	0.8	0.9	0
V18	-9.5	-0.5	0.0	0.0	0.5	5.0	0.8	1.0	0
V19	-7.2	-0.5	0.0	0.0	0.5	5.6	0.8	0.9	0
V20	-54.5	-0.2	-0.1	0.0	0.1	39.4	0.8	0.3	0
V21	-34.8	-0.2	0.0	0.0	0.2	27.2	0.7	0.4	0
V22	-10.9	-0.5	0.0	0.0	0.5	10.5	0.7	1.1	0
V23	-44.8	-0.2	0.0	0.0	0.1	22.5	0.6	0.3	0
V24	-2.8	-0.4	0.0	0.0	0.4	4.6	0.6	0.8	0
V25	-10.3	-0.3	0.0	0.0	0.4	7.5	0.5	0.7	0
V26	-2.6	-0.3	-0.1	0.0	0.2	3.5	0.5	0.6	0
V27	-22.6	-0.1	0.0	0.0	0.1	31.6	0.4	0.2	0
V28	-15.4	-0.1	0.0	0.0	0.1	33.8	0.3	0.1	0
Amount	0.0	5.6	22.0	88.3	77.2	25691.2	250.1	71.6	0
Class	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0

TABLE 3.2: Summary statistics of credit card fraud dataset

The distribution of the features can be plotted to understand the skewness. It would be interesting to understand how these features differ between fraudulent cases and non-fraudulent cases to determine which features affect the overall outcome of the transaction. Figure 3.2 shows the distribution plots for the features which showed a high

degree of skewness in the dataset.

V4, V11, V12, V14, V16 and V18 had different distributions and could be important variables to predict whether a transaction is fraudulent or not. From preliminary investigation, it was seen that the Time variable did not offer significant value to the study and was removed. This feature relates to the time elapsed from the first transaction recorded, however a time stamp was not provided to relate the transactions to a specific time of day, lessening the usefulness of the data. The Amount feature was highly skewed to the right. Here, there could be many outliers affecting the distribution.

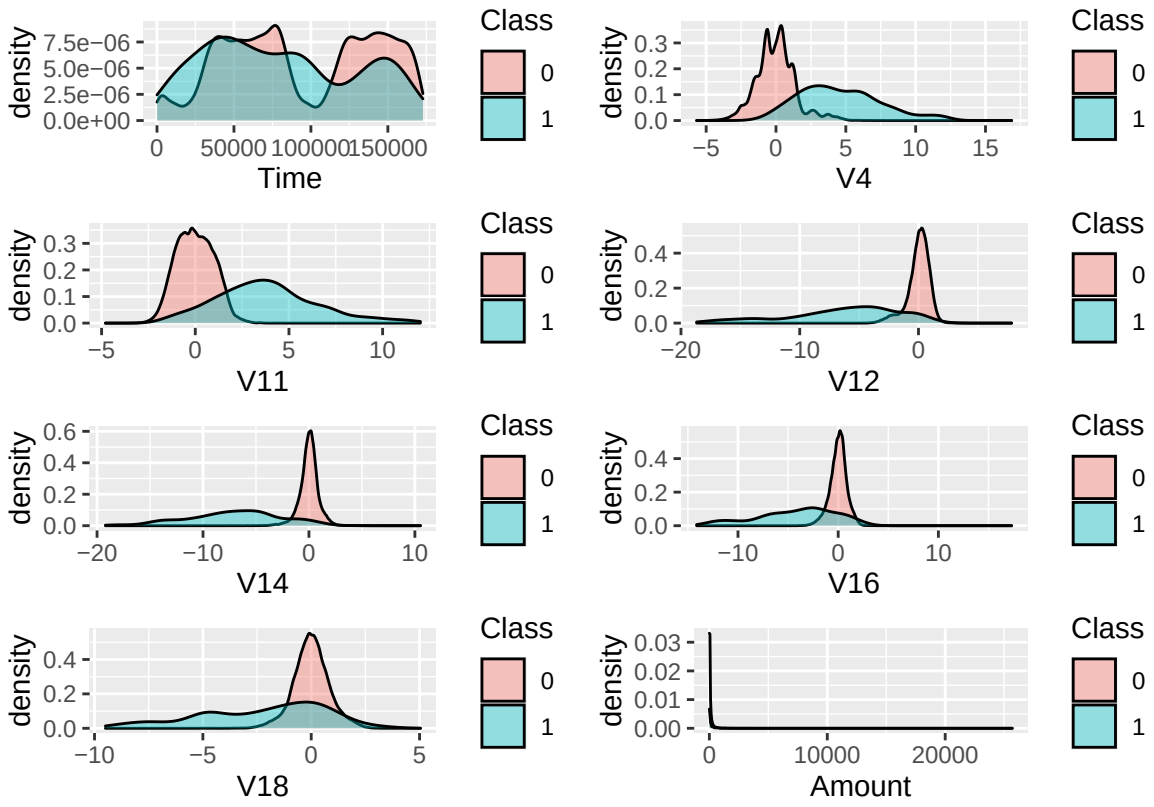

FIGURE 3.2: Distribution plots for selected features

The Amount feature was focused on to understand a bit more of the detail. There were numerous outliers detected, as seen in Figure 3.3. The visual representation shows the skewness of the data, particularly in the non-fraudulent cases. A decision was made to keep the outliers in the dataset due to the limited amount of data within the minority class. Removal of the outliers could results in elimination of useful data. It was also be seen that there are differences in the mean values between fraudulent and non-fraudulent classes (from Table 3.3), where the fraudulent class has a higher average amount. This could prove to be useful in model building.

Class	Min	Mean	Max
0	0.00	88.29	25691.16
1	0.00	122.21	2125.87

TABLE 3.3: Statistics for Amount feature per class before standardisation

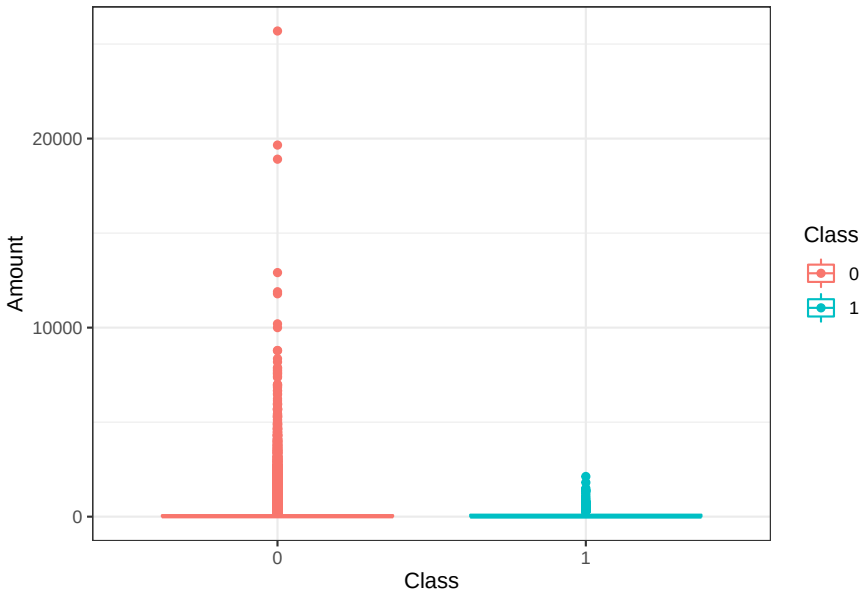

FIGURE 3.3: Boxplot showing outliers for the Amount feature

3.4 Conclusion

This chapter presents information around the data used for this study. This was broken down into the source of the data, its characteristics and an exploratory data analysis on the features. It serves as a prelude into Chapter 4, which introduces the methodolgy used to complete the investigation.

Chapter 4

Methodology

4.1 Introduction

This chapter gives the details of the methodology used in this study, including the experimental set-up used to meet the objectives of the investigation. The sampling methods and supervised learning techniques implemented in this study are dicussed in detail. Additionally, a summary of the experiments conducted for the study is provided. Each of these topics were divided into sub-sections, which can be seen below in the body of the chapter.

4.2 Imbalance techniques

There are a variety of sampling techniques which have been used to overcome the imbalance problem. In this study, the choice of technique to implement was driven by previous literature. Batista et al. (2004) and Batista et al. (2012) showed that ROS outperformed the SMOTE method for various classifiers. In addition, it was computationally less expensive than SMOTE. Japkowicz and Stephen (2002) also found that from RUS, ROS and cost modification methods, random oversampling performed the best. Similarly, Mohammed et al. (2020) showed that oversampling performed better than undersamping when using various supervised learning models. From this, a decision was made to use ROS, SMOTE and RUS as the sampling techniques of choice.

4.2.1 Random sampling

Following from Section 2.4.1.2, this subsection will give the detail on how ROS and RUS techniques are implemented during data pre-processing. These concepts were described by He and Garcia (2009) and are used as a basis of explanation of the mechanism behind sampling.

Random oversampling, as the name suggests, occurs when the observations from the minority class are randomly selected and duplicated. These duplicates are then introduced into the original dataset to reduce imbalance. While ROS adds minority class observations to the original dataset, random undersampling aims to reduce imbalance by removing

observations from the dataset. A randomly selected set of majority class observations are removed from the dataset until the desired imbalance ratio (IR) is met.

These techniques are fairly simple to conceptualise and application of this during the pre-processing stages is straightforward, especially in programming languages such as R. Details on a R related package for use is in the paragraph below.

Implementing random sampling techniques in R Addressing the imbalance learning problem in R can be completed through a number of packages. One notable packages developed by Kuhn (2014) is the `caret` package, which introduces methods for random sampling. It encompasses functions for regression and classification learning, with specific focus on the imbalance problem using simple sampling techniques. The more sophisticated `ROSE (Random Oversampling Examples)` package was developed by Lunardon et al. (2014), which incorporates tools to deal with binary classification using imbalanced data. All stages of the learning process, from model estimation to evaluation of the classification accuracy, are covered by the functions provided in the package. When implementing this in R, the user generally selects a solution to the class imbalance as well as a classifier in order to estimate the learning process. To balance the sample in the first step, the techniques of (''`over`'', ''`under`'', ''`both`'') can be used.

For this package, oversampling with replacement is completed, with respect to the minority class . This sampling method is driven by two possible conditions: 1) either a specified sample size N is met, or 2) when the probability p of minority class observations have reached the set value.

The undersampling method is conducted without replacement using the majority class as a basis. Once again, the two criteria of N and p are used as driving forces to determine when the undersampling has been completed.

Lastly, there is a hybrid option involved oversampling of the minority class with replacement as well as undersampling of the majority class without replacement. This will not be considered in this dissertation.

4.2.2 SMOTE

The SMOTE method generates synthetic data through the use of interpolation between minority class observations lying in close proximity to each other in a feature space. Deep diving into the intricacies of the algorithm, it is seen that there are several points to consider to create the new data points. Considering the minority class observations, one instance within the minority sample space is selected at random. This data point is assigned the variable of x_i. A k value is chosen to represent the number of k-Nearest Neighbours to x_i ($k = 5$ is the default). The k value is used to pinpoint the minority class observations which have the smallest Euclidean distance between itself and x_i in the feature space. In order to create a synthetic data point, the following equation is used:

$$x_{new} = x_i + (\hat{x}_i - x_i) \times R \tag{4.1}$$

where $\hat{x}_i$ is one of the randomly selected k-nearest neighbour points identified above and R is a randomly generated number between [0, 1].

Using the Equation 4.1, the resulting synthesised data point will lie between x_i and its k-nearest neighbour, $\hat{x}_i$.

Figure 4.1 depicts the SMOTE procedure using $k = 4$. Figure 4.1a shows an imbalanced dataset, with the minority class represented by red stars. A minority class example, x_i is randomly chosen as a starting point for the algorithm. A k value is selected at 4, therefore the 4 closest minority class observations are identified based on Euclidean distance. One of these neighbours are chosen at random to complete the sampling. Figure 4.1b shows the generated sample x_{new}, which lies in between x_i and the k-Nearest Neighbour, $\hat{x}_i$.

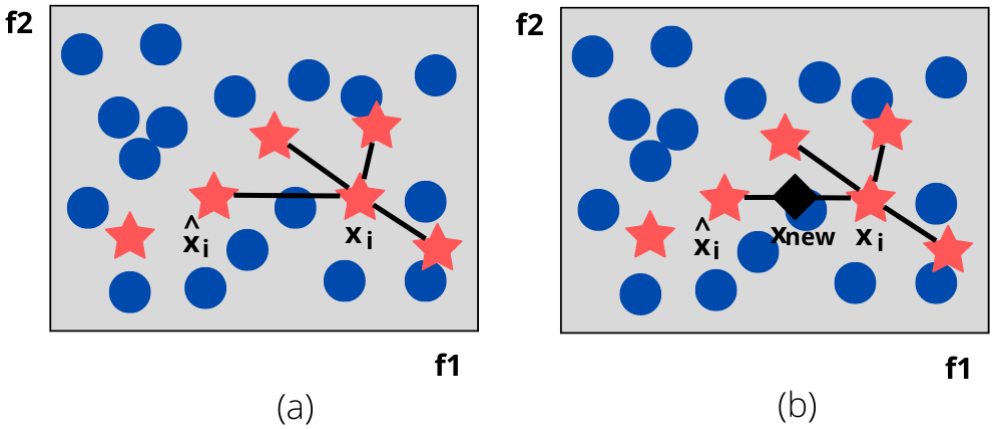

FIGURE 4.1: a) Image showing a randomly selected minority example x_i with its k-nearest neighbours identified ($k = 4$); b) Image showing a generated sythentic example (adapted from He and Garcia (2009))

Implementing SMOTE technique in R The simplest method to implement SMOTE is through the package `smotefamily` (Siriseriwan, 2021). This is used to synthesise new data points in relation to existing minority class instances and its k-Nearest Neighbours. It takes in two parameters to complete the sampling: `k` and `dup_size`.

`k` is chosen based on the number of k-Nearest Neighbours that would be used during the sampling process. `dup_size` represents the number of times the minority class data points are used for generating new observations. If `dup_size = 1`, one data point is synthesised for every existing minority class point. When `dup_size` is set to 0, the function will create a balanced dataset.

4.3 Supervised learning models

Five different supervised learning methods were chosen for investigation based on previous literature on this specific dataset. Several authors conducted studies focusing on fraud

detection using the credit card data and made use of the following learning techniques in Table 4.1 and was used in this dissertation:

Supervised Learning Method	Literature Source
Logistic Regression (LR)	(Lakshmi and Kavilla, 2018; Bhattacharyya et al., 2011; Sahin and Duman, 2011)
C4.5 Decision Tree (DT)	(Lakshmi and Kavilla, 2018; Husejinovic, 2020; Mijwil and Salem, 2020)
Random Forest (RF)	(Lakshmi and Kavilla, 2018; Bhattacharyya et al., 2011; Jemima Jebaseeli et al., 2021)
Extreme Gradient Boosting (XGBoost)	(Meng et al., 2020; Priscilla and Prabha, 2020; Abdulghani et al., 2021; Mohbey et al., 2022)
Neural Network (NN)	(Ghosh and Reilly, 1994; Patidar et al., 2011; Sahin and Duman, 2011; Georgieva et al., 2019)

TABLE 4.1: Summary of models used and literature sources

Each of these methods will be discussed briefly in the subsequent sections to understand the theory behind the algorithms.

4.3.1 Logistic regression

4.3.1.1 Mechanism behind model

Logistic regression (LR) is one of the most used supervised learning techniques and works in a similar manner to linear regression. While linear regression is used for solving problems where the output is a continuous variable, logistic regression is used to predict a binary output when given a set of explanatory variables.

Mathematically, logistic regression relies on the concept of probability to drive the classifications. The logistic regression equation (Equation 4.2) models the likelihood of an outcome based on specific characteristics (Sperandei, 2014).

$$\log\left(\frac{\pi}{1-\pi}\right) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_i x_i \tag{4.2}$$

where π is the probability of an event,
β_i are the regression coefficients, and
x_i are the explanatory variables

The cost function associated to this learning model is defined as the *sigmoid function*, otherwise known as the logistic function. Figure 4.2 below shows an example of the sigmoid function. The function is used to transform the predicted values into probabilities. As seen from the graph, the limits of the function are between 0 and 1. When the data are fed into the function, a probability score will be returned lying in the range of 0 and 1. A further step will be required to map these values into the desired classes. This is where the threshold value comes into play. This value is used as a cut-off point to determine the classification into the various classes. A general rule of thumb is to set the threshold at 0.5, which means any probability value less than

0.5 will be classified as Class 1 and a value of greater than 0.5 will be placed into Class 2, for binary classification problems.

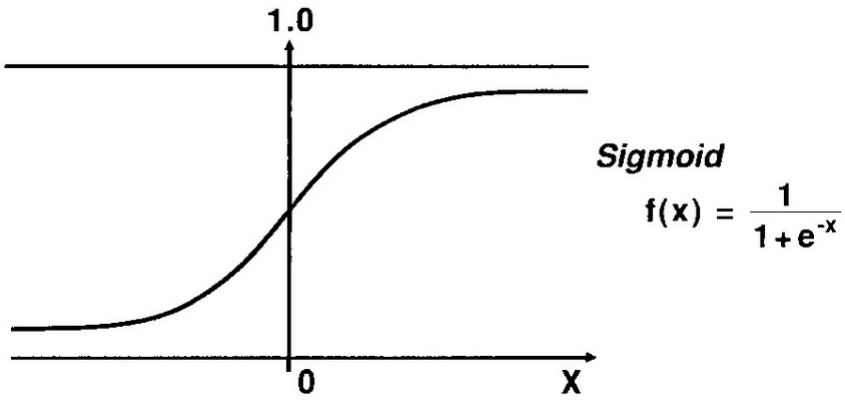

FIGURE 4.2: Depiction of the sigmoid function (from Næs et al. (1993))

4.3.1.2 Lasso regularisation

Regularisation is a technique used to minimise overfitting and improve model accuracy. Lasso regularisation was investigated in this study to determine its effect when using imbalanced data. The lasso model outcome is highly reliant on the lambda parameter, which is the tuning parameter in the algorithm as seen below in Equation 4.3:

$$\sum_{i=1}^{n}(y_i - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \tag{4.3}$$

The aim of the regularisation is to minimise the sum of squares, including the β coefficients. λ essentially controls the shrinkage amount of β coefficients. In the case where λ tends to zero, no coefficients will be eliminated. In the opposite manner, if λ value is very high, the coefficients will tend to zero and are removed from the model. In the case where lambda is infinity, all coefficients are removed. Therefore, the lambda value chosen plays a significant role in the outcome of the modelling.

4.3.2 C4.5 decision tree

One of the most popular classification techniques is the decision tree. The algorithm searches for the best attribute to divide the data by. Several splitting criteria exist aiming to reduce the impurity of a node (Singh and Gupta, 2014). The optimal approach to separate the records can be determined using a variety of splitting metrics.

The well-known ID3 developed by Quinlan (1986) chooses attributes based on a principle known as information gain. More recently, the C4.5 algorithm was created and uses the conncept of gain ratio as a way to split data in the nodes (Quinlan, 2014). The basics behind these concepts will be explained in further detail below to show the differences and advantages that the C4.5 has over other methods.

4.3.2.1 Splitting criteria

Information gain is a method used to determine which attribute to use to split the data in a node. It is known as an impurity criterion and is based on the entropy measurements at a specific node. The gain is calculated from Equation 4.4, defined as the change between the node's entropy prior to splitting (parent node) and the after splitting (child node) (Singh and Gupta, 2014). The attribute with the highest value is chosen.

$$Gain(A) = Entropy(D) - Entropy_A(D) \tag{4.4}$$

where D represents the parent node and
A represents the child node.

The entropy relates to the impurity of the node. Another way to look at this concept is the effort required to classify a record into a specified node (Equation 4.5).

$$Entropy(D) = -\sum_{i=1}^{m} p_i \log_2(p_i) \tag{4.5}$$

where m is the number of class labels and
p_i is the probability of an example in the specified node belonging to class i.

$Entropy_A(D)$ is the information required to categorise the data in a particular node once it has been divided using a specific attribute:

$$Entropy_A(D) = \sum_{j=1}^{n} p_j Entropy(D_j) \tag{4.6}$$

where n is the number of partitions,
p_j is the probability that an example is in partition j, and
D_j is resultant dataset which has been split into partition j.

While the information gain is widely used as a splitting criteria, the downfall lies in its bias towards attributes which have a wide range of values. The probability of predicting these values is low, thereby leading towards a high gain. This is where the C4.5 algorithm has advantages over the ID3 decision tree. A new concept of a normalised gain value was introduced (known as the Gain Ratio) which leads to an unbiased selection of which attribute to split on. The Gain Ratio is calculated using Equation 4.7:

$$GainRatio(A) = \frac{Gain(A)}{SplitInfo(A)} \tag{4.7}$$

and the $SplitInfo$ is the normalisation factor, given by Equation 4.8 below:

$$SplitInfo(A) = -\sum_{j=1}^{n} p_j \log_2(p_j) \tag{4.8}$$

4.3.2.2 Decision tree construction

Decision trees (DTs) are built through a recursive process, as seen in Figure 4.3 below. The training data are fed into the tree via the top node or root node. Here, the attribute which results in the highest gain ratio is chosen to split on. Once the first split is complete, the subsequent nodes formed will follow the same process of determining the attribute with the highest gain ratio using the data fed into it. Based on the class label of the majority of the records in each node, a majority label is assigned to each node. This process is carried out until one of the following stopping criteria is satisfied: 1) the node does not contain any records; 2) all of the data in the node are from the same class; or 3) none of the attributes result in additional information gain (Rim and Liu, 2020). The last node is referred to as a leaf and given a class label whenever a stopping criterion has been satisfied. The construction of a decision tree is completed once each record in the dataset has been assigned to a leaf.

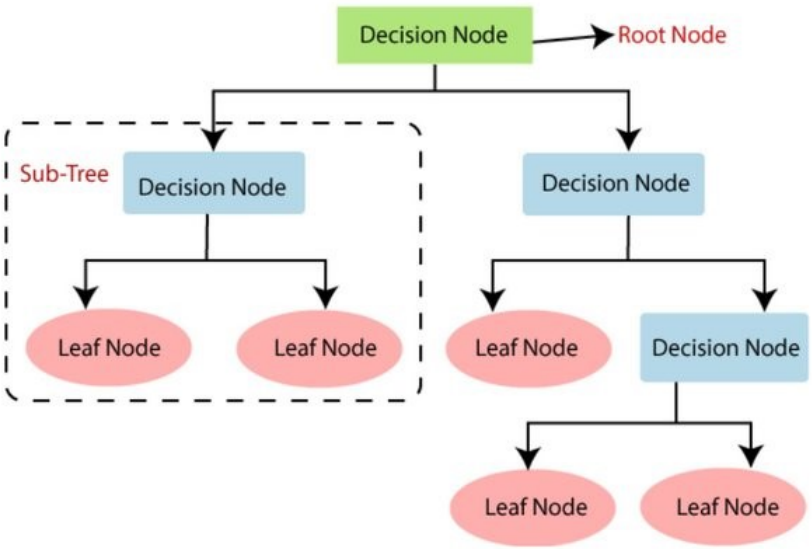

FIGURE 4.3: Depiction of decision tree (from Hanafy and Ming (2021))

4.3.3 Random forest

Random forest (RF) is a widely used supervised learning method for both classification and regression problems. The random forest is an ensemble method, which combines the results of various decision trees to achieve a single outcome by means of majority voting. Decision trees are used as the basis of this algorithm with the main differences being:

- Random forests make use of multiple, independent decision trees instead of a single tree.
- Random forests include feature randomness, which subsets the independent variables randomly while building the tree (Ayyadevara, 2018). This guarantees that the trees being built have a low correlation with each other. Unlike with decision trees, where all the features are incorporated into the model.

While decision trees (as discussed in Section 4.3.2) are prone to issues such as overfitting and bias, the ensemble learning method produces more accurate predictions, especially when each individual tree is built independent of each other.

4.3.3.1 Bagging

Bagging or *Bootstrap Aggregation* is an ensemble method introduced by Breiman (1996). The term bootstrap refers to the random sampling of the data in the training set. The sampling is done with replacement, such that data points can lie in more than one sample. Once multiple sample datasets have been created, they are fed into the models for training. Each tree will produce a prediction and depending on the problem, classification or regression, the majority or average of the results will be used as the final prediction. Figure 4.4 below shows the workflow of the random forest algorithm.

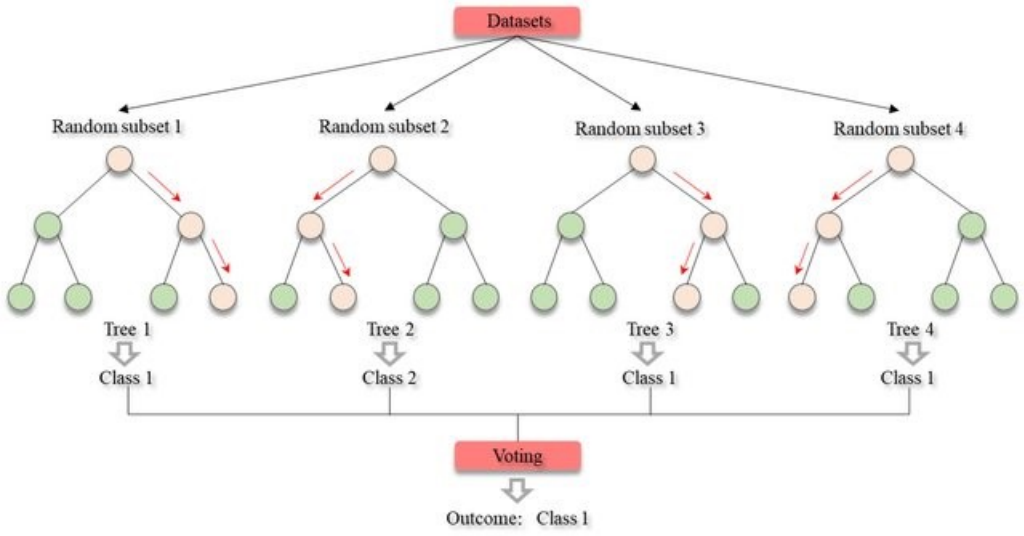

FIGURE 4.4: Depiction of the random forest model (from Yang et al. (2019))

4.3.4 XGBoost

eXtreme Gradient Boosting, commonly referred to as XGBoost, is a gradient boosting algorithm developed by Chen and Guestrin (2016), which has shown to surpass other supervised learning models in terms of performance in a variety of applications. In this ensemble method, decision trees are built sequentially in a manner that each succeeding tree minimises the errors obtained from the previous tree. The premise of the algorithm is based on the concept of boosting, which will be discussed in further detail below.

4.3.4.1 Gradient boosting

A *weak learner* is the building block in the boosting method. Brownlee (2016) defined a model as *weak learner* if it performs slightly better than random guessing. These weak learners provide crucial information in the predicting capabilities of the model. When combined, the learners produce a strong learner, resulting in reduced bias and variance. When compared to bagging techniques which develops trees to their full potential, boosting reduces the number of splits in the trees.

The gradient boosting process takes place over three basic steps, as stated by Ramraj et al. (2016); Gareth et al. (2013), and can be seen in Figure 4.5 below.:

1. A suitable loss function is defined based on the case study.
2. A weak learner is built to output predictions. For boosting, decision trees are the learners of choice.
3. An additive model is created which combines the outputs from the weak learners causing a reduction in the loss function. This can be represented as:

$$\hat{y}^{(b)} = \hat{y}^{(b-1)} + \lambda \hat{r}^{(b)} \tag{4.9}$$

and residuals are represented by:

$$r^{(b)} = r^{(b-1)} - \lambda \hat{r}^{(b)} \tag{4.10}$$

where b represents a number from $1, ..., B$
$\hat{y}^b$ is the predictions once b trees have been built,
$\hat{r}^b$ is the predictions of residuals of tree b,
λ is a shrinkage parameter used to weight the outcome of each tree, and
r^b is the residuals once b trees have been built.

This process occurs one tree at a time. The newly created tree will include the information gained from the previous sequence of trees to enhance the model's overall predictive capability. Once the loss function is optimised, the procedure is completed. The final predictions from the model are:

$$\hat{y} = y - r^{(B)} = \bar{y} + \lambda \sum_{b=1}^{B} \hat{r}^{(b)} \tag{4.11}$$

where $\bar{y} = \hat{y}^{(0)}$.

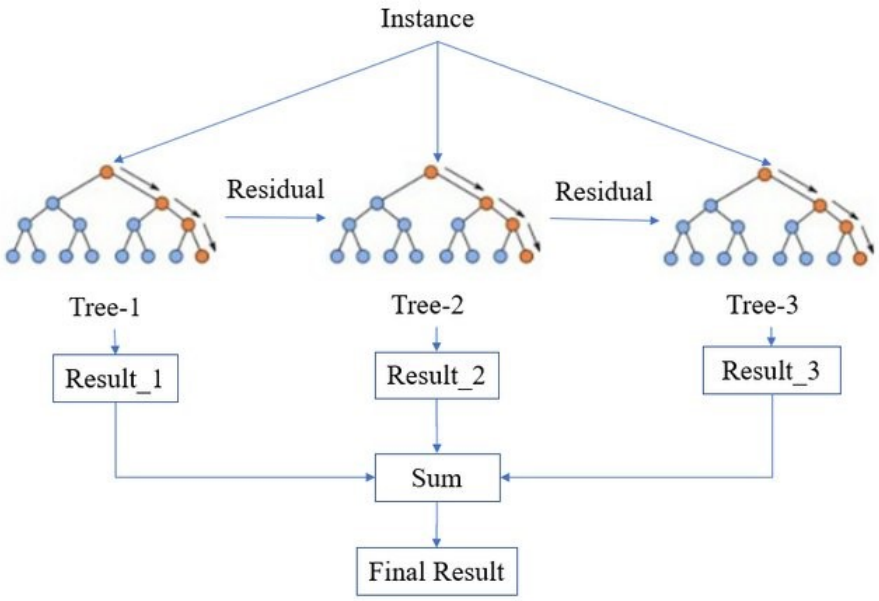

FIGURE 4.5: Depiction of the XGBoost model (from Wang et al. (2020))

The XGBoost model utilises the gradient boosting principle as the basis of the algorithm. The advantage that XGBoost offers over the standard boosting models comes in the addition of regularisation. This assists in controlling overfitting, thereby improving performance. In addition to this, the model is effective when using sparse datasets and reduces training time through parallelisation while building of the trees (Ramraj et al., 2016).

4.3.5 Neural network

Neural Networks (NN), also known as artificial neural networks (ANN), are the foundation of deep learning algorithms and are modelled around the functioning of the human nervous system. The basis of the model specifically focuses on the process where the neurons in the human brain send signals to each other to create a significant outcome. An input layer, one or more hidden layers, and an output layer make up the architecture of a neuron in a NN. A simple depiction of this can be seen in Figure 4.6. Each neuron is interconnected with others and has a threshold and weight that are specific to it. These factors control whether data are sent to the next layer of the network. The building blocks of a neuron are called perceptrons, which is a single neuron with one or more inputs and a single output.

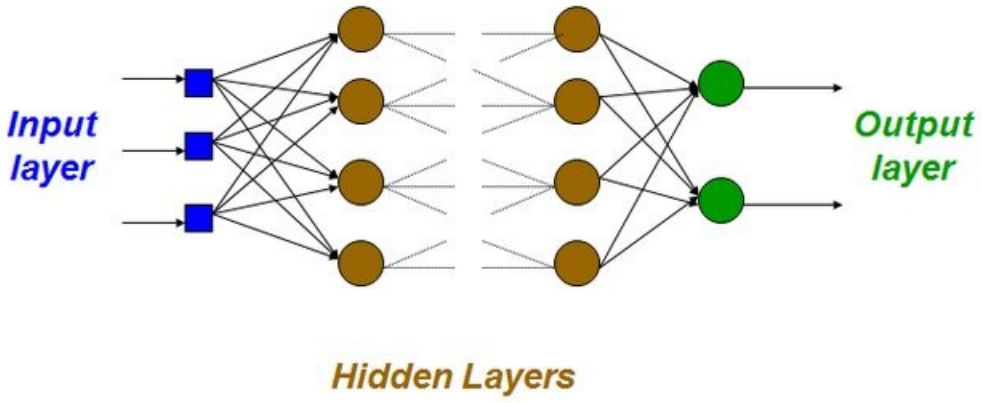

FIGURE 4.6: Depiction of a Simple Neural Network (from Nwankwo and Ihueze (2018))

4.3.5.1 Mechanism behind model

The mechanism behind the neurons in a network is broken down as follows:

- Input signals are received by the neuron originating from either a raw dataset or from the previous neurons in the network.
- The neuron completes a transformation of the input data.
- Output signals are passed onto the next layer in the network through a synapse.

Synapses play an important function in the model. These are used as connection points between the neuron and the previous layers. Each synapse has a weight that determines how important the preceding neuron is in the broader neural network (McCullum, 2020). The adjustment of these weights is the driving factor for training deep learning models and modified using a gradient descent algorithm. A basic depiction of the neuron is seen in Figure 4.7. The neuron receives an input, combines the signal and its associated weight and sends the information to an activation function. The activation function transforms the data to lie within its thresholds and, in turn, determines the output of the neuron. There are various activation functions which can be used, the commonly used ones being: Linear, Sigmoid/Logistic, Tanh and Rectified Linear Unit (ReLU). The output is sent through to the next layer through a synapse.

The modelling for an ANN consists of four high-level steps, as stated by McCullum (2020):

1. Inputs and initial weights are used to calculate predictions.
2. A cost function is used to determine the error of the neural network resulting from the predictions.
3. The weights are adjusted using a gradient descent algorithm.
4. The above steps are repeated until the weights remain constant or there is a minimal change in the values.

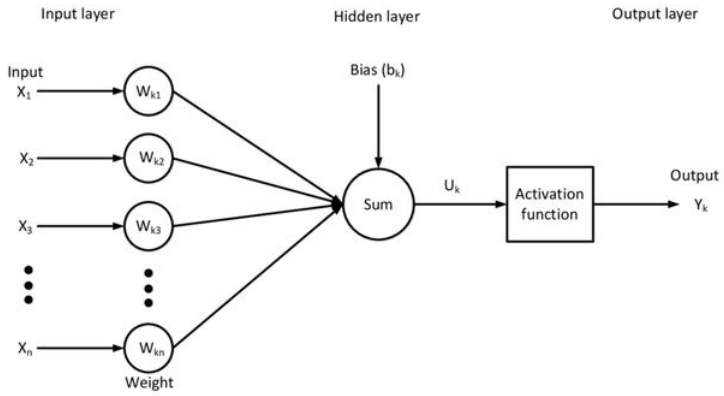

FIGURE 4.7: Depiction of a Neuron (from Joseph et al. (2019))

4.4 Experimental set-up

The above sections focused on the sampling methodology and supervised learning techniques used in this study. These two factors, together with the prevalence rates, formed the basis of the experimentation, which will be discussed in the section below.

4.4.1 Prevalence rates

Experiments conducted by Batista et al. (2012) made use of an extensive selection of imbalance ratios. To meet the objective of this study, the models would need to be built using a range of datasets from highly imbalanced to the point where the datasets reach a balanced state. Therefore, it was decided to confine the experimentation to the following prevalence rates as it provides good coverage of different imbalance percentages: original dataset at 0.2%, 1%, 10%, 20%, 30%, 40% and 50%. This would allow for a more accurate analysis of how different levels of imbalance affects the performance of the classifiers.

4.4.2 Model hyperparameters

One of the most important factors in model building is deciding on which hyperparameters to use. This choice affects the end performance of the model. In general, one should perform hyperparameter tuning, which selects the parameters which optimise the model performance. Due to the large number of models built in this study, hyperparameter tuning was not conducted. In addition, the main focus of this study was to understand the effect of imbalance on the performance of models, therefore the hyperparameters were not subjected to thorough investigation. Instead, these parameters were chosen based on past literature studies which have used this credit card dataset. In some models, default parameters were used, as seen in Table 4.2 below.

Model	Parameters	Reference
Logistic Regression	family = binomial Lasso regularisation	Default for Binary Classification
C4.5 Decision Tree	None	
Random Forest	num.trees = 500 importance = "impurity" max.depth = 50 min.node.size = 1	
XGBoost	nrounds = 4000 min_child_weight = 3 max_depth = 15 eta = 0.3 gamma = 0.1 colsample_bytree = 0.4 objective = "binary:logistic"	(Prabha and Priscilla, 2020)
Neural Network	Input layer = 29 neurons & ReLU 1 hidden layer = 13 neurons & ReLU Output layer = 2 neurons & Softmax Loss function = Binary cross-entropy Optimiser = Adam	(Georgieva et al., 2019)

TABLE 4.2: Optimal hyperparameters used in the study

4.4.3 Procedure

The experiments were designed based on the objectives set out on Section 1.3. The data pipeline in Figure 4.8 shows the protocol followed for completing the experiments set out for the investigation, with a total of 114 models being built. The detail around the set-up can be seen below:

- Pre-processing of the data was conducted. Three sampling techniques were chosen: Random oversampling, SMOTE and Random undersampling. Various degrees of imbalance were investigated. The data were rebalanced based on the prevalence rate stipulated in Section 4.4.1.
- Supervised learning techniques were applied to the rebalanced datasets. For this study, five supervised learning models were chosen: Logistic Regression (Vanilla and addition of Lasso Regularisation), C4.5 Decision Tree, Random Forest, XGBoost and Neural Networks.
- An additional investigation was completed on the optimal threshold values used on the Logistic Regression model.
- The models were evaluated using three metrics: Precision, Recall and F2 score. AUC was also investigated and results can be seen in Appendix B.

Below is a detailed diagram (Figure 4.8) outlining the steps performed in this study, summarised in the above research steps.

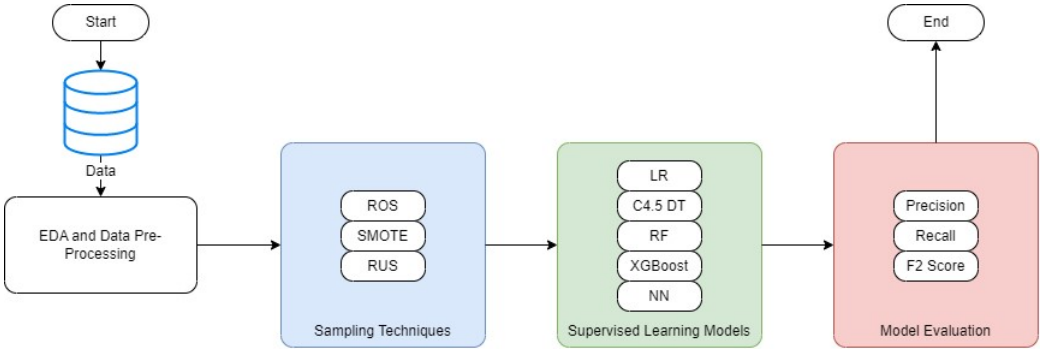

FIGURE 4.8: Pipeline followed during modelling

Prior to building these models, exploratory data analysis (EDA) was conducted to understand the dataset, as well as improve the model performance by transforming the data to a suitable structure. The main findings of the EDA can be found in Chapter 3.

Once EDA was completed, the data were split into a 80:20 training and testing portion using stratified sampling to ensure that there is an even distribution of minority class observations over the datasets. The test dataset was kept aside for predictions and performance analysis. Table 4.3 below shows a summary of the split datasets produced.

Dataset	Split	Prevalence Rate	Number of Features	Number of Samples
Train	80%	0.2%	29	227846
Test	20%	0.2%	29	56961

TABLE 4.3: Description of train and test datasets

The training set was put through a sampling exercise where the 3 sampling techinques were completed to create new datasets for each prevalence rate. Table 4.4 displays the split between majority and minority class observations for the 50% imbalance percentage per sampling technique. Upon analysis of the resampled datasets, it was noted that the number of minority and majority class observations produced do not match exactly. When sampling was implemented in R, the parameter used to complete the sampling was p, which represents the probability of the minority class observations in the resulting dataset. Therefore, the sampling algorithms created datasets with minority and majority class observations which meet the imbalance percentage required rather than a specific sample size per class. As seen in Table 4.4, the number of minority class and majority class observations were not the same, however it did meet the criteria of being a 50% imbalanced dataset. More detail around the datasets produced can be seen in Appendix A.

Imbalance %	Sampling Method	Number of minority observations in dataset	Number of majority observations in dataset
50%	ROS	227353	227452
	SMOTE	227338	227452
	RUS	394	385

TABLE 4.4: Summary of dataset generated for one prevalence rate

These datasets were fed into each supervised learning model, where the performance was evaluated and discussed in Chapter 5.

4.5 Conclusion

This chapter provides information on the methodolody used in this study. This includes a description of the three factors and the parameters used for each, which are prevalence rates, sampling techniques and supervised learning methods. It serves as a prelude to Chapter 5, which presents the results from the experiments and a discussion, providing an analysis to results obtained.

Chapter 5

Results and Discussion

5.1 Introduction

This chapter presents the results from the experiments conducted, as stipulated in Section 4.4. A brief overview of the metrics used to evaluate the models is given. In addition, a breakdown of the results from the six experiments conducted is provided, one section per supervised learning model. The results of each experiment is presented in tables with an interpretation. In the last section of the chapter, a comparison between the models is discussed with an analysis of the performance based on supervised learning technique, sampling method and prevalence rates.

5.2 Evaluation metrics

In order to determine which metrics to use in evaluating the performance of the models, the objectives of this study need to be taken into account. In this investigation, the model should minimise the number of false negatives predicted, which was the basis for selecting the evaluation metrics. Precision, recall and F2 score were used to compare model performances. The best models based on F2 score are bolded in the results tables included in the subsections of this chapter. AUC was investigated, which represents a model's accuracy based on its ability to identify the classes, where 1 is a perfect classification and 0.5 indicates a random choice of class. However, the AUC was not included in this chapter due to the insignificant changes in performance over prevalence rates, sampling techniques and supervised learning methods. Appendix B includes the results and a brief analysis from the AUC investigation.

5.3 Model results

5.3.1 Model 1: Logistic regression

5.3.1.1 Vanilla model

The LR models were built using default parameters, as seen in Section 4.4.2. Presentation of these findings can be found in Table 5.1, where the main trends can be highlighted for further interpretation and discussion. A high-level overview of the results showed that the precision

values decreased as the prevalence rate increased. The highest precision value was found to be 0.82 using the original dataset. On the other hand, recall resulted in a proportional relationship with the prevalence rate. The recall value peaked at 0.94 for several models, these being 30% RUS, 40% RUS, 50% SMOTE and 50% RUS models. Finally, highlighting the F2 score, one can observe that the values decreased as the prevalence rate increased. The best models based on the F2 score were the 1% ROS and 1% SMOTE models, with a value of 0.80. Comparing the F2 scores across sampling techniques, it can be seen that RUS resulted in models which performed the poorest overall. Conversely, ROS showed the best performances compared to SMOTE.

	Prevalence Rate	Sampling Technique	Precision	Recall	F2 Score
1	Original	NA	0.82	0.67	0.70
2	**1%**	**ROS**	**0.73**	**0.82**	**0.80**
3	**1%**	**SMOTE**	**0.73**	**0.82**	**0.80**
4	1%	RUS	0.71	0.82	0.79
5	10%	ROS	0.35	0.88	0.67
6	10%	SMOTE	0.31	0.87	0.64
7	10%	RUS	0.31	0.88	0.64
8	20%	ROS	0.17	0.90	0.49
9	20%	SMOTE	0.16	0.88	0.47
10	20%	RUS	0.19	0.89	0.51
11	30%	ROS	0.12	0.91	0.40
12	30%	SMOTE	0.12	0.90	0.38
13	30%	RUS	0.02	0.94	0.10
14	40%	ROS	0.09	0.92	0.32
15	40%	SMOTE	0.08	0.92	0.30
16	40%	RUS	0.07	0.94	0.26
17	50%	ROS	0.06	0.93	0.24
18	50%	SMOTE	0.06	0.94	0.24
19	50%	RUS	0.05	0.94	0.19

TABLE 5.1: Results from LR model based on performance on test dataset

5.3.1.2 Investigation on threshold-tuning

As an objective within the logistic regression modelling, fine tuning for the optimal threshold (p) value was investigated for imbalanced datasets. One of the methods to overcome the imbalance problem is adjusting the p value, which controls the classification as to whether a predicted probability will lie in the 0 class or 1 class. Generally, the threshold value is set at a default probability of 0.5, where a value above 0.5 is classified as 1 (fraud) and a value below 0.5 is classified as 0 (non-fraud). However, due to the highly imbalanced nature of the dataset used in this study, it was decided to perform an analysis on the effect of differing threshold values on the performance of the models.

For this investigation, the LR model, from Section 5.3.1.1 above, with the highest F2 value was chosen. In this instance, there were two models which resulted in the same metric values, these being the ROS and SMOTE techniques with a 1% prevalence rate. The ROS model was

chosen to complete the analysis.

The threshold value was increased in increments of 0.1 from 0.1 to 0.9 to determine the relationship between the performance as the threshold varies. A ROC curve (Figure 5.1) was plotted to visually depict the results obtained from this investigation. In this graph, the threshold values are represented by the colour scale from blue to red, representing 0 and 1 respectively. To obtain the optimal threshold value, one would identify the point on the graph whereby the FPR is low and the TPR is high or obtaining the point in the graph closest to the top left corner. Interpretation of the plot revealed that at a threshold value of 0.1, the highest TPR was obtained for the lowest FPR. Therefore, it was deduced that this threshold should yield the best performance.

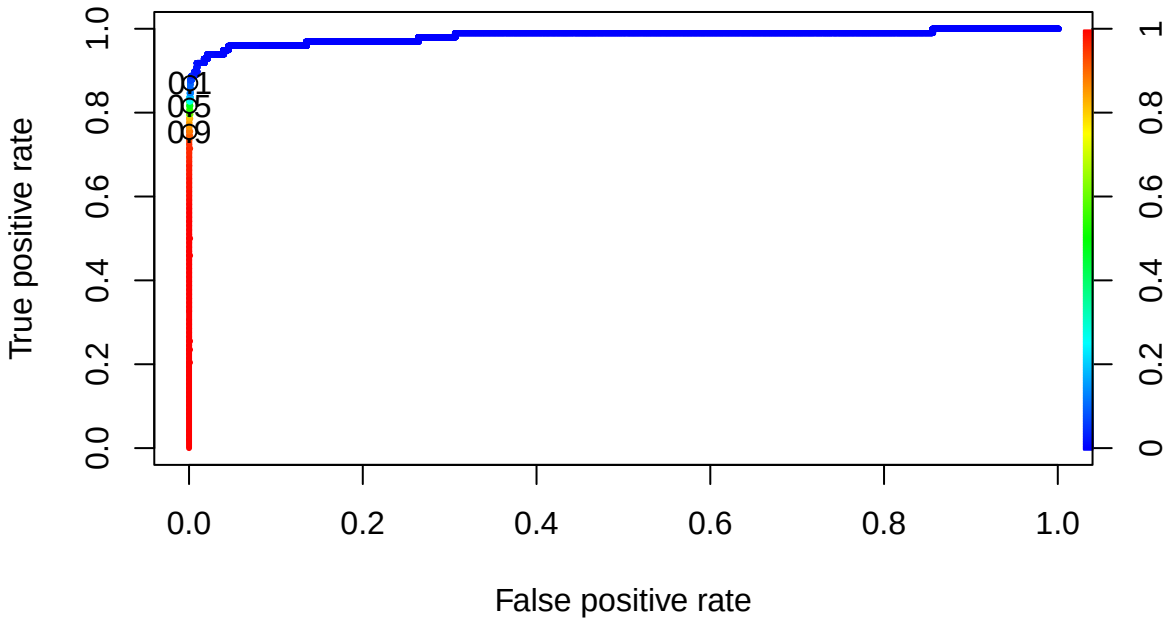

FIGURE 5.1: ROC curve for 1% Prevalence Rate using ROS showing various threshold values

Taking a closer look at the metrics at each increment (Table 5.2), the results corroborate when focusing on the recall values, where the highest value is observed at a threshold of 0.1. The F2 score showed a slight improvement when increasing the threshold value, but was not significant to draw a definite conclusion. As this specific case study places more importance on recall, a decision to adjust the threshold value to 0.1 could be made, which would provide the best outcome.

Threshold Value	Precision	Recall	F2 Score
0.1	0.55	0.87	0.78
0.2	0.65	0.84	0.79
0.3	0.70	0.83	0.80
0.4	0.70	0.83	0.80
0.5	0.73	0.82	0.80
0.6	0.75	0.80	0.79
0.7	0.77	0.80	0.79
0.8	0.78	0.80	0.79
0.9	0.83	0.74	0.76

TABLE 5.2: Evaluation metrics at different threshold values using test set

For the purposes of the rest of the dissertation, the threshold value was kept constant at 0.5. However, it is interesting to note the effect of imbalanced datasets on optimum threshold values.

5.3.1.3 Lasso model

Introducing lasso regularisation into the study was done to reduce overfitting and improve model performance. These models could then be used in an analysis to compare the effect of regularisation to the vanilla LR models explored in Section 5.3.1.1. The performance metrics for the lasso models can be seen in Table 5.3. Delving into the results obtained, an interesting discovery was made for the models using the highly imbalanced datasets, these being the original dataset and the 1% prevalence rate datasets. It was found that these datasets did not converge and provided metrics results which were not interpretable.

Evaluating the results obtained from the model using the highly imbalanced datasets (original to 1%), it was interesting to note that all the coefficients tend to zero and are eliminated from the model, thereby producing uninterpretable values for the performance metrics. Cross-validation was used to determine λ, where $\lambda_m in$ was chosen for further modelling and evaluation. Precision and F2 score result in NAs and recall was found to be zero. As the prevalence rate increased, the performance metrics became more understandable. It can be seen that there is an inversely proportional relationship between the prevalence rate and the precision and F2 score metrics. As the rate increased, the performance decreased. The 10% prevalence rate results were the better performing models, and 50% prevalence rate datasets performed the worst. Recall showed the opposite trend, where the 50% datasets performed better than the 10%. Based on the F2 score, the best performing model was the 10% prevalence rate with the RUS dataset. It can also be seen that generally, across the models, the RUS sampling technique resulted in the best performance when compared to its counterparts of ROS and SMOTE.

When comparing the F2 scores between the lasso model and the vanilla LR model (see Table 5.3), the lasso model performed slightly better for most model combinations, barring the original and 1% datasets. The ROS and SMOTE models performed similarly, while the RUS models showed an improvement in F2 score. This result can be expected as Lasso regularisation is known to perform well on sparse datasets with a large number of irrelevant

features. Undersampling techniques tend to produce sparse datasets through the elimination of observations. Therefore, Lasso regularisation could be more effective in undersampled datasets compared to oversampled datasets. Through random oversampling, the duplication of data points could affect feature selection and lead to overfitting. In addition, this could increase the density of the dataset, making the identification of important features more difficult.

	Prevalence Rate	Sampling Technique	Precision	Recall	F2 Score	Unregularised F2 Score
1	Original	NA	NA	0.00	NA	0.70
2	1%	ROS	NA	0.00	NA	0.80
3	1%	SMOTE	NA	0.00	NA	0.80
4	1%	RUS	NA	0.00	NA	0.79
5	10%	ROS	0.35	0.88	0.67	0.67
6	10%	SMOTE	0.31	0.88	0.64	0.64
7	**10%**	**RUS**	**0.46**	**0.86**	**0.73**	**0.64**
8	20%	ROS	0.17	0.90	0.49	0.49
9	20%	SMOTE	0.16	0.88	0.47	0.47
10	20%	RUS	0.19	0.89	0.52	0.51
11	30%	ROS	0.12	0.91	0.40	0.40
12	30%	SMOTE	0.12	0.90	0.39	0.38
13	30%	RUS	0.11	0.91	0.37	0.10
14	40%	ROS	0.09	0.92	0.32	0.32
15	40%	SMOTE	0.08	0.92	0.31	0.30
16	40%	RUS	0.09	0.92	0.33	0.26
17	50%	ROS	0.06	0.93	0.25	0.24
18	50%	SMOTE	0.06	0.93	0.23	0.24
19	50%	RUS	0.07	0.92	0.26	0.19

TABLE 5.3: Results from LR model with lasso regularisation based on performance on test dataset

5.3.2 Model 2: C4.5 decision tree

The C4.5 DTs were built using the default parameters in R. The results from this experiment were tabulated and presented in Table 5.4 below. As with the LR and lasso models, similar trends were picked up for all the performance metrics. Here, the best performing model used the original dataset, with a F2 score of 0.81. The precision was seen to decrease as the prevalence rate increased. Once again, the RUS models resulted in poor precision performance, the worst being 0.02 for the 50% dataset. ROS showed the best precision values in comparison to the other sampling techniques. The recall metric showed the opposite pattern, where the values increased with prevalence rate. Comparison between the sampling techniques showed that ROS was the worst performing and RUS the best performing models in terms of recall. The overall performance was shown by the F2 score, which showed a slight decrease as the prevalence rate increased. In general, one could conclude that the ROS models performed the best using the F2 score when compared to SMOTE and RUS.

	Prevalence Rate	Sampling Technique	Precision	Recall	F2 Score
1	**Original**	**NA**	**0.86**	**0.80**	**0.81**
2	1%	ROS	0.70	0.81	0.78
3	1%	SMOTE	0.58	0.81	0.75
4	1%	RUS	0.66	0.85	0.80
5	10%	ROS	0.69	0.81	0.78
6	10%	SMOTE	0.43	0.86	0.71
7	10%	RUS	0.31	0.86	0.64
8	20%	ROS	0.58	0.82	0.75
9	20%	SMOTE	0.41	0.83	0.69
10	20%	RUS	0.06	0.88	0.22
11	30%	ROS	0.59	0.82	0.76
12	30%	SMOTE	0.36	0.84	0.66
13	30%	RUS	0.04	0.91	0.16
14	40%	ROS	0.52	0.84	0.75
15	40%	SMOTE	0.41	0.83	0.69
16	40%	RUS	0.04	0.90	0.16
17	50%	ROS	0.52	0.83	0.74
18	50%	SMOTE	0.38	0.83	0.67
19	50%	RUS	0.02	0.92	0.09

TABLE 5.4: Results from C4.5 DT model based on performance on test dataset

5.3.3 Model 3: Random forest

The random forest models were built using the hyperparameters stipulated in Section 4.4.2. Using these parameters, the results shown in Table 5.5 were obtained. Upon evaluation of the metrics, it can be seen that the models perform fairly well on the various datasets. One can see that the recall remains fairly stable across prevalence rates and sampling techniques. One noticeable result is seen for the RUS models, where the models perform the best in terms of recall, resulting in a value of 0.91. In general, the precision values decreased as the prevalence rate increased. The exception to this was the ROS models, whereby the precision showed a slight improvement moving from highly imbalanced to balanced datasets. The overall performance was evaluated on the F2 score. Here, it can be seen that the F2 score was similar across models with sampling techniques of ROS and SMOTE. The RUS models had a poor performance when moving towards higher prevalence rates. The best model produced, in terms of F2 score, was the 30% ROS model and produced both high recall (0.84) and precision (0.94) rates.

	Prevalence Rate	Sampling Technique	Precision	Recall	F2 Score
1	Original	NA	0.92	0.82	0.84
2	1%	ROS	0.92	0.82	0.84
3	1%	SMOTE	0.90	0.83	0.84
4	1%	RUS	0.77	0.86	0.84
5	10%	ROS	0.93	0.84	0.85
6	10%	SMOTE	0.84	0.84	0.84
7	10%	RUS	0.50	0.87	0.76
8	20%	ROS	0.92	0.83	0.84
9	20%	SMOTE	0.82	0.84	0.83
10	20%	RUS	0.36	0.88	0.68
11	**30%**	**ROS**	**0.94**	**0.84**	**0.86**
12	30%	SMOTE	0.82	0.84	0.83
13	30%	RUS	0.14	0.89	0.43
14	40%	ROS	0.92	0.82	0.84
15	40%	SMOTE	0.80	0.84	0.83
16	40%	RUS	0.10	0.91	0.34
17	50%	ROS	0.95	0.82	0.84
18	50%	SMOTE	0.80	0.84	0.83
19	50%	RUS	0.05	0.91	0.22

TABLE 5.5: Results from RF model based on performance on test dataset

5.3.4 Model 4: XGBoost

As with the Random Forest, the XGBoost models were built using optimised hyperparameters found in the works of Prabha and Priscilla (2020). These parameters were incorporated into the models and the performance metrics were calculated, which are presented below in Table 5.6. At a quick glance, it is noticeable that the XGBoost classifier performed fairly well on the dataset, regardless of prevalence rate. Deeper analysis showed that the recall metric increased with prevalence rate and showed the best results as it moved towards the more balanced RUS models, with the maximum value at 0.93. On the other hand, precision decreased with prevalence rate, where RUS performed the worst, with a value of 0.04 for the 50% dataset. Focusing on the F2 score to understand the overall performance, one can observe that the ROS and SMOTE models performed similarly, with the 50% SMOTE model performing the best with a value of 0.85. The RUS models showed poor performance as the prevalence rate increased, which is mainly attributed to the fact the precision values were extremely low for these models.

	Prevalence Rate	Sampling Technique	Precision	Recall	F2 Score
1	Original	NA	0.95	0.82	0.84
2	1%	ROS	0.93	0.82	0.84
3	1%	SMOTE	0.92	0.83	0.84
4	1%	RUS	0.79	0.85	0.84
5	10%	ROS	0.90	0.82	0.83
6	10%	SMOTE	0.84	0.84	0.84
7	10%	RUS	0.34	0.88	0.66
8	20%	ROS	0.91	0.83	0.84
9	20%	SMOTE	0.85	0.85	0.85
10	20%	RUS	0.20	0.90	0.53
11	30%	ROS	0.91	0.82	0.83
12	30%	SMOTE	0.82	0.85	0.84
13	30%	RUS	0.11	0.93	0.37
14	**40%**	**ROS**	**0.90**	**0.84**	**0.85**
15	40%	SMOTE	0.82	0.85	0.84
16	40%	RUS	0.07	0.93	0.27
17	50%	ROS	0.90	0.83	0.84
18	50%	SMOTE	0.85	0.85	0.85
19	50%	RUS	0.04	0.93	0.16

TABLE 5.6: Results from XGBoost model based on performance on test dataset

5.3.5 Model 5: Neural network

There are a variety of options when it comes to the architecture of a NN model. The optimal parameters discovered by Georgieva et al. (2019) for this specific dataset is shown in Section 4.4.2. The outcome from implementing this architecture during the model building can be seen in Table 5.7. A high-level analysis of the results showed similar results to that found in Sections 5.3.2, 5.3.3 and 5.3.4 for the C4.5 DT, RF and XGBoost models. Here, the precision decreased as prevalence rate increased. The RUS models performed the worst using the precision as an evaluation metric, with the lowest value being 0.05 at 50% prevalence rate. Conversely, the SMOTE models showed the best precision when compared to the other sampling techniques. The recall values showed a slight increase as prevalence rate increased within the ROS and SMOTE models. The RUS models showed a significant increase and resulted in the highest recall value of 0.93 when it reached a balanced dataset. Comparing the models against the F2 score, it can be seen that the 1% SMOTE model performed the best with a value of 0.85.

	Prevalence Rate	Sampling Technique	Precision	Recall	F2 Score
1	Original	NA	0.86	0.77	0.78
2	1%	ROS	0.73	0.85	0.82
3	**1%**	**SMOTE**	**0.77**	**0.87**	**0.85**
4	1%	RUS	0.74	0.84	0.82
5	10%	ROS	0.62	0.87	0.80
6	10%	SMOTE	0.66	0.87	0.82
7	10%	RUS	0.41	0.86	0.70
8	20%	ROS	0.58	0.86	0.78
9	20%	SMOTE	0.58	0.86	0.78
10	20%	RUS	0.20	0.89	0.53
11	30%	ROS	0.47	0.85	0.73
12	30%	SMOTE	0.48	0.85	0.73
13	30%	RUS	0.13	0.91	0.42
14	40%	ROS	0.44	0.87	0.73
15	40%	SMOTE	0.53	0.86	0.76
16	40%	RUS	0.07	0.93	0.27
17	50%	ROS	0.53	0.87	0.77
18	50%	SMOTE	0.52	0.87	0.77
19	50%	RUS	0.05	0.93	0.20

TABLE 5.7: Results from NN model based on performance on test dataset

5.4 Discussion: Comparison of models

The results obtained for each supervised learning technique were analysed to determine which models produce the best results on this credit card dataset. The F2 score was used as the basis for this investgation. A visual depiction of the data can be seen in Figure 5.2 for interpretation. In the graph, conclusions between supervised learning models, sampling techniques and prevalence rate were drawn and are discussed in the paragraphs below.

5.4.1 Sampling techniques

One of the factors investigated was the effect of sampling technique on the overall performance of the models. For the ROS and SMOTE graphs in Figure 5.2, it was observed that the models resulted in similar F2 scores across the prevalence rates for the two sampling methods. For the XGBoost and C4.5 DT models, ROS showed a slight improvement in performance compared to SMOTE. Whereas for the RF and NN models, SMOTE resulted in an overall higher F2 score. The LR and lasso models performed the same across ROS and SMOTE, barring the original and 1% datasets, where the lasso model did not result in an F2 score. When comparing the ROS and SMOTE results to the RUS results, it was seen that the performance of the models drastically decreased. While the original and 1% models remained at a high F2 score, similar to that of ROS and SMOTE, the remaining prevalence rates showed a sharp decline in performance across the supervised learning methods. Using this visual, RUS would not be the optimal sampling technique to use for this specific dataset. The best

performances are generally seen by the ROS and SMOTE methods. Studies by Japkowicz and Stephen (2002) and Batista et al. (2012) corroborate the results obtained above. It was seen that ROS performed the best when dealing with imbalanced datasets in comparison to RUS and cost modification methods, while it only slightly outperformed SMOTE. In principle, it is understandable why this is the case. These oversampling techniques use the minority class samples as a basis to duplicate and sythesise new examples. There is no elimination of data, therefore allowing training to be completed on a wider set of information where patterns and trends can be picked up more easily by the algorithms.

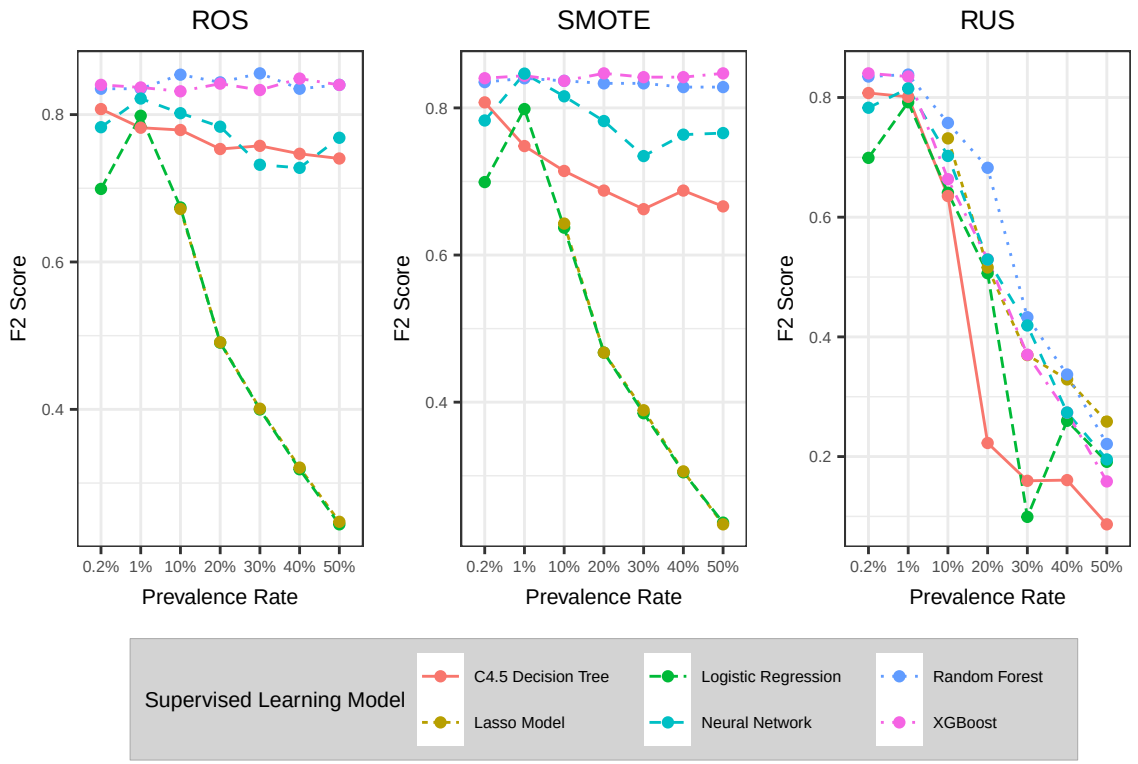

FIGURE 5.2: Depiction of F2 scores for all models across prevalence rates based on performance on test dataset for; *Left*: ROS, *Middle*: SMOTE and *Right*: RUS

Conversely, Van Hulse et al. (2007) showed that the RUS method performed better than other sampling techniques for imbalanced datasets with a less than 5% prevalence rate. The study used 35 different datasets, with prevalence rates ranging from 1.3% to 35%. This behaviour was not observed in this study and could be attributed to this specifc dataset which was used for this study. In the end, the various datasets react in different ways to these methods, allowing for variations in the results.

When comparing the metrics for each supervised learning model across sampling techniques, RUS generally performed the worst in terms of precision, but the best in recall. Since the RUS method removes some of the positive class examples, the process could be eliminating useful data and choosing examples with noise to remain in the dataset. The negative class examples

all remain in the dataset, which could be contributing to the high recall rate, where there were a large number of minority class examples correctly identified out of the total number in the minority class. The oversampling models generally showed the opposite result. The precision tended to be high, whereas the recall was low in comparison to RUS over the prevalence rates. The oversampling methods create minority class samples, thereby introducing possible noise and inaccuracies. This could lead to the lower recall values, thus lowering the precision.

5.4.2 Supervised learning techniques

Secondly, the supervised learning technique perfomance based on the datasets was analysed. The outcome of this investigation was quite interesting to note. In general, it can be seen that the best performing models were the XGBoost and RF techniques. These are followed by the NN and the C4.5 DT models. The poorest performing models were found to be the LR and lasso models. The XGBoost and RF algorithms were expected to perform well as compared to the other techniques due to its ability to discover complex patterns within data, while reducing overfitting.

Random forest is a bagging model which builds multiple trees in parallel and uses majority voting to make a decision on the final class. Each tree is built independently, therefore have no influence on each other. On the other hand, XGBoost is a boosting model, where an ensemble of trees are built sequentially, improving the model as the training occurs. Using this knowlegde, one would expect that the XGBoost model had an advantage over the RF. The results for both of these models were very similar, therefore concluding that these tree models were a good choice when dealing with imbalanced datasets as compared to the other techniques.

The C4.5 DT was expected to result in a lower performance when compared to the other tree models (XGBoost and RF). Japkowicz and Stephen (2002) used a variation of the C4.5 DT, namely C5.0 to complete their study. A similar result was found, where it was concluded that this model was sensitive to imbalance. The outcome from C4.5 and C5.0 is due to the fact that these models comprise of a single tree and do not make use of ensemble methods, which assist in increasing the predictive ability of the models. In addition, these trees are highly sensitive to changes in the data causing instability. In this case, the dataset is modified through sampling, which could lead to the lower performances measured.

The NN model was another model which performed well comparatively. While NN is not well designed to handle imbalanced data, it resulted in the third highest F2 scores. NNs optimise based on gradient descent and aims to maximise the accuracy of predictions. In the end, this means that the algorithm treats positive and negative classes in the same manner. Therefore, a poorer performance can be expected in comparison to the other models, as seen in Figure 5.2.

Lastly, the LR and lasso models performed the worst overall, across different sampling techniques. These models base its predictive capability around maximising the log-likelihood function through maximising the overall accuracy. Due to this reason, there could be a bias towards the majority class during the modelling process. This could lead to inaccurate

predictions and poor performance, which can be seen in the graph (Figure 5.2).

5.4.3 Prevalence rates

The last factor for analysis was the prevalance rates. It was interesting to note how the various models reacted to the change prevalence rates. From Figure 5.2, it can be seen that the performance of the models decreased as the dataset moved towards being balanced. In some instances, such as the RF and XGBoost models, the performance remained similar across prevalence rates for the ROS and SMOTE sampling techniques. The supervised learning techniques implemented are designed to work best with balanced datasets as they focus on: 1) maximising the overall accuracy of the predictions, 2) assume equal costing of misclassification errors on each class, and 3) follow bias model training, which favours the majority class and have difficulty capturing patterns on the minority class (Visa and Ralescu, 2005). Therefore, the result obtained from this study is unexpected. A study by Napierala and Stefanowski (2016) showed that the general trend across six different classifiers showed a directly proportional relationship with prevalence rate and performance. The more balanced the dataset, the higher the performance.

However, there could be several factors influencing the behaviour observed. One such explanation could be found in the methodology of the sampling techniques. For ROS, with the increase in the prevalence rate, comes the increase in duplication of minority class samples. This could lead to overfitting, thereby creating poor performing models. Additionally, SMOTE uses the k-Nearest Neighbour technique to create a new data point in the minority class. For this study, a k value of 5 was used. However, taking into account the sparseness of the minority class, when the algorithm creates this new data point, the assumption is that the surrounding neighbours belong to the minority class, which might not be the case in a highly imbalanced dataset. Therefore, the data points created could lead to inaccuracies during the synthesising process, leading to a poor performing model. The performance of the RUS models also sees a decline in performance as the prevalence rate increased. With undersampling, majority class examples are removed from the dataset thus potentially leading to valuable data being lost. As the prevalence rate increases, more data are being removed, which could lead to the poor performance of the models as the dataset moved to a more balanced one.

5.5 Best performing models

The best performing models per each supervised learning technique were summarised into Table 5.8 below. These results can be analysed in conjuction with the results and discussion from Figure 5.2. The overall best performer, in terms of F2 score, was the RF model at 30% prevalence rate using ROS as the sampling technique. Certain trends could be picked up from this table, confirming the analysis in Section 5.4. The first one being the sampling techniques which produced the best performances across supervised learning method. Here, it was seen that ROS and SMOTE were the techniques resulting in the best performances. Secondly, one can observe that the lower prevalence rates generally ranked the highest in terms of F2 score. The

XGBoost and RF models were exceptions as they performed best at a higher prevalence rate. These insights were also gathered from the previous sections and led to the same conclusions.

Supervised Learning Model	Prevalence Rate	Sampling Technique	Precision	Recall	F2 Score
Logistic Regression Model	1%	ROS	0.73	0.82	0.80
Logistic Regression Model	1%	SMOTE	0.73	0.82	0.80
Logistic Regression with Lasso	10%	RUS	0.46	0.86	0.73
C4.5 Decision Tree Model	Original	NA	0.86	0.80	0.81
Random Forest Model	30%	ROS	0.94	0.84	0.86
XG Boost Model	40%	ROS	0.90	0.84	0.85
Neural Network Model	1%	SMOTE	0.77	0.87	0.85

TABLE 5.8: Best F2 scores per supervised learning technique based on performance on test dataset

While the RF model at 30% prevalence rate using ROS showed the best F2 score, it was worthwhile noting that the NN model at 1% prevalence rate using SMOTE had a similar value, however resulted in different precision and recall values. It is interesting to understand the compromise which would need to be made between the cost of false positives versus that of false negatives. For the application of fraud detection, a higher cost on a false negatives would be required. An initial analysis was completed on the confusion matrices of these two models (seen in Tables 5.9 and 5.10).

From the matrices, RF resulted in more false negatives than NN (16 versus 13). However, the false positives were higher in the NN model (25) when compared to RF (5). Relating these results back to the intention of fraud detection, one would prefer the model yielding fewer false negatives at the cost of having a higher number of false positives. Therefore, a decision could be made to use NN as the model of choice for this dataset.

		Reference	
		0	1
Prediction	0	56858	16
	1	5	82

TABLE 5.9: Confusion matrix on test dataset for RF model with 30% prevalence rate using ROS

		Reference	
		0	1
Prediction	0	56838	13
	1	25	85

TABLE 5.10: Confusion matrix on test dataset for NN model with 1% prevalence rate using SMOTE

5.6 Conclusion

This chapter presented an overview of the results obtained from the six experiments conducted in this investigation. The results were interpreted and an analysis into the comparison of the models was given. Key points were highlighted in the chapter, which will be reiterated in the following chapter, together with a conclusion and recommendations for future studies.

Chapter 6

Conclusions and Recommendations

6.1 Conclusions

Application of machine learning methods in the detection of fraud has been on the increase over the past decade. One such application is in the detection of credit card fraud. The challenge in this case study comes with the imbalanced nature of these datasets. While there have been methods to overcome this issue, research is ongoing to improve the predictive capability of these algorithms. This study is one that focused on the imbalanced nature of datasets and its reaction to different supervised learning methods and sampling techniques.

The overall objective of this study was to determine the effect of different level of imbalance in a dataset on the performance of classifiers. In this study, five different supervised learning models were used, these being LR, C4.5 DT, RF, XGBoost and NN. Three sampling techniques were used to create the varying prevalence rates: ROS, SMOTE and RUS. This objective was answered through a series of experimentation.

The first factor analysed was the effect of different sampling techniques on the performance of the models. In this case, it was found that the ROS and SMOTE models resulted in the better performing classifiers based on F2 score. The explanation around this outcome can be found in the theory of oversampling versus undersampling. The undersampling method removes examples from the dataset, possibly removing important information and reducing the performance of the models. In terms of precision, RUS performed the worst, but performed the best in recall. Once again, this can be attributed to the method by which the algorithm removes samples for the dataset. The majority class samples are removed while all minority class samples remain, thus allowing for the results presented.

Focusing on the supervised learning models, it was seen that RF and XGBoost models performed the best across all models built. It was concluded that the tree models are well designed to handle this specific credit card transaction dataset and could be applied to similar datasets. The C4.5 DT model performed fairly well, however, one could see limitations when compared to the ensemble methods of RF and XGBoost. NN performed well comparatively, however is not well-suited to imbalanced problems as it aims to maximise the overall accuracy. The LR and Lasso models were the worst performing classifiers. One explanation

for this result was the algorithm's bias towards the majority class as its goals is to maximise the overall accuracy.

The investigation around prevalence rates saw an interesting discovery being made. For this specific dataset, the overall performance of the models decreased as the prevalence rates increased. Once again using the theory behind the sampling techniques, whereby examples are either added or removed, could introduce a level of inaccuracy into the training of the models, resulting in the poor performance as the prevalence rate increased.

The secondary objective was investigating how a change in threshold value would affect the models performance. The 1% ROS model for LR was used as the basis of this study. The results showed that the optimal threshold value was obtained at 0.1, which showed the highest performance.

This study provided a comprehensive overview of the factors that could be used for future work to overcome the problem of imbalanced datasets. The results from this dissertation can provide a good baseline of which supervised learning models to use with certain levels of imbalance in a dataset, as well as which sampling technique to choose to optimise model performance.

6.2 Recommendations and future work

The limitations and recommendations of this study are listed below. Due to the nature of the study, which contained a large volume of models required, and time contraints, these points could be improved in future studies:

- During the model building stages of the study, default hyperparameters or hyperparameters used in previous literature were used. This route was chosen due to the volume of models that were required to be built. For future studies, introducing hyperparameter tuning into the models would produce better performances.
- The sampling techniques only focused on the commonly used ROS, SMOTE and RUS. As with the other limitations, this was due to the high number of models being worked with in the study. Further exploration could be done to include other sampling techniques into the experiments.
- Five supervised learning techniques were used in this study, but could be expanded to include other classifiers.
- To understand the full extent of various prevalence rates on the performance of models, other imbalanced datasets could be studied to determine how these would perform under similar experimental conditions.
- A threshold investigation was completed on only the 1% ROS LR model as a indication of how the value changed based on this dataset. Future work could include a more thorough investigation on the optimal threshold values across all models.

Appendix A

Sampling Results

Before building the models, sampling techniques were applied to the datasets. In this instance, 3 techniques were be used: ROS, SMOTE and RUS. Eighteen different datasets were generated, across the varying prevalence rates chosen, which was described in Section 4.4.3. The training dataset was used to complete the resampling. The test data remained untouched as this needed to represent real-life data and not include synthesised data, which could result from the sampling.

The below table, Table A.1, shows the resulting dataset information produced per sampling technique and prevalence rate.

Imbalance %	**Sampling Method**	**Number of minority examples in dataset**	**Number of majority examples in dataset**
	ROS	2356	227452
1%	SMOTE	2758	227452
	RUS	394	39000
	ROS	25289	227452
10%	SMOTE	25610	227452
	RUS	394	3550
	ROS	57118	227452
20%	SMOTE	57130	227452
	RUS	394	1579
	ROS	97689	227452
30%	SMOTE	97712	227452
	RUS	394	926
	ROS	151711	227452
40%	SMOTE	152084	227452
	RUS	394	596
	ROS	227353	227452
50%	SMOTE	227338	227452
	RUS	394	385

TABLE A.1: Summary of dataset generated across all sampling techniques and prevalence rates

Appendix B

AUC Results

One of the popular metrics to evaluate performance on imbalanced datasets is the AUC value. The initial objective of this study was to utilise this metric in analysing the models, however once the results were produced, the AUC values for the models did not offer much variance to complete a thorough investigation. These results can be seen in Table B.1 and Figure B.1. Upon observation, the performances of the models using AUC were stable across prevalence rates, sampling techniques and choice of classifier.

Comparison of models

One noticable trend found related to the lasso model. For the original and 1% datasets, the AUC values were 0.5. In AUC terms, a value of 0.5 is a model that produces predictions no better than random guessing and is fruitless. The high imbalances for the lasso model resulted in models that did not converge and were not interpretable. Therefore, this result was expected. Other models performed well using AUC as a basis.

From Figure B.1, one can note that there were minimal changes in AUC for the models between sampling techniques. For LR and lasso model, the increase in prevalence rates brought about an slight increase in AUC across the various models. Most models remained constant across prevalence rates, concluding that the prevalence rates did not impact the AUC values significantly. Lastly, the differences in AUC between classifier were minimal. The RF, XGBoost, C4.5 DT and NN all performed fairly similarly. The LR and lasso model strayed from this pattern. At high imbalances, the AUC values were low whereas, however increases significantly as the dataset became more balanced. Eventually, these models resulted in the best AUC at 50% prevalance rate.

Prevalence Rate	Sampling Technique	Logistic Regression	Lasso Model	C4.5 Decision Tree	Random Forest	XGBoost	Neural Network
Original	NA	0.84	0.50	0.90	0.91	0.91	0.88
1%	ROS	0.91	0.50	0.90	0.91	0.91	0.92
1%	SMOTE	0.91	0.50	0.90	0.91	0.91	0.93
1%	RUS	0.91	0.50	0.92	0.93	0.92	0.92
10%	ROS	0.94	0.94	0.90	0.92	0.91	0.93
10%	SMOTE	0.93	0.94	0.93	0.92	0.92	0.93
10%	RUS	0.94	0.93	0.93	0.93	0.94	0.93
20%	ROS	0.95	0.95	0.91	0.91	0.91	0.93
20%	SMOTE	0.93	0.93	0.91	0.92	0.92	0.93
20%	RUS	0.94	0.94	0.93	0.94	0.95	0.94
30%	ROS	0.95	0.95	0.91	0.92	0.91	0.92
30%	SMOTE	0.94	0.94	0.92	0.92	0.92	0.92
30%	RUS	0.93	0.95	0.93	0.94	0.96	0.95
40%	ROS	0.95	0.95	0.92	0.91	0.92	0.93
40%	SMOTE	0.95	0.95	0.91	0.92	0.92	0.93
40%	RUS	0.96	0.95	0.93	0.95	0.95	0.95
50%	ROS	0.95	0.95	0.91	0.91	0.91	0.93
50%	SMOTE	0.96	0.95	0.91	0.92	0.92	0.93
50%	RUS	0.95	0.95	0.92	0.94	0.94	0.95

TABLE B.1: AUC results for all models

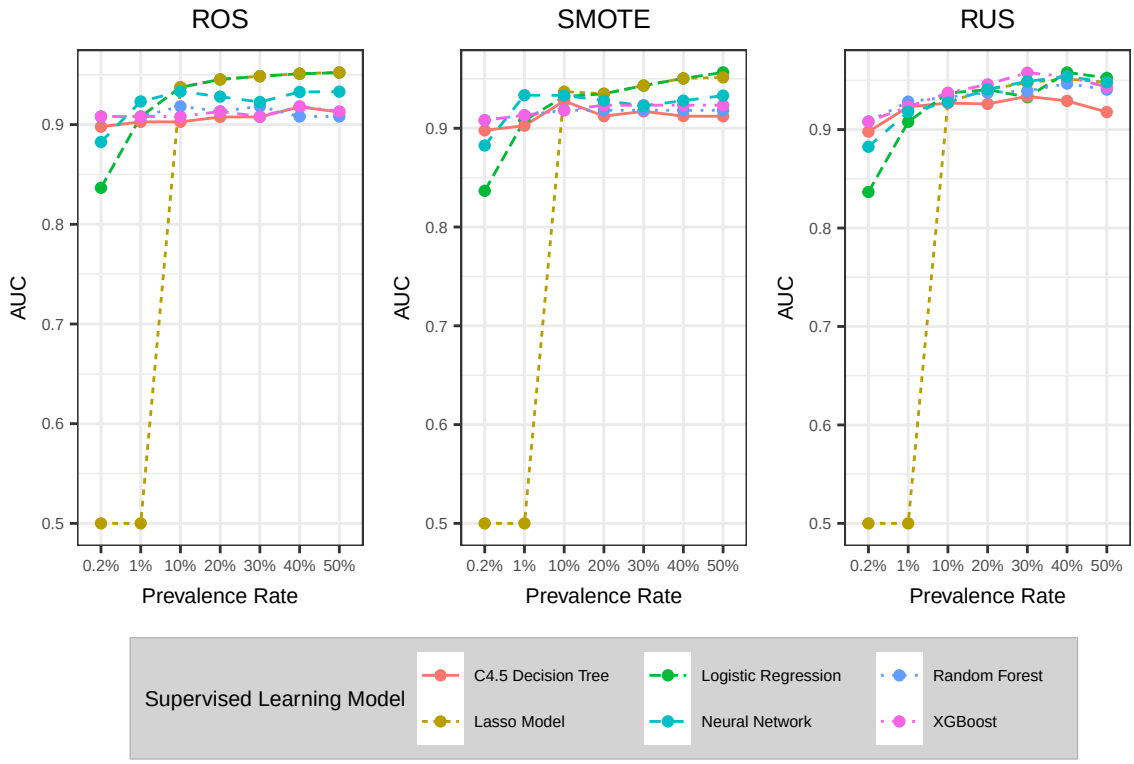

FIGURE B.1: Depiction of AUC for all models across prevalence rates for; *Left*: ROS, *Middle*: SMOTE and *Right*: RUS

Bibliography

A. Q. Abdulghani, O. N. Uçan, and K. M. A. Alheeti. Credit card fraud detection using xgboost algorithm. In *2021 14th International Conference on Developments in eSystems Engineering (DeSE)*, pages 487–492. IEEE, 2021.

A. Ali, S. M. Shamsuddin, and A. Ralescu. Classification with class imbalance problem: A review. 7:176–204, 01 2015.

H. Ali, M. Salleh, R. Saedudin, K. Hussain, and M. Mushtaq. Imbalance class problems in data mining: A review. *Indonesian Journal of Electrical Engineering and Computer Science*, 14, 03 2019. doi: 10.11591/ijeecs.v14.i3.pp1552-1563.

V. K. Ayyadevara. Random forest. In *Pro Machine Learning Algorithms*, pages 105–116. Springer, 2018.

R. M. Aziz, M. F. Baluch, S. Patel, and P. Kumar. A machine learning based approach to detect the ethereum fraud transactions with limited attributes. *Karbala International Journal of Modern Science*, 8(2):139–151, 2022.

M. Bach, A. Werner, and M. Palt. The proposal of undersampling method for learning from imbalanced datasets. *Procedia Computer Science*, 159:125–134, 01 2019. doi: 10.1016/j.procs.2019.09.167.

S.-H. Bae and K.-J. Yoon. Polyp detection via imbalanced learning and discriminative feature learning. *IEEE Transactions on Medical Imaging*, 34(11):2379–2393, 2015. doi: 10.1109/TMI.2015.2434398.

G. Batista, R. Prati, and M.-C. Monard. A study of the behavior of several methods for balancing machine learning training data. *SIGKDD Explorations*, 6:20–29, 06 2004. doi: 10.1145/1007730.1007735.

G. Batista, D. Silva, and R. Prati. An experimental design to evaluate class imbalance treatment methods. volume 2, 12 2012. doi: 10.1109/ICMLA.2012.162.

M. Beckmann, N. Ebecken, and B. Lima. A knn undersampling approach for data balancing. *Journal of Intelligent Learning Systems and Applications*, 7:104–116, 11 2015. doi: 10.4236/jilsa.2015.74010.

S. Bhattacharyya, S. Jha, K. Tharakunnel, and J. C. Westland. Data mining for credit card fraud: A comparative study. *Decision support systems*, 50(3):602–613, 2011.

J. Błaszczyński, M. Deckert, J. Stefanowski, and S. Wilk. Integrating selective pre-processing of imbalanced data with ivotes ensemble. In M. Szczuka, M. Kryszkiewicz, S. Ramanna, R. Jensen, and Q. Hu, editors, *Rough Sets and Current Trends in Computing*, pages 148–157, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg. ISBN 978-3-642-13529-3.

L. Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

J. Brownlee. *XGBoost With python: Gradient boosted trees with XGBoost and scikit-learn*. Machine Learning Mastery, 2016.

K. Chaudhary, J. Yadav, and B. Mallick. A review of fraud detection techniques: Credit card. *International Journal of Computer Applications*, 45, 01 2012.

N. Chawla. *Data Mining for Imbalanced Datasets: An Overview*, volume 5, pages 853–867. 01 2005. doi: 10.1007/0-387-25465-X_40.

N. Chawla, K. Bowyer, L. Hall, and W. Kegelmeyer. Smote: Synthetic minority over-sampling technique. *J. Artif. Intell. Res. (JAIR)*, 16:321–357, 06 2002. doi: 10.1613/jair.953.

N. V. Chawla, A. Lazarevic, L. O. Hall, and K. W. Bowyer. Smoteboost: Improving prediction of the minority class in boosting. In *European conference on principles of data mining and knowledge discovery*, pages 107–119. Springer, 2003.

T. Chen and C. Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

M. Dorn, B. I. Grisci, P. H. Narloch, B. C. Feltes, E. Avila, A. Kahmann, and C. S. Alho. Comparison of machine learning techniques to handle imbalanced covid-19 cbc datasets. *PeerJ Computer Science*, 7:e670, 2021.

A. Fernández, S. García, M. Galar, R. C. Prati, B. Krawczyk, and F. Herrera. *Learning from imbalanced data sets*, volume 10. Springer, 2018.

A. Fernández, S. García, and F. Herrera. Addressing the classification with imbalanced data: Open problems and new challenges on class distribution. pages 1–10, 01 2011.

A. Fernández, S. García, M. Galar, R. Prati, B. Krawczyk, and F. Herrera. *Learning from Imbalanced Data Sets*. 01 2018. ISBN 978-3-319-98073-7. doi: 10.1007/978-3-319-98074-4.

M. Galar, A. Fernández, E. Barrenechea, H. Sola, and F. Herrera. A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 42:463 – 484, 07 2012. doi: 10.1109/TSMCC.2011.2161285.

L. Gao and S. Siu. Study of data imbalanced problem in protein-peptide binding prediction. pages 61–66, 05 2020. doi: 10.1145/3405758.3405764.

J. Gareth, W. Daniela, H. Trevor, and T. Robert. *An introduction to statistical learning: with applications in R*. Spinger, 2013.

S. Georgieva, M. Markova, and V. Pavlov. Using neural network for credit card fraud detection. volume 2159, page 030013, 10 2019. doi: 10.1063/1.5127478.

S. Ghosh and D. L. Reilly. Credit card fraud detection with a neural-network. In *System Sciences, 1994. Proceedings of the Twenty-Seventh Hawaii International Conference on*, volume 3, pages 621–630. IEEE, 1994.

K. Gwirtz, M. Morzfeld, A. Fournier, and G. Hulot. Can one use earth's magnetic axial dipole field intensity to predict reversals? *Geophysical Journal International*, 225(1):277–297, 2021.

G. Haixiang, L. Yijing, J. Shang, G. Mingyun, H. Yuanyue, and G. Bing. Learning from class-imbalanced data: Review of methods and applications. *Expert Systems with Applications*, 73:220–239, 2017. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2016.12.035. URL https://www.sciencedirect.com/science/article/pii/S0957417416307175.

H. Han, W.-Y. Wang, and B.-H. Mao. Borderline-smote: A new over-sampling method in imbalanced data sets learning. volume 3644, pages 878–887, 09 2005. ISBN 978-3-540-28226-6. doi: 10.1007/11538059_91.

M. Hanafy and R. Ming. Machine learning approaches for auto insurance big data. *Risks*, 9(2): 42, 2021.

H. He and E. A. Garcia. Learning from imbalanced data. *IEEE Transactions on knowledge and data engineering*, 21(9):1263–1284, 2009.

A. Husejinovic. Credit card fraud detection using naive bayesian and c4. 5 decision tree classifiers. *Husejinovic, A.(2020). Credit card fraud detection using naive Bayesian and C*, 4:1–5, 2020.

Y. Jain, N. Tiwari, S. Dubey, and S. Jain. A comparative analysis of various credit card fraud detection techniques. *International Journal of Recent Technology and Engineering*, 7:402–407, 01 2019.

N. Japkowicz. The class imbalance problem: Significance and strategies. In *In Proceedings of the 2000 International Conference on Artificial Intelligence (ICAI*, pages 111–117, 2000.

N. Japkowicz. Concept-learning in the presence of between-class and within-class imbalances. In E. Stroulia and S. Matwin, editors, *Advances in Artificial Intelligence*, pages 67–77, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg. ISBN 978-3-540-45153-2.

N. Japkowicz and S. Stephen. The class imbalance problem: A systematic study. *Intell. Data Anal.*, 6:429–449, 11 2002. doi: 10.3233/IDA-2002-6504.

T. Jemima Jebaseeli, R. Venkatesan, and K. Ramalakshmi. Fraud detection for credit card transactions using random forest algorithm. In *Intelligence in Big Data Technologies—Beyond the Hype*, pages 189–197. Springer, 2021.

D. T. Jo and N. Japkowicz. Class imbalances versus small disjuncts. *SIGKDD Explorations*, 6: 40–49, 06 2004a. doi: 10.1145/1007730.1007737.

D. T. Jo and N. Japkowicz. Class imbalances versus small disjuncts. *SIGKDD Explorations*, 6: 40–49, 06 2004b. doi: 10.1145/1007730.1007737.

S. Joseph, E. Dada, D. Mshelia, and A. Hamidu Alkali. Students graduation on time prediction model using artificial neural network. 21:28–35, 05 2019. doi: 10.9790/0661-2103012835.

K. Kamusweke, M. Nyirenda, and M. Kabemba. Data mining for fraud detection in large scale financial transactions. *EasyChair*, (1729), 2019.

M. Kirlidog and C. Asuk. A fraud detection approach with data mining in health insurance. *Procedia-Social and Behavioral Sciences*, 62:989–994, 2012.

J. Kittler, M. Hater, and R. Duin. Combining classifiers. In *Proceedings of 13th International Conference on Pattern Recognition*, volume 2, pages 897–901 vol.2, 1996. doi: 10.1109/ICPR.1996.547205.

B. Krawczyk. Learning from imbalanced data: Open challenges and future directions. *Progress in Artificial Intelligence*, 5, 04 2016. doi: 10.1007/s13748-016-0094-0.

M. Kuhn. Package 'caret'. the r project for statistical computing, 2014.

A. Kulkarni, D. Chong, and F. Batarseh. Foundations of data imbalance and solutions for a data democracy, 07 2021.

S. Lakshmi and S. D. Kavilla. Machine learning for credit card fraud detection system. *International Journal of Applied Engineering Research*, 13(24):16819–16824, 2018.

N. Lunardon, G. Menardi, and N. Torelli. Rose: A package for binary imbalanced learning. *R journal*, 6(1), 2014.

V. López, A. Fernández, S. García, V. Palade, and F. Herrera. An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. *Information Sciences*, 250:113–141, 2013. ISSN 0020-0255. doi: https://doi.org/10.1016/j.ins.2013.07.007. URL https://www.sciencedirect.com/science/article/pii/S0020025513005124.

Machine Learning Group. Credit card fraud detection, 2018. URL https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud.

S. Makki, Z. Assaghir, Y. Taher, R. Haque, M.-S. Hacid, and H. Zeineddine. An experimental study with imbalanced classification approaches for credit card fraud detection. *IEEE Access*, 7:93010–93022, 2019. doi: 10.1109/ACCESS.2019.2927266.

N. McCullum. Deep learning neural networks explained in plain english. *FreeCodeCamp. Saatavissa: https://www. freecodecamp. org/news/deep-learning-neural-networks-explained-in-plain-english/. Hakupäivä*, 25:2021, 2020.

C. Meng, L. Zhou, and B. Liu. A case study in credit fraud detection with smote and xgboost. In *Journal of Physics: Conference Series*, volume 1601, page 052016. IOP Publishing, 2020.

M. M. Mijwil and I. E. Salem. Credit card fraud detection in payment using machine learning classifiers. *Asian Journal of Computer and Information Systems*, 8(4), 2020.

R. Mohammed, J. Rawashdeh, and M. Abdullah. Machine learning with oversampling and undersampling techniques: Overview study and experimental results. pages 243–248, 04 2020. doi: 10.1109/ICICS49469.2020.239556.

K. K. Mohbey, M. Z. Khan, and A. Indian. Credit card fraud prediction using xgboost: An ensemble learning approach. *International Journal of Information Retrieval Research (IJIRR)*, 12 (2):1–17, 2022.

T. Næs, K. Kvaal, T. Isaksson, and C. Miller. Artificial neural networks in multivariate calibration. *Journal of Near Infrared Spectroscopy*, 1(1):1–11, 1993.

K. Napierala and J. Stefanowski. Types of minority class examples and their influence on learning classifiers from imbalanced data. *Journal of Intelligent Information Systems*, 46(3): 563–597, 2016.

C. Nwankwo and C. C. Ihueze. Corrosion rate models for oil and gas pipeline systems: A numerical approach. *International Journal of Engineering Research and*, V7, 08 2018. doi: 10.17577/IJERTV7IS080082.

H. Patel, D. Rajput, T. Gadekallu, C. Iwendi, A. Bashir, and O. Jo. A review on classification of imbalanced data for wireless sensor networks. *International Journal of Distributed Sensor Networks*, 16:155014772091640, 04 2020. doi: 10.1177/1550147720916404.

R. Patidar, L. Sharma, et al. Credit card fraud detection using neural network. *International Journal of soft computing and Engineering (IJSCE)*, 1(32-38), 2011.

P. Prabha and C. Priscilla. Influence of optimizing xgboost to handle class imbalance in credit card fraud detection. 10 2020. doi: 10.1109/ICSSIT48917.2020.9214206.

R. C. Prati, G. E. A. P. A. Batista, and M. C. Monard. Learning with class skews and small disjuncts. In A. L. C. Bazzan and S. Labidi, editors, *Advances in Artificial Intelligence – SBIA 2004*, pages 296–306, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg. ISBN 978-3-540-28645-5.

C. V. Priscilla and D. P. Prabha. Influence of optimizing xgboost to handle class imbalance in credit card fraud detection. In *2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT)*, pages 1309–1315. IEEE, 2020.

J. R. Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986.

J. R. Quinlan. *C4. 5: programs for machine learning*. Elsevier, 2014.

S. Ramraj, N. Uzir, R. Sunil, and S. Banerjee. Experimenting xgboost algorithm for prediction and classification of different datasets. *International Journal of Control Theory and Applications*, 9(40), 2016.

D. Ramyachitra and P. Manikandan. Imbalanced dataset classification and solutions : A review. 2014.

P. Rim and E. Liu. Optimizing the c4. 5 decision tree algorithm using msd-splitting. *International Journal of Advanced Computer Science and Applications*, 11(10):41–47, 2020.

Y. Sahin and E. Duman. Detecting credit card fraud by ann and logistic regression. In *2011 international symposium on innovations in intelligent systems and applications*, pages 315–319. IEEE, 2011.

B. Santoso, H. Wijayanto, K. Notodiputro, and B. Sartono. Synthetic over sampling methods for handling class imbalanced problems : A review. *IOP Conference Series: Earth and Environmental Science*, 58:012031, 03 2017. doi: 10.1088/1755-1315/58/1/012031.

S. Singh and P. Gupta. Comparative study id3, cart and c4. 5 decision tree algorithm: a survey. *International Journal of Advanced Information Science and Technology (IJAIST)*, 27(27):97–103, 2014.

W. Siriseriwan. Smotefamily: A collection of oversampling techniques for class imbalance problem based on smote. 2018. *URL http://cran. r-project. org/package= smotefamily*, 2021.

S. Sperandei. Understanding logistic regression analysis. *Biochemia medica*, 24(1):12–18, 2014.

Y. Sun, M. S. Kamel, A. K. Wong, and Y. Wang. Cost-sensitive boosting for classification of imbalanced data. *Pattern Recognition*, 40(12):3358–3378, 2007. ISSN 0031-3203. doi: https://doi.org/10.1016/j.patcog.2007.04.009. URL https://www.sciencedirect.com/science/article/pii/S0031320307001835.

Y. Sun, A. K. Wong, and M. S. Kamel. Classification of imbalanced data: A review. *International journal of pattern recognition and artificial intelligence*, 23(04):687–719, 2009.

K. M. Ting. A comparative study of cost-sensitive boosting algorithms. In *In Proceedings of the 17th International Conference on Machine Learning*. Citeseer, 2000.

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

J. Van Hulse, T. Khoshgoftaar, and A. Napolitano. Experimental perspectives on learning from imbalanced data. volume 227, pages 935–942, 01 2007. doi: 10.1145/1273496.1273614.

S. Visa and A. Ralescu. Issues in mining imbalanced data sets - a review paper. *Proc. 16th Midwest Artificial Intelligence and Cognitive Science Conference*, 01 2005.

A. A. Vorobeva. Examining the performance of classification algorithms for imbalanced data sets in web author identification. In *2016 18th Conference of Open Innovations Association and Seminar on Information Security and Protection of Information Technology (FRUCT-ISPIT)*, pages 385–390, 2016. doi: 10.1109/FRUCT-ISPIT.2016.7561554.

L.-X. Wang and J. Mendel. Generating fuzzy rules by learning from examples. *IEEE Transactions on Systems, Man, and Cybernetics*, 22(6):1414–1427, 1992. doi: 10.1109/21.199466.

W. Wang, G. Chakraborty, and B. Chakraborty. Predicting the risk of chronic kidney disease (ckd) using machine learning algorithm. *Applied Sciences*, 11:202, 12 2020. doi: 10.3390/app11010202.

G. Weiss. Mining with rarity: A unifying framework. *SIGKDD Explorations*, 6:7–19, 01 2004.

G. Weiss. *The Impact of Small Disjuncts on Classifier Learning*, volume 8, pages 193–226. 10 2010. ISBN 978-1-4419-1279-4. doi: 10.1007/978-1-4419-1280-0_9.

G. Weiss and F. Provost. Learning when training data are costly: The effect of class distribution on tree induction. *J. Artif. Intell. Res. (JAIR)*, 19:315–354, 07 2003. doi: 10.1613/jair.1199.

M. Wozniak. *Hybrid classifiers. Methods of Data, Knowledge, and Classifier Fusion*, volume 519. 10 2013. ISBN 978-3-642-40996-7. doi: 10.1007/978-3-642-40997-4_1.

J. Yang, J. Gong, W. Tang, Y. Shen, C. Liu, and J. Gao. Delineation of urban growth boundaries using a patch-based cellular automata model under multiple spatial and socio-economic scenarios. *Sustainability*, 11(21):6159, 2019.

Z.-H. Zhou and X.-Y. Liu. On multi-class cost-sensitive learning. *Computational Intelligence*, 26: 232–257, 08 2010. doi: 10.1111/j.1467-8640.2010.00358.x.