Credit Card Fraud Detection System using Extreme Gradient Boosting Machine and Isolated Forest

---

Tselahale Lloyd Serongwa

Supervisor(s):
Prof Wilbert Chagwiza

A research report submitted in partial fulfillment of the requirements for the degree of Master of Science in the field of e-Science

in the

School of Computer Science and Applied Mathematics
University of the Witwatersrand, Johannesburg

12 January 2023

Contents

List of Figures

List of Tables

Declaration

I, Tselahale Lloyd Serongwa, declare that this research report is my own, unaided work. It is being submitted for the degree of Master of Science in the field of e-Science at the University of the Witwatersrand, Johannesburg. It has not been submitted for any degree or examination at any other university.

Tselahale Lloyd Serongwa

12 January 2023

Abstract

An extreme gradient boosting machine (a supervised technique) and an isolation forest model (an unsupervised technique) are compared in the development of a credit card fraud detection system based on the misuse of credit cards. Simulated data from Sparkov Data Generation was used with 1 296 675 data points and 99.4% of the transactions being non-fraudulent. Adasyn was used to sample the dataset and the weight of evidence was used to bin classes and reduce dimensionality. XGBoost resulted in a superior model as iForest predictors failed to scale up to any variation of the XGBoost predictors. The top eleven predictive variables are from five original variables, the time, age, job, amount and the merchant. These had a combined prediction power of 48.2%. This work established that temporal information is a key aspect of the dataset. The date and timestamp established features which resulted in 18% total predictive power, the highest predictive variable for the optimal model. An aggregation of the amount and age was amongst the top predictors at 8.5% predictive power. It was shown that supervised techniques yield reliable and powerful predictive models in dynamic multi-dimensional mixed data type problems as compared to their unsupervised counterparts. The complex dynamic nature of fraud detection dataset is more compatible to a model that learns from labels as many fraudulent transactions mimic legitimate transactions closely. It was gathered however, that reducing the number of features improved the performance of an iForest model in a multi-dimensional dataset. The optimal XGBoost model learnt at a 0.1 rate, built decision trees 10 levels deep with 40 estimators. The model had an average of 98% f1-score, 0.96 Cohen's kappa and 96% gini coefficient for the testing dataset as compared to 98% f1-score, 0.973 Cohen's kappa and 97.3% gini coefficient for the training dataset. The optimal model is stable and robust as no single variable contributes more than 18% total predictive power.

Acknowledgements

I would like to extend my gratitude to Prof Wilbert Chagwiza for his guidance in the work presented in this document and more so, the National e-Science Postgraduate Teaching and Training Platform for the funding that made this work possible.

Chapter 1

Introduction

1.1 Background

Fraud is the intentional deception and illegitimate acquisition of financial gains [15]. In 2016, it was estimated that fraud could account for $10 billion of transactions in the United States of America alone. The instances of successful fraudulent transactions are in comparison way less than those which are legitimate (1 in 10 000) in some practical cases [26]. Fraud can harm the reputation of financial institutions and can result in significant losses for customers [26]. In 2018, Experian reported that 65% of businesses have had an increased number of fraudulent activities that impacted their revenue and returns [13]. Of all these businesses, only 40% had confidence in the systems they use to detect fraudulent activities [13].

Fraudulent activities are dynamic, continuously changing its characteristics with the advancement of technology [1]. Fraudsters are getting better at emulating legitimate transactions, and this poses increased risk to financial institutions. Credit card fraud is the most dominant of all the fraud activities and it is that which many researchers have enough data to investigate [1, 5]. The early detection and interception of fraudulent activities in electronic transactions is a growing concern and still require robust systems to combat. Modern techniques need to be explored as traditional techniques employed rigid rules and have been proven to be simple but attracts high maintenance costs as financial experts are needed to draft the rules [50]. Customers want to be certain that the institutions they entrust their finances with have effective measures in place to ensure that only genuine transactions are processed on their credit cards [37]. Customers also do not want their legitimate transactions to be flagged and blocked as being fraudulent. Credit card fraud data is scarce as businesses may choose to not publicise their exposure to attacks in order to avoid bad reputation and unwanted publicity [37]. Many fraud insights are not publicised in open domains in order to prevent notifying fraudsters on ways to avoid detections [15, 37].

This paper seeks to compare an extreme gradient boosting (XGBoost) machine and an isolation forest (iForest) model in the development of a credit card fraud detection system based on multidimensional mixed data type credit card dataset. The comparison is extended to the sampling techniques where the synthetic minority oversampling techniques will be used. The models will be evaluated by accuracy, precision, recall, f1-score, Cohen's kappa, gini co-efficient, Kolmogorov-Smirnov

statistic plot and the area under the receiver operating characteristic curve metrics.

The rest of this chapter presents the problem statement in Section 1.2, the research questions are listed in Section 1.3 and the research objectives are defined in Section 1.4. The significance of the research work is outlined in Section 1.5. The study limitation and delimitations are provided in Section 1.6 and 1.7 before the thesis overview is presented in Section 1.8.

1.2 Problem Statement

Various approaches have been implemented in the development of credit card fraud detection systems. There has not been however, a solid permanent solution to date due to the dynamic nature of fraud and new data becoming available daily. Research work in [3, 18, 38, 29], which explored the idea of comparing a supervised and unsupervised machine model did not explore the effect of aggregate features in the predictive performance of their detection systems. In [18, 29], the comparison of the supervised and unsupervised models were done using purely numeric variables from data that has been processed using Principal Component Analysis (PCA). PCA is a linear decomposition technique to reduce the dimensions of the dataset to a lower, more interpretable dimensional plane [40].

The purpose of this research work is to build a robust and accurate model to detect credit card fraud by comparing the XGBoost and an iForest model. An in-depth comparison will be done to compare the predictive performance of these two well-established methods using multi-dimensional mixed data type credit card dataset. The comparison is extended to the sampling techniques where the synthetic minority oversampling techniques will be used. Features will be engineered and evaluated to establish the best aggregates that enhance the predictive power of the final credit card fraud detection model.

1.3 Research Questions

The two models, the supervised and unsupervised will be built and the following questions will be answered:

a) Which features provide the best predictive strength in indentifying a fraudulent transaction?
b) Which combination of attributes improves the predictive performance of the model?
c) Which synthentic minority oversampling technique improves the predictive performance of the model?

1.4 Research Aims and Objectives

The aim of this work is to build and determine the most effective credit card fraud detection system between the XGBoost and the iForest model.

The primary objective of this study is to determine the most robust approach available with which to detect fraudulent transactions. This will be accomplished by the following secondary objectives that will achieve the aim and answer the outlined research questions:

a) To determine which model between the XGBoost and the iForest has superior fraud prediction strength.
b) To determine features that have the greatest predictive power in detecting fraudulent credit card transactions.
c) To determine optimal aggregates and ratios that improve the model's performance.
d) To determine the most effective synthentic minority oversampling technique that improve the predictive performance of the model.

1.5 Significance of the Study

Various models have been built to tackle credit card fraud problems to provide a solution to this billion-dollar industry that continuously trouble the financial sector. The problem persists as a result of many traditional models scaling very poorly against the availability of streaming, big and messy data. It has been shown that no amount of feature engineering can act as a magic tool to extract non-existent information from a poor or messy dataset [52]. A good dataset must be sought, and feature engineering techniques chosen scientifically.

A good model depends on the accuracy and reliability of historical data. Work in [18, 29] that compared unsupervised and supervised machine learning models in detecting credit card fraud used data that has been processed using PCA to build their solutions. PCA is a decomposition technique that can only be applied to linear, numeric variables that can be set up as $m \times n$ matrices and eigenvectors [44]. Multiple researchers, [3, 10, 15, 47], recommend exploring feature creation, aggregation techniques and sampling methods to find the optimal combination that results in a robust system.

A model's fitness for purpose is based on opacity, scalability and damage [30]. Sustainable model must be transparent, have the capability to scale up and do so fairly as not to be classified as what Cathy O'Neil term *weapons of math destruction* [30]. These are complex mathematical models that are not clear in how they predict outcomes that impact human lives [30]. This work takes advantage of the availability of vast amount of data and aims to be one of the few to compare a supervised and unsupervised model in a credit card fraud detection system. There has also been a limited number of research work that either employed XGBoost, isolation forest or a comparison of the two well

established machine learning techniques without using data processed by PCA or purely numeric datasets. None has extensively investigated the effect of feature aggregates or ratios on the predictive performance of credit card detection systems. The work aims to address that gap by using modern techniques and multi-dimensional mixed data type dataset to further enrich the credit card fraud detection knowledge area.

1.6 Limitations

The study is limited by the following key aspects:

- Censored real data. The datasets that are available in the public domain are mostly synthetic or heavily censored. This narrows down the amount of insights that are likely to be extracted from credit card data and makes it difficult to benchmark models with those in real life applications.
- Fraud is an evolving industry and as such, the behaviour in credit card datasets is dynamic [38].
- Fraudulent transactions are very few compared to legitimate ones. Most if not all datasets in the public domain is as a result highly skewed towards the legitimate transaction class.
- The richness of the dataset is also another aspect that limits the modelling method that can be applied as a detection solution. Most fraud detection methods detect misuse of credit cards than the behavioural trends due to absence of rich datasets.
- An ethical clearance will be required for this work if the dataset that is found to suit this work is sourced from an institution with actual real life customers. Customers may be exposed to identification and as such data anonymization techniques will have to be employed before such information is used in a public domain.

1.7 Delimitation

The fraud detection system to be modelled will be based on the misuse of credit cards. There isn't enough successive sample data to characterize the transactional behaviour correctly. The patterns will be extracted and generalised. No personalised behaviour per individual or credit card will be sought and modelled due to the limitation brought about by the dataset.

1.8 Thesis Overview

The research work was documented through a set of chapters and this section outlines and explains the purpose of each chapter and how they are structured.

Chapter 2 outlines the survey of the methodologies and techniques as presented in previous studies. Insights on sampling methods, feature engineering techniques will be highlighted, and recommendations explored as a foundation for this research work. All relevant machine learning models and feature creation methods that are available for use will be explained in detail. The methodology to build and compare the two machine learning models is described in Chapter 3. The dataset will be described, and its source outlined. The pre-processing techniques, sampling methods, evaluation metrics will be selected and discussed. Chapter 4 explores and prepares the dataset. The statistical characteristic of the features, their correlations and selection are described. The description and analysis of the features engineered are also presented. Chapter 5 will present the results of the models built and in-depth discussion will be presented analysing the evaluated performance metrics to determine the most robust credit card fraud detection system. In the last chapter, Chapter 6, the recommendations, future work suggestions and the conclusion of the research study are presented.

Chapter 2

Literature Review

2.1 Introduction

The prevalence and frequency of credit cards usage has created a lucrative business for fraud. Supervised and unsupervised algorithms have been build based on the availability and type of credit cards data [29]. Mainly supervised techniques have been employed as most datasets have targets. Supervised models such as artificial neural networks, gradient boosting machines, random forests have been employed in attempts to create the most optimal credit card fraud detection system as described in [1, 3, 8, 26, 28, 38]. Isolation forest, local outlier factor and Auto-Encoder are some of the unsupervised methods that has been used due to their ability to detect anomalies as detailed in [25, 27, 43, 46, 50]. Traditional statistical model and their variations have been used in other work that explored fraud detection.

In this chapter, a survey of feature creation and selection methods are explored for the pre-processing of the dataset in Section 2.2. Sampling methods are explored in Section 2.3. Section 2.4.1 discusses the traditional statistical methods used in detecting fraud over the years before the traction of machine learning techniques. Section 2.4.2 discusses machine learning algorithms that are applicable to the credit card fraud detection systems as options which the suitable model or models will be selected from. A selected number of research work that has focused on credit card fraud detection systems are explored in both Section 2.4.1 and 2.4.2. The survey aims to highlight some of the key findings and learnings that will form the basis of the fraud detection system that this work proposes. Supervised and unsupervised learning algorithms are discussed in Section 2.5. A summary is then presented where the theoretical advantages of the machine learning algorithms, traditional statistical methods and feature creation methods will be compared to some related work findings that will be highlighted.

2.2 Feature Creation and Selection Methods

A Forbes survey showed that data scientists spend 60% of their modelling time cleaning and organising data [33]. It is thus very critical to explore all applicable feature creation and engineering methods for effective modelling. The techniques are not equally effective across all machine learning

algorithms [33]. The developer needs to understand their dataset and machine learning techniques to be able to choose the best modelling approaches.

2.2.1 Imputation

Real data is not perfect. A lot of missing values are encountered in these datasets [45]. Data points are created by human beings and errors are prominent. Errors and privacy concerns are some of the factors that results in missing values[32]. Many models cannot handle missing values and these need to be remedied [32, 45]. Although some models can drop or ignore data points with missing values, this is at the detriment of performance due to the reduction of training data size [45]. Dropping may seem like the easiest approach but imputation(estimation) is a better alternative [35]. The methods of imputation that will be considered when preparing the dataset are:

- Numerical imputation: An approach where you replace missing values with a sensible average, median or mean (i.e. 0 for NA values where it's a binary class where other values are 1).
- Categorical imputation: This is column based technique. The mode might be the sensible estimate to impute or adding a category such as *Other* to the dataset.

2.2.2 Handling Outlier

Outliers can be handled when correctly identified. If the data can be visualised, this becomes the best and precise method to detect outliers [35]. The most popular approach however is the statistical method where one computes either the standard deviation and percentile then the outliers can be dropped. Although an alternative could be to cap the outlier's size so not to reduce the training set, the distribution of the dataset may be affected [35].

The data in-depth analysis for our work will include feature visualisation that will enable the application of the outlier handling technique.

2.2.3 Binning

Binning is a technique that aims to group data into classes based on numeric distance or hierarchy for the categorical dataset [35]. The model is made robust (less-overfitting) by binning but may result in information loss as data is regularised [16]. The trade-off between overfitting and performance is a key aspect that must be considered based on the type of machine learning model being used for numeric data. For categorical data, binning the less frequent labels tend to improve the robustness of the model. Binning is not very effective in continuous features as it results in significant loss of information [16]. Modern statisticians apply binning by incorporating the weight of evidence measure as a criterion [34]. This is the measure of strength of the predictors. It is defined as the log of Bayes factor, the distribution of one event against the other in a binary classification prediction problem. The effect of using weight of evidence was investigated and found that it is useful in improving the

predictive power of a model [16, 34]. It enables a linear modelling of non-linearities and reduces the effective categories in datasets that have a lot of categories [16].

The weight of evidence has been shown to be the most effective binning technique and will be employed in the preparation of the dataset for this work.

2.2.4 One Hot Encoding

The technique encodes the entries of a categorical attribute and transform them into dummy attributes constituted by the entries themselves [34, 37]. Each entry becomes a new feature represented by a binary 0 or 1. One attribute from the newly created dummy features can be dropped as it can be inferred from the others. One hot encoding enables the ability to group data without the loss of information [16].

The implementation of XGBoost is not compatible with categorical data and as such all categorical data will be encoded before modelling [32].

2.2.5 Grouping Operators and Feature Split

Tidy data is a concept that describe data whose features form a column, and each observation forms a row whilst each type of observational unit forms a table [52]. Messy data is that which whose rows and columns are values and not feature names. In such cases, data is grouped by instances and all instances are represented by one row (usually with the frequency counts). This technique extract hidden information that may be contained inside other features. This tidies up the dataset and improve performance than using the previously condensed feature [16, 35]. The performance is improved because the model can comprehend the feature better, as they can be binned precisely and extract more information.

The dataset will be cleaned by ensuring that there is no condensed feature that may contain multiple information in it. The values in each column will be resolved into its most basic format.

2.2.6 Extracting Date

Dates are key aspects in mapping the dataset to the labels. It is critical that dates are in formats that models can comprehend [16]. Date can be pre-processed to create separate columns of years, months, days, weekdays and holidays. The transformation creates a more explicit depth of information to map the targets.

2.3 Sampling methods

The solution to tackle highly skewed data is to sample the available entries. Random under and oversampling is the reduction or duplication of the majority class to match the minority class [17, 41]. The proportion of minority to majority is often so large that undersampling reduces more than 90% of the data [41]. The model we end up building losses so much richness of data to learn from. Undersampling is not considered for this work based on this fundamental issue.

Oversampling is the multiplying of minority class to match the majority class [17]. Sampling is done with replacement to create as many minority instances as possible to balance the classes. When instances are repeated, there is a huge risk of overfitting. The model learns a few insights a lot of times (reinforced that it becomes biased to the pattern). The traditional oversampling would merely duplicate entries and not add significant value in training the model [17, 41]. The frequently used sampling techniques are detailed in Section 2.3.1, 2.3.2 and 2.3.3 highlighting how they work and may be applicable to the task at hand.

2.3.1 Synthetic Minority Oversampling Technique (SMOTE)

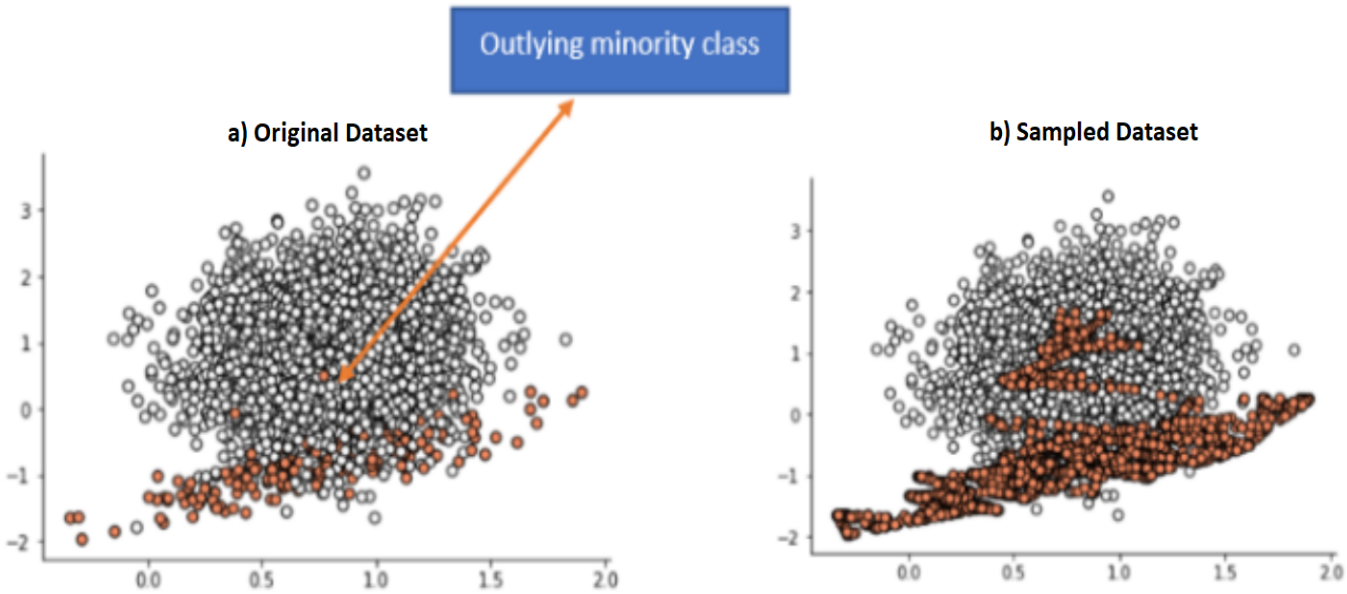

FIGURE 2.1: SMOTE data augmentation diagram [17].

SMOTE is an oversampling technique that creates synthetic instances based on existing ones, randomly [17]. The synthetic generation of data points will avail data points that mimics the minority class closely [17]. Assume there are these two points in a dataset (x_1, y_1) and (x_2, y_2), a random number $r \in 0, 1$ is picked to generate a synthetic point: $(x_1 + r(x_2 - x_1), y_1 + r(y_2 - y_1))$ [17]. The challenge with SMOTE is that outliers in the minority class that may appear in the majority class create a problem where they blur the boundary line between classes as evident in Figure 2.1(b). It can be seen that SMOTE technique may reduce performance due to duplicated minority instances that were outliers.

2.3.2 Borderline SMOTE

Bordeline SMOTE classifies instances into normal minority, borderline, noise and majority clusters [17, 41]. It then uses borderline instances to create synthetic duplicates of the minority instances improving the boundary between the majority and minority classes. The technique ignores the noise and normal minority instances and focuses only on the borderline instances [17]. The dataset is thus filled with extreme cases which is not ideal even when the boundary line is not blurred.

2.3.3 Adaptive Synthetic (Adasyn)

Adaptive Synthetic (Adasyn) is a sampling technique that generates examples based on the level of hardness of the examples to learn from [17]. Unlike SMOTE, the technique does not equally duplicate the minority samples but rather uses density distribution to assign weights to the data points before compensating the skewness of the dataset. It uses the impurity of the nearest objects by using the ratio of majority instances in the neighbourhood. The impurity ratio is converted to probabilities and that is used to determine the number of synthetic instances to generate. Adasyn is an improvement of SMOTE and borderline SMOTE in terms of generalisation of synthetic instances.

In summary, we see that synthentic sampling techniques are able to balance the imbalanced dataset satisfactorily. SMOTE however blur the boundary line between classes as evident in Figure 2.1 when it replicates minority classes that may appear in the majority class [17]. This impacts the model's performance. Bordeline SMOTE improves on the traditional SMOTE technique by ignoring the noise and normal minority instances and focusing only on the borderline instances [17]. This however fills the dataset with extreme cases which is not ideal despite not blurring boundary lines [17]. Adasyn was shown to be the superior of the three as it does not equally duplicate the minority samples. It uses density distribution to assign weights to the data points before compensating the skewness.

2.4 Fraud Detection Models

Fraud detection systems are a secondary measure, a second line of defence when a preventative method has failed to detect fraudulent transactions before they happen [5]. These models are thus detection techniques for when a suspected fraudulent activity has already occured [5]. Early detection techniques however still need to be applied to stop new fraudsters that may not know that certain techniques have already been identified and catered for.

Statistical methods are based on the comparison of observations and the expected patterns. The expected values are determined by numeric summaries and graphical representations that will expose outliers and behaviour profiles. These models can only point out the candidates for fraud for further investigation. It is usually very hard to conclusively determine a fraudulent transactions based on the use of a statistical models alone [21]. They output a suspicion score and flag an alert for manual

investigations. They merely infer relationships between features [5]. Section 2.4.1 explores these statistical methods and the related work that has applied them to give insights on the relevance of these models in the current era.

Machine learning is a form of Artificial Intellegence (AI) that allows iterative learning from data and is not restricted to rule-based programming [21]. The technology was developed in the 1950s but did not gain traction because of the computational complexity and the limitation of computer processors at the time. Processing power has drastically improved over the years and there is an abundance of data that has made machine learning to gain traction in the latest decade [21]. This technology enables repeatable predictions. It is result orientated and at times trades-off interpretability [21]. Most algorithms will obtain specious inference whilst making correct predictions. It is built on statistical frameworks and uses test dataset to validate its accuracy and the sanity of predictions. Section 2.4.2 outline algorithms that are applicable to fraud detection and the previous work for insights and recomendations to consider.

Machine learning cannot exist without statistics. Statistical methods alone however, are not strong on predictive power but are interpretable [21]. In statistical models, the mean square error between all data points is minimised and no training is required [26]. Statistical models do not need a test data for validation, only the robustness of the model parameter is tested. Regression parameters are tested through confidence and significance tests to evaluate the legitimacy of the model [21].

2.4.1 Traditional statistical models

2.4.1.1 Distance-based outlier

The distance-based outlier technique is a system that determines anomalies by using the neighbourhood approach [53]. It uses a threshold k neighbours to determine if an object is an outlier or inlier within the set distance threshold R.

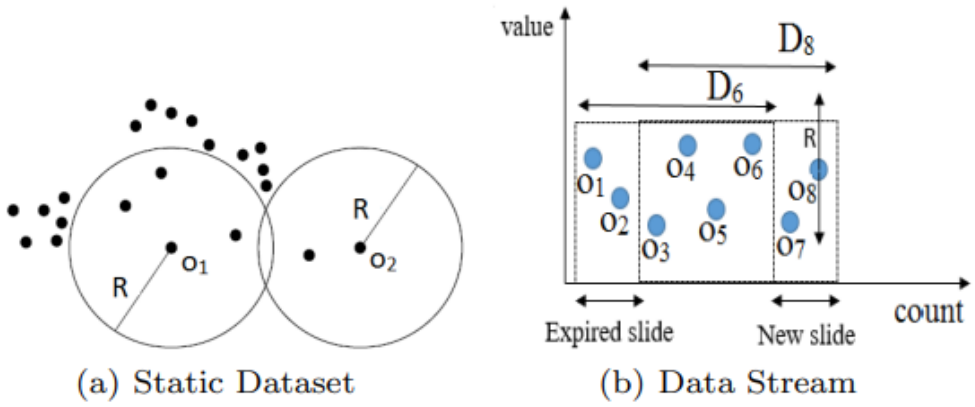

FIGURE 2.2: Distance-based outlier diagrams [49].

Given a dataset D, a neighbourhood neighbour count threshold $k(k > 0)$ and a distance threshold $R(R > 0)$. A distance-based outlier is any object that has less than k neighbours in D [49]. An object with at least k neighbours will thus be classified as an inlier. In Figure 2.2(a), if we set k=4, then both $o_1$and o_2 are outliers since they have 3 and 1 neighbour within R, respectively. Figure 2.2(b) shows how the method can be used in streaming data.

2.4.1.2 Peer group analysis

Peer group analysis is a model similar to anomaly detection methods as it does not need to build a normal behaviour profile for accounts [14]. The model does not require extensive training as a result. It determines a peer group for every object and find other accounts that behaved similarly over a period [14]. These groups are monitored (time evolution of the objects) and an anomalous behaviour is one that deviate strongly from the peer group it had been assigned [19].

Assuming we have determined peer groups for target accounts with size k:

- y_n is a continuous multivariate column vector.
- μ is the mean of the population $x_{\pi(1),n}, ..., x_{\pi(k),n}$
- C is the covariance matrix of $x_{\pi(1),n}, ..., x_{\pi(k),n}$
- The Mahalanobis distance (the distance in multivariate space) of a target object from its peer group:

$$\sqrt{(y_n - \mu)^T C^{-1} (y_n - \mu)} \tag{2.1}$$

- The object is flagged if the threshold set is exceeded.

To make robust peer groups, an object that has deviated strongly from the peers at time t must not contribute to the group at time t [19]. A heuristic can be used on the covariance matrix. A good quality peer group is one that follows the target closely over time (day s to day e).

$$Q_{s,e} = \frac{1}{t_e - t_s} \sum_{t=t_s}^{t_e} q_t \tag{2.2}$$

2.4.1.3 Outlier mining algorithm based on distance sum

Outlier mining algorithm based on distance sum algorithm differs from the original distance-based outlier detection as it determines the anomaly based on the sum of all distances [53].

If we let the universe of discourse $X = X_1, ..., X_n$ be the object to detect, the object would have indeces in a multivariable space such that:

$$X = X_{i1}, ..., X_{in} \qquad where \quad i = 1, 2..., n.$$

$$X = \begin{bmatrix} X_{11} & X_{12} & \dots & X_{1m} \\ X_{21} & X_{22} & \dots & X_{2m} \\ \vdots & \dots & \ddots & \vdots \\ X_{n1} & X_{n2} & \dots & X_{nm} \end{bmatrix}$$

The dispersion (distance) between any two object is given by d_{ij} [53]:

$$A = \begin{bmatrix} d_{11} & d_{12} & \dots & d_{1m} \\ d_{21} & d_{22} & \dots & d_{2m} \\ \vdots & \dots & \ddots & \vdots \\ d_{n1} & d_{n2} & \dots & d_{nm} \end{bmatrix}$$

Using the Euclidian distance method (any distance function can be chosen):

$$DIS(U, f) = \sqrt{\sum_{i=1}^{n} (o_i - f_i)^2} \tag{2.3}$$

Let:

$$P_i = \sum_{j=1}^{n} d_{ij} \qquad P_i \text{ is the } i^{th} \text{ row in matrix } A \tag{2.4}$$

The bigger the P_i, the bigger the distance between the i^{th} object and the others. P_i is thus a candidate for an outlier set.

$$\lambda_i = \frac{P_i - P_{min}}{P_{min}} \times 100\% \qquad \lambda = threshold,\ \lambda_i \geq \lambda \text{ is an outlier set.} \tag{2.5}$$

2.4.1.4 Outlier mining algorithm based on similar co-efficient sum

Outlier mining algorithm based on similar co-efficient sum is a variation of the outlier mining algorithm based on distance sum described above. It uses a co-efficient to make up the dispersion matrix [23]. The dispersion between any two objects is thus given by r_{ij}:

$$A = \begin{bmatrix} r_{11} & r_{12} & \dots & r_{1m} \\ r_{21} & r_{22} & \dots & r_{2m} \\ \vdots & \dots & \ddots & \vdots \\ r_{n1} & r_{n2} & \dots & r_{nm} \end{bmatrix}$$

Where similar co-efficients are calculated by:

$$r_{ij} = 1 - \sqrt{\frac{1}{n} \sum_{k=1}^{m} (x_{ik} - x_{jk})^2} \tag{2.6}$$

Let:

$$P_i = \sum_{j=1}^{n} r_{ij} \qquad P_i \text{ is the sum of the } i^{th} \text{ line in coefficient matrix } A \tag{2.7}$$

P_i is the smallest, furthest between object i and others [23]. P_i is thus a candidate for an outlier set.

$$\lambda_i = \frac{P_{max} - P_i}{P_{max}} \times 100\% \qquad \lambda = threshold,\ \lambda_i \geq \lambda\ is\ an\ outlier\ set. \tag{2.8}$$

An outlier mining with distance sum model was used to detect credit card fraud to improve predictive power of methods that would traditionally use standard distance-based method [53]. The method used standard deviation to standardise the transaction samples of the dataset. Commercial bank dataset was used with 16 584 data points and 15 135 being non-fraud. The consuming habits of clients were explored from the transactions. The highest accuracy measure achieved was 89.4% and the technique achieved accuracy range of 85% to 89.4% when the neighbour threshold k was varied from 8 to 12 [53]. The method is practiced on the basis of fraud being an infrequent and unconventional instance. A similar co-efficient sum technique was used to improve on the outlier mining with distance sum technique, and accuracy was improved to range from 89.5 to 91% for similar threshold ranges [23]. The method used similar coefficient sums to make up the dispersion matrix.

2.4.1.5 K-means clustering

Given n data points in a dataset B, and k the number of clusters desire, the partitioning algorithm into k clusters is given by 2.1 [2].

TABLE 2.1: K-means clustering algorithm.

Algorithm: K-means clustering algorithm
1: Determine number of clusters designed, n from the dataset, ($k < n$).
2: Select initial n centroids of the dataset.
3: Map each data point into the nearest cluster.
4: Update centroids.
4: Repeat step 3-4 until centroids are stable (do not change).

The method minimises the total squared errors within the cluster sum of square (variance). Given a dataset $S = S_1, S_2, ..., S_n$, the objective is to find:

$$\underset{\mathbf{S}}{\operatorname{argmin}} \sum_{i=1}^{k} \sum_{x \in S_i} ||x - \mu_i||^2 = \underset{\mathbf{S}}{\operatorname{argmin}} \sum_{i=1}^{k} \sum_{i=1} |S_i| Var S_i \tag{2.9}$$

Where μ_i is the mean of all the point in S_i. Equivalently, we could minimize equation 2.10:

$$\underset{\mathbf{S}}{\operatorname{argmin}} \sum_{i=1}^{k} \frac{1}{2|S_i|} \sum_{x,y \in S_i} ||x - y||^2 \tag{2.10}$$

2.4.1.6 K-medoids clustering

This method minimise the sum of dissimilarity between data points within the cluster and optimally choose the centre medoid (central objects) for each cluster [2].

TABLE 2.2: K-medoids clustering algorithm.

Algorithm : K-medoids clustering algorithm
Inputs: k - Number of clusters. B - Data with n objects
Output: A set of k clusters.
1: Arbitrarily choose k objects in B as initial centres or seeds.
2: **Repeat**
3: Assign each remaining object to the cluster with the nearest seed object.
4: Randomnly select an object not included in seeds ($\mathbf{o}_{random}$)
5: Compute the total cost, S, of swapping seed object, $\mathbf{o}_j$ with $\mathbf{o}_{random}$
6: If S<0 then swap $\mathbf{o}_j$ with $\mathbf{o}_{random}$ to form the new set of k representative(seeds)
7: **Until** no change;

In [2], a comparison of k-means and k-medoids models based on the Recency, Frequency and Monetary (RFM) score was done to improve customer segmentation and profiling. Internet banking transactions were used, and the clustering methods were compared. From the seven attributes in the data, only four were selected for the clustering (transaction date, account number, customer id, and amount of transaction). The model had six resultant clusters created. The k-means technique was better on both the Average Within Cluster (AWC), (3.045 compared to 7.777) and the Davies-Bouldin Index (DBI), (0.934 compared to 0.986) performance calculators. The most valuable cluster showed the clients with the highest frequency and monetary value (loyalty). The work recommended a combination of transactional, socio-demographic and product ownership data to improve the cluster quality [2].

The traditional statistical models described in Section 2.4.1 depend on the calculation of distances between objects. Distance-based outlier uses a neighbourhood approach [53] whilst peer group analysis determines a peer group for every object and find other accounts that behaved similarly over a period [14]. The outlier mining algorithm based on distance sum determines the anomaly based on the sum of all distances [53]. The outlier mining algorithm based on similar co-efficient uses a co-efficient to make up the dispersion matrix [23]. K-means partitions the data into k clusters and k-medoids minimises the sum of dissimilarity between data points within the cluster and optimally choose the central objects [2]. Our work will use a mixed data type dataset and as such, distance-based algorithms cannot be used.

2.4.2 Machine Learning models

In order for one to know which model is best suited for the work ahead, a deep understanding of the algorithms is needed. Algorithms from the most commonly used models are explored to discriminate them and explicitly class the theoretical advantages and disadvantages.

2.4.2.1 Logistic Regression (Logit)

Logistic regression is a statistical technique to estimate a two class or binary outcome based on a mapping of a score from a set of features. The outcome is achieved by a determination of the line of best fit on a non-linear sigmoid function [27].

The sigmoid activation function is defined as:

$$h_\theta(x) = g(\theta x) = g(z) = \frac{1}{1+e^{-z}} = \frac{1}{1+e^{-\theta^T X}} \tag{2.11}$$

x-input data and θ are the best fit co-efficient (the learnt optimizing from the cost function).

A classification is determined by the value of the output, y, where it is bounded between 0 and 1 and a threshold can be set to discriminate the two classes. The parameters are determined by a stochastic gradient descent method where the performance of the classifier is measured during the incremental step changes (as new data becomes available).

$$\theta_k \leftarrow \theta_k \alpha \frac{\partial}{\partial \theta_k} J(\theta)\ ,\ \forall k \tag{2.12}$$

$$z = x_0\theta^T x = x_0\theta_0 + x_1\theta_1 + x_2\theta_2 + ... + x_k\theta_k$$

The generalised cost function is defined as:

$$\begin{aligned} J(\theta) &= -\frac{1}{N}\left[\sum_{n=1}^{N} cost\left(h_\theta(x^{(n)}), y^{(n)}\right) + \frac{\lambda}{N}\sum_{k=1}^{d}\theta_k\right] \\ &= -\frac{1}{N}\left[\sum_{n=1}^{N} -y^{(n)} log\left(h_\theta(x^{(n)}\right) - \left(1-y^{(n)}\right) log\left(1-h_\theta(x^{(n)})\right) + \frac{\lambda}{2}\sum_{k=1}^{d}\theta_k^2\right] \end{aligned}$$

A logistic regression model was compared in [1] against the Naive Bayes, support vector machine and k-near neighbourhood techniques in the implementation of a credit card fraud detection. It was found that conventional methods are no longer as reliable due to the accelerating change in technology and the availability of big data [1]. The average amount of transactions per day, the amount, whether the transaction was declined or successful and the risk classification are some of the features in the thin data. Logistic regression was determined to be the best performing technique using four metric measures: accuracy, sensitivity, specificity and precision with an average of 96% [1]. The performance was restricted by the thin data on actual fraudulent activities. It was recommended that

a technique that can deal with a skewed dataset and that more data must be sought to improve the model [1]. They also recommended that fraud detection models must focus on card patterns rather than the owners as an owner may depict unique behaviours across multiple cards [1].

A different research team looked into the effects of hybrid sampling on a model's performance by comparing logistic regression model, KNN, Naive Bayes on a highly skewed dataset [3]. The performance metric measures were accuracy, sensitivity, specificity and Matthews correlation. PCA was used to reduce dimensionality, the data was purely numeric features [3]. There was an increase in performance when the ratio was changed from 1:9 to 34:66. In this study, KNN outperformed both the logistic regression and Naive Bayes models in the 34:66 sampled dataset and logistic regression had the lowest performance of the three models. Logistic regression model performed well on unsampled dataset. This was a consistent performance when the dataset was smaller, and the logistic model tends to overfit when the training data increases [3, 38]. They found that fraudsters mimic legitimate transactions, and this creates a dynamic profile for fraudulent behaviour which is hard to detect or model. They also found that the performance is influenced by the sampling method, features selected, the model, and the analysis of the card spending behaviour.

In the logistic regression work described above, we learn that fraud detection modelling must be focused on card patterns rather than the card owners as an owner may depict unique behaviours across multiple cards [1]. It is also worth the modeller's time to explore sampling methods, features selection, the models, and the analysis of the card spending behaviour as it impact the model's performance [3, 38].

2.4.2.2 Artificial Neural Networks (ANN)

The fundamental approach of the ANN technique is the ability to accept multiple complex inputs, aggregate them using the application of non-linear transfer function to produce a prediction. The most common ANN model is the feed forward multilayer perceptron's network [38].

$$\text{Neural model: } x_0\theta_0 + x_1\theta_1 + x_2\theta_2 + ... + x_k\theta_k \rightarrow h_\theta(x).$$

The sigmoid activation function is defined as:

$$h_\theta(x) = \frac{1}{1 + e^{-\theta^T X}} \tag{2.13}$$

A multilayer perceptron model comprises of an input, hidden layer and an output layer. Where the hidden layer can be as complex as required by the mapping. The parameters of the model are determined by minimizing the cost function $J(\theta)$.

$$\text{Parameters: } \hat{\theta} = \underset{\theta}{\text{argmin}}\ \text{J}(\theta)$$

This requires the computation of the partial derivative of the cost function with respect to the layer sequence in the network.

$$\frac{\partial}{\partial \theta_{j,i}^{(l)}} J(\theta)$$

Where l is the number of layers, i, j is the corresponding row and column of the multilayer of the network. The cost function $J(\theta)$ for the ANN model in its expanded form is:

$$J(\theta) = -\frac{1}{N}\left[\sum_{n=1}^{N}\sum_{k=1}^{K} y_k^{(n)} log\left(h_\theta(x^{(n)})_k\right) + \left(1 - y_k^{(n)}\right) log\left(1 - h_\theta(x^{(n)})_k\right)\right] + \frac{\lambda}{2N}\sum_{l=1}^{L-1}\sum_{i}^{n_l}\sum_{j=1}^{s_l+1}\left(\theta_{j,i}^{(l)}\right)^2 \tag{2.14}$$

The forward propagation model described in Figure 2.3 consists of m number of layers: Input/Output - Layer 1/Layer 4 and Layer 2 & 3 consisting of the hidden layer (where the designed method is constructed). The reverse process, backpropagation accounts for the error that may exist and propagates the error back to minimise it.

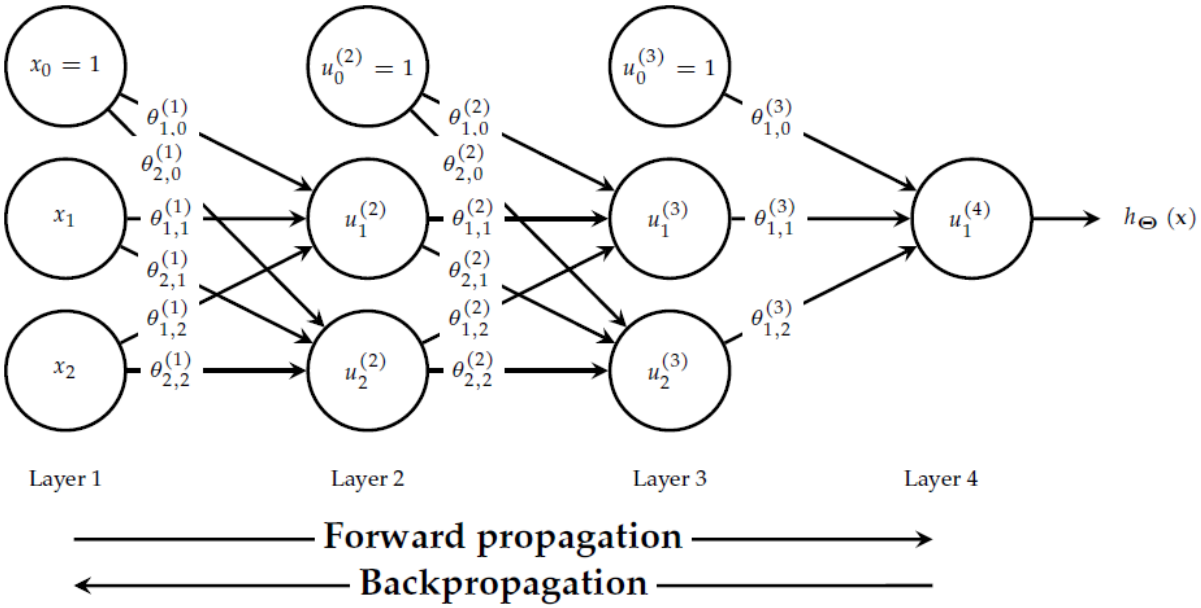

FIGURE 2.3: A representation of the artificial neural network model [8].

In work that compared ANN and logistic regression on selected descriptors [38], a suspicion score classification technique was used. A credit card was treated as unique entity and a customer can have multiple entities. It was detected that individuals that own multiple credit cards have distinct behaviours with each specific card [38]. Stratified samples were used to undersample the majority class. They used a technique which preserved the characteristic of features that show different distribution with respect to the target class. Preservation is achieved by using the said features as key features in the new dataset. The models were built on the IBM SPSS Clementine 12. The logistic regression models tend to overfits more and ANN out-performed logistic regression models in all test cases of the different models' variations and skewness levels.

ANN has been shown to outperform many traditional technique as in [38]. The definition of the technique however shows how complex the model builds its predictive network [8]. The bigger the network the harder it gets to explain what the model does inside the black-box.

2.4.2.3 Random Forest

Random Forest is an ensemble machine learning technique that uses multiple, independent random trees to make a prediction of the output class [48]. The class that is mostly predicted by these independent predictors becomes the result of the output predicted. The strength of the model is in building uncorrelated decision trees which minimises bias and overfitting [48]. The chances of making the correct prediction increases with the number of uncorrelated trees built. This method tackles the variance challenge that most tree-based algorithms suffers from. Decision trees use greedy algorithm, and this causes them to have a huge variance due to a method that heavily focuses on split node optimisation [6].

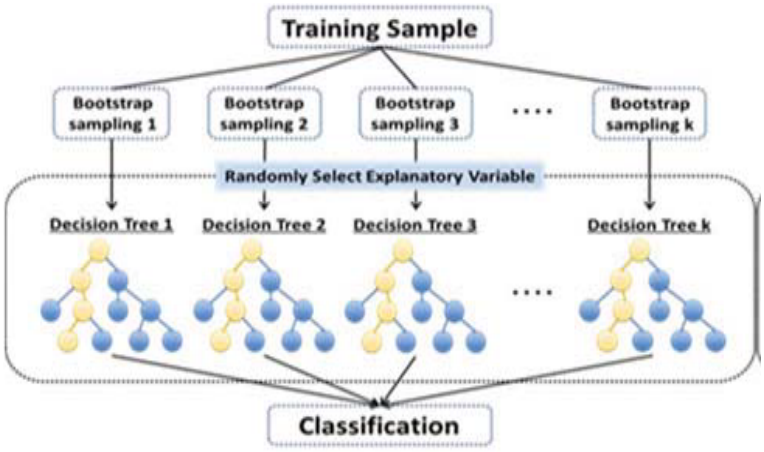

FIGURE 2.4: A representation of a random forest model [8].

To build random forest trees, the technique uses the bootstrap aggregation or bagging method and feature randomness. This ensures that there is no correlation between the predictions and thus, no error influence [56]. Bootstrapping makes more out of a small dataset; it acts as a multiplier of the dataset in how it samples with replacement during training. A diagrammatic representation is shown in Figure 2.4. The method takes advantage of scenarios where one cannot get more data. The dataset must still be large enough for the samples to be a good enough representation of the population. The algorithm is described in Table 2.3:

TABLE 2.3: Random forest algorithm.

Algorithm: Random forest algorithm
1: Create N bagged sample of size n from the dataset, (n<N).
2: Train the decision tree with each N bagged dataset. During node-split, don't explore all the features in the dataset. Randomly select M features, less than the total features and select the best split impurity.
3: Aggregate result into a single output

In [26], a random forest model was compared to a support vector machine, logistic regression, decision tree and various other ensemble models to improve the detection of credit card fraud. The

dataset was from a European credit card facility where only 492 of the 284807 entries were fraudulent. They found that the synthetic samples that are created by using synthetic minority over sampling technique may not be a true resemblance of the data regardless of the fact that the technique balances the data [26]. In their work they used random undersampling to achieve a 1:1 data distribution that does not change the original data. A 10-fold cross validation was used on both undersampled data and actual data. The classifiers performed above 90% for the undersampled data on both recall and precision whilst the random forest achieved 96% recall. The gradient boosted tree came second to the random forest model. They recommended the exploration of deep neural networks to improve accuracy in detection [26].

Another misuse method credit card fraud detection work compared Classification and Regression Tree (CART) algorithm and pure tree based random forest models [55]. The CART method employed the use of gini impurity index in the splitting criterion. This used random undersampling on the legitimate transactions. The work used the B2C real life dataset that had 30 million entries with labels across three months. The model was trained on a train-test ratio split of 70:30. The research work noted that anomaly detection created a normal behaviour profile database. The anomaly approach is not easily achievable as it needs enough successive sample data to characterize the transactional behaviour correctly [55]. CART based random forest model out-performed the tree-based model on the recall metric, 95% compared to 67%. The other metrics were highly comparable, and the two methods' performance could not be separated based on those.

The random forest work described above found that the synthetic sampling may not be a true resemblance of the data [26]. Sampling methods need to be fully investigated to improve the performance of models. Also, k-fold cross validation is a key feature of building a good model [26]. It was also found that the anomaly approach is not easily achievable as it needs enough successive sample data to characterize the transactional behaviour correctly.

2.4.2.4 Gradient Boosted Machines (GBM)

Many traditional data-driven models use a single, strong predictor to make theoritical classifications. A more recent approach employed the ensemble of different weak predictors [28]. The combinations of these weak models frequently result in a better aggregated model. The most common ensemble model is the random forest whose approach is to average multiple predictors. Boosting is an ensemble learning method that uses a sequential (stagewise additive modelling) combination of weak classifiers [9, 20]. Weak classifiers are those classifiers that are slightly better than a random guess. In boosting, predictors are created by randomly sampling the dataset with replacement over weighted data. The new tree learns from the errors of the previous predictor. Features which are hardest to classify appear most in subsequent predictors. The models are fit to the pseudo-residuals instead of being weighted. The diagram representing the model is shown in Figure 2.5 to demonstrate how trees are built and aggregated. To predict on new data, this technique takes a weighted summation of all predictors. Since the model focuses on difficult features, overfitting is highly likely and as such,

shallow trees (biased) are recommended [9].

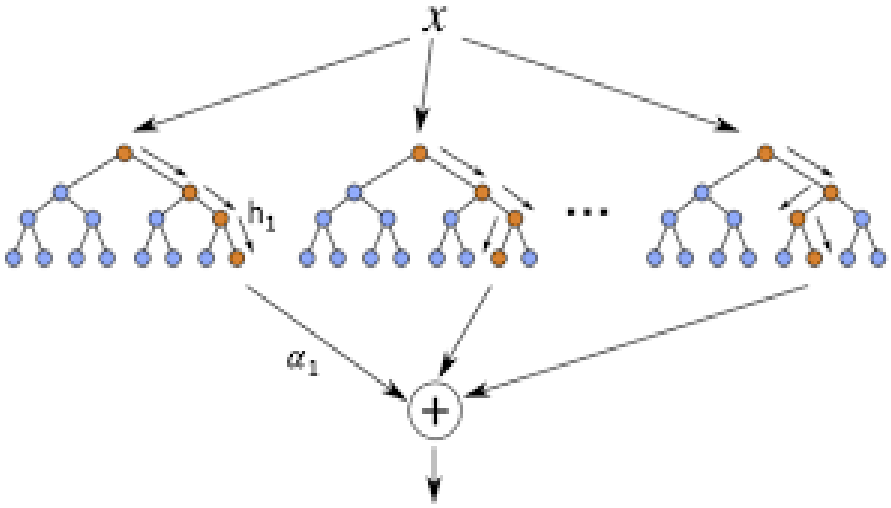

FIGURE 2.5: A representation of a gradient boosted tree model [31]. [x = dataset, h_1 = hypothesis and α_1 = weighted gradient result]

Given $(x, y)_{i=1}^{N}$ dataset, we aim to map $x \xrightarrow{f} y$ such that $\hat{f}(x)$ minimizes a loss function J(θ). Table 2.4 outlines the steepest decent algorithm.

TABLE 2.4: Gradient boosting machine steepest descent optimisation algorithm.

Algorithm: Gradient boosted machine steepest descent optimisation algorithm

Require: Optimised parameters $\hat{\theta}^t$
Ensure: $\hat{f}(x)$ minimizes a loss function $J(\theta)$
1: initialise $\hat{\theta}_0$, the parameters estimate
For each iteration k, repeat:
2: Obtain all $\hat{\theta}^t$ parameter estimate from the previous iteration
$\hat{\theta}^t = \sum_{i=1}^{t-1} \hat{\theta}_i$
3: Evaluate the gradient of the loss function $\bigtriangledown J(\theta)$
$\bigtriangledown J(\theta) = \{\bigtriangledown J(\theta_i)\} = \left[\frac{\delta J(\theta)}{\delta J(\theta_i)}\right]_{\theta=\hat{\theta}^t}$
4: Calculate the new incremental parameter estimate $\hat{\theta}_t$:
$\hat{\theta}_t \leftarrow \bigtriangledown J(\theta)$
5: Add the new estimate $\hat{\theta}_t$ to the ensemble.

In gradient boosting, a gradient descent algorithm is employed for minimizing the loss function. Gradient descent boosting machine were created to circumvent the challenges of the original boosting algorithm. In the improved approach, the newest base-learners are correlated to the negative gradient of the loss or cost function of the entire ensemble [28]. The choice of the loss function is

arbitrary and can be selected based on the type of data available and is customizable.

Tree boosting techniques achieve great results in ranking problems and they are the popular choice in practice amongst the ensemble methods [11]. GBM struggles with overfitting if not applied properly. Subsampling, shrinkage, and early stopping is used to reduce and limit the possibility of overfitting [11].

Features created from domain expertise and autoencoder were shown to improve the predictive power of a credit card detection system [37]. A gradient boosting machine model was being compared with the logistic regression and deep learning models by using the k-fold cross-validation's area under curve (AUC) to determine the algorithmic impact of the predictive power of a model. They found that boosted trees could handle missing value data and do not need extensive hyper-parameter tuning [37]. The GBT came second to deep learning model by a 1% margin. When extra features were created using domain expertise, the AUC values increased by 1-4% compared to 0.07% when an autoencoder method was used. RF model was better at predicting sugar cane yields than the GBT model even though the GBM model was still 20% more accurate than a human expert and projections based on the previous year's yield [10]. However, a recommendation was made on this work to study the effects of hyper-parameter, feature engineering and selection. In [47], they recommended using a system that ranked transaction based on their probability of being fraud instead of aiming to classifying them correctly to avoid the problem of imbalance. They further proposed a system that will be able to mine the dynamic and time-series data that depict the characteristics of a fraudulent transaction [47].

In GBM work, we learnt how the algorithm struggled with overfitting and subsampling, shrinkage, and early stopping is used to eliminate that effect [11]. Boosted trees can handle missing values and do not need extensive hyper-parameter tuning [37]. It is recommended to study the effects of hyper-parameter, feature engineering and selection [10]

2.4.2.5 Light GBM

GBM uses an entire dataset at each iteration for the estimation of information gain for all possible split nodes. This however is computationally costly and non-trivial as there is no sample weight in gradient boosting sampling [24]. Light GBM proposes an approach that circumvent this limitation by the introduction of a Gradient-based One-Sided Sampling (GOSS) and Exclusive Feature Bundling (EFB).

GOSS (outlined in Table 2.5) uses the fact that data instances have different gradients. Instances with large gradients contribute more to the information gain computation [36]. GOSS keeps these instances when it down samples the data and only randomly drop the instances with small gradient. The instances with large gradients are under-trained, the approach focuses on them while it down samples the well-trained instances with small gradients. This has shown to yield better results than

randomly sampling instances.

TABLE 2.5: Gradient-based one-sided sampling algorithm.

Algorithm: Gradient-based one-sided sampling
1: Sort instance by absolute value of gradient and select top $a \times 100\%$ instances.
2: Randomnly sample $b \times 100\%$ from the remainder.
3: Amplify sampled data with small gradient by $1 - a$ during information gain calculation.

Exclusive Feature Bundling is an NP-Hard problem and bundles unique features that would normally be sparse. Bundling reduces the effective features which reduces the feature bundling problem to a graph coloring problem. A greedy algorithm used to compute the solution to the problem with a constant approximation ration. Light GBM has been experimentally shown to perform twenty times better than traditional gradient boosting machines [36].

In [15], a light GBM is built on the IEEE-CIS dataset from the Vesta Corporation. They found that many traditional models do not have engineered features and they fail in the volatile production environment that deals with fraud detection. They removed anomalies in their data to improve the performance of their model. Compared to SVM, LR and XGBoost, the light GBM was shown to have superior strength. This was influenced by the engineered features and the employment of recursive feature elimination with cross validation (RFECV) to select important features. The model adopted optimization techniques such as GOSS and exclusive feature bundling to select features that have significant gradients to make estimations [15]. XGBoost had an area under curve performance of 94.2% compared to 95.5% of the light GBM built.

2.4.2.6 Extreme Gradient Boost (XGBoost)

XGBoost is the regularized and more robust improvement on the traditional gradient boosting machines. The algorithm uses parallel trees to learn, cross-validation is done at each iteration, trees are deep and pruned. With reduced complexity, XGBoost cuts down on the computational time. The overfitting backside of gradient boost is eliminated by regularization in XGBoost.

1. The algorithm and definition for the XGBoost is described below [54]:

   (a) The final prediction model for the XGBoost, when T trees are built, is defined as:

   $$\hat{y}_i = \sum_{t=1}^{T} f_t(x_i) \ , \ f_t \in F \tag{2.15}$$

   An ensemble of all trees f_t in the space of all possible tree F.

   (b) The objective function to minimize (optimize) is defined as:

   $$Obj(\theta) := L(\theta) + \Omega(\theta) \tag{2.16}$$

where: $L(\theta)$ - loss function and $\Omega(\theta)$ - regularization term that prevents overfitting.
In terms of the prediction model 2.15, the t_{th} objective function is:

$$Obj^{(t)} = \sum_{i=1}^{n} L(y_i, \hat{y}_i) + \sum_{t=1}^{T} \Omega(f_t) \tag{2.17}$$

By ensembling weak predictors, the loss term of the t_{th} tree based on the previous tree is:

$$Obj^{(t)} = \sum_{i=1}^{n} L(y_i, \hat{y_i}^{(t-1)} + f_t(x_i)) + \Omega(f_t) + constant$$

By the Taylor expansion, using the first and second order partial derivatives:

$$Obj^{(t)} = \sum_{i=1}^{n} \left[g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \right] + \Omega(f_t)$$

where: $g_i = \partial_{\hat{y_i}^{(t-1)}} L(y_i, \hat{y_i}^{(t-1)})$ and $h_i = \partial^2_{\hat{y_i}^{(t-1)}} L(y_i, \hat{y_i}^{(t-1)})$

(c) The regularization term is described from the decision tree:

$$f_t(x) = w_{q(x)} \ , \ w \in R^T \ , \ q : R^d \ \rightarrow \ 1, 2, ..., T \tag{2.18}$$

as:

$$\Omega(f) = \gamma T + \frac{1}{2} \lambda \sum_{j=1}^{T} w_j^2 \tag{2.19}$$

where: T - Total leaf nodes. $q(x)$ - function that maps data points to leaf node. w - records leaf node scores. γ and λ are parameters that control complexity of the model.

(d) The final expression for the objective function can thus be:

$$Obj^{(t)} = \sum_{j=1}^{T} \left[G_j w_j + \frac{1}{2} (H_j + \lambda) w_j^2 \right] + \gamma T \tag{2.20}$$

where: $G_j = \sum_{i \in I_j} g_i$ and $H_j = \sum_{i \in I_j} h_i$.
To get the minima, we set $I_j = 0$. Thus the leaf score becomes: $w_j^* = -\dfrac{G_j}{H_j + \lambda}$
$\therefore$ The optimal objective function is:

$$Obj^{*(t)} = -\frac{1}{2} \sum_{j=1}^{T} \frac{G_j^2}{H_j + \lambda} + \lambda T \tag{2.21}$$

(e) The rule for splitting trees can thus be define as:

$$Gain = \frac{1}{2} \left(\frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda} + \frac{(G_L + G_R)^2}{H_L + H_R + \lambda} \right) - \gamma. \tag{2.22}$$

where a tree branch is omitted if the gain after splitting is less than γ.

XGBoost model was found to be the best performing model in [29] in work that compared supervised and unsupervised machine learning models. The work noted the difficulty of accurately modelling credit card transactions due to data imbalance and dynamic patterns [29]. They used purely numeric dataset which was transformed by PCA due to confidentiality and sensitivity of the data variables. The performance were evaluated and compared using the area under the receiver operating curve (AUROC) as accuracy measure is not suitable in imbalance dataset [29]. The modellers used k-fold cross validation to reduce overfitting. XGBoost, a supervised model had an AUROC of 0.989 as compared to its closest unsupervised conterpart, Generative Adversarial Networks (GAN) at 0.953. The work concluded that supervised models performed slightly better than unsupervised models though unsupervised models are useful when there are no labels [29].

XGBoost was shown to be out-performed (only achieving AUROC of 0.942) by a light GBM whose optimization techniques such as gradient-based one-side and exclusive feature bundling to select features that have significant gradients to make estimations [15]. XGBoost however achieved AUROC of 0.989 using PCA transformed data [29].

2.4.2.7 Isolation Forest (iForest)

An iForest is an ensemble method that explicitly isolates anomalies in a dataset without the need to first profile the normal instances. The method builds an ensemble of isolation trees and anomalies are those instances that are isolated with the shortest path length. The technique's performance depends only on the number of trees and the sub-sampling size [25]. Setting the two features reasonably low yields the best performance and the method can handle datasets without any anomalies. Isolation forest is a linear time complex method as it does not need the large part of forest that profiles the normal instances. It does not use density nor distance measure techniques and the use of sub-sampling reduces the effect of masking and swamping [25]. Masking is the abundance of anomalies which enables anomalies to conceal their presence. Swamping is the wrongful identification of normal instances as anomalies [25].

iForest trees are created by the recursive partitioning of dataset instances until the instances are isolated or the set threshold tree height is achieved. The tree height is determined automatically by the relation that result from the chosen sub-sampling size ψ: $ceiling(log_2\psi)$ which is an equivalent of an average height. Only points that are shorter than the average height are more likely to be anomalies [25]. The only two features to set when building an iForest are the number of tree t and the sub-sampling size ψ. It has been shown that when the model detects reliably until a ceiling for the value of ψ = 256, after which no gains are achieved by increasing the sub-sampling size.

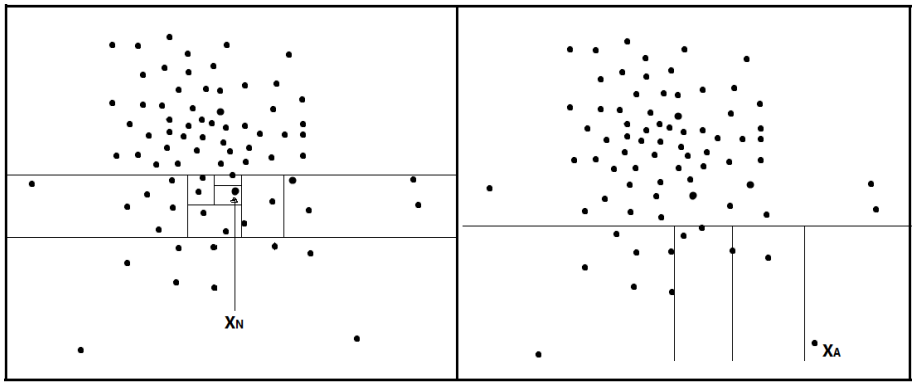

FIGURE 2.6: An isolation of a normal data point X_N and abnormal data point X_A [46].

Figure 2.6 shows a diagrammatic representation of how a normal instance (X_N) and an anomaly (X_A) are iteratively found. It can be seen that a normal instance needs multiple iterations to single out whilst the outlier is found in fewer iterations. Table 2.6 and 2.7 outlines the algorithms to build an isolation tree and forest respectively.

TABLE 2.6: Algorithm for building a tree for an iForest.

Algorithm : iForest - Tree building algorithm $iTree(X, e, l)$

Inputs: X - Input data, e - current tree height, l - height limit
Output: an iTree

if $e \geq l$ or $|X| \leq 1$ **then**
  return $exNodeSize \leftarrow |X|$
else
  let Q be a list of all attributes in X
  randomnly select an attribute $q \in Q$
  randomnly select a split point p from max and min values of attribute q in X
  $X_l \leftarrow filter(X, q < p)$
  $X_r \leftarrow filter(X, q \geq p)$
  return $inNode\{Left \leftarrow iTree(X_l, e+1, l)$
    $Right \leftarrow iTree(X_r, e+1, l)$,
    $SplitAtt \leftarrow q$,
    $SplitValue \leftarrow p$
end if

TABLE 2.7: Algorithm for ensembling the tree to build the iForest.

Algorithm : iForest - Forest Ensemble $iForest(X, t, \psi)$

Inputs: X - Input data, t - number of trees, ψ - sub-sampling size
Output: a set of t iTree

Initialise Forest
set height limit $l = ceiling(log_2 \psi)$
for $i = 1$ to t **do**
  $X' \leftarrow sample(X, \psi)$
  $Forest \leftarrow Forest \cup iForest(X', 0, l)$
end for
return Forest

iForest was shown to be effective in detecting credit card fraud by assigning a degree of probability in which the transaction exhibits abnormal characteristics [50]. The study used a binary and multi-classifier to detect fraudulent transactions and to interpret the type of anomalies that each transaction exhibits. A random forest was used to build the interpreter model to complete the hybrid system proposed. The work was on a no-sql elastic search database with 41 features to detect the anomalies in the geo-location of users during transactions. 300 and 500 trees were built for the isolation forest and the random forest, respectively. The hybrid system had an accuracy and specificity of 90%.

An iForest technique for anomaly detection is proposed in [25], and isolates abnormal instances without first profiling normal instances. The technique is based on the concept that abnormal instances are different and few in any large dataset. It was shown that setting the number of trees at a hundred and sub-sampling size at 256 yields the best computational and efficient detection system. The path length converges well at these settings. The system showed its ability in reducing masking and swamping [25]. It achieved an AUROC of 0.91 when sub-sampling was used compared to 0.67 when the entire sample was used. A comparable performance was achieved when the model was trained without any anomalies by a slight increase in the number of trees and sub-sampling size.

In [46], an iForest model is used to detect false data injected on the power system network. The technique is used on the basis that the false data is located sparsely and can be separated quickly from the true data. The anomalies are ranked using the path length and the anomaly score. The performance achieved in this work supports what was found in [25] for the settings of the number of trees and sub-sampling size. They experimented with several trees against sub-sampling size curve and concluded that the increase in performance is insignificant compared to the increase in computational time when the threshold settings are increased. This work achieved an AUROC of 0.963, which was slightly higher than that of the local outlier factor technique. The local outlier factor technique has a computational time of 25 seconds compared to 1.9 seconds for the isolation forest. The adaptiveness of the isolation forest in boosted by the fact that it does not need pre-training procedures [46].

The study in [43] employs the local outlier factor and the iForest technique in its work to improve the cost and security of the credit card transactions. An iForest model is used to separate the anomalies from the normal data and the local outlier factor scores the anomalies and determine their authenticity [43]. The local outlier factor technique calculates the deviation with respect to the data point neighbours instead of the global sample. The iForest randomly separates the data attentively and find instances such that those with the shortest path length are flagged as possible anomalies. A forest is built, then the isolation trees and and lastly an evaluation is made. The anomalies were determined by the analysis of the following features, amount, transaction limit, average transaction, merchant vendor, location, country, time of transaction and issuing bank.

A supervised and unsupervised learning survey in [18] compared iForest, random forest, k-means,

local outlier factor, logit and SVM using credit card fraud data. iForest had the highest accuracy at 0.9977 but the second worst f1 score, recall, and precision at 0.67 for all these measures. SVM, random forest, k-means and logit had f1 score, recall, and precision of at least 0.92 for the three measures. Local outlier factor had 0.25 for the three measure and was the worst performer. This work showed that accuracy is a misleading measure and unsupervised techniques are still behind supervised models in performance when labelled data is available [18].

The iForest work provided lessons on the settings of the number of trees and sub-sampling size for optimal performance [25, 46]. iForest performs slightly higher than the local outlier factor technique on predictive strength and computational time [25]. Accuracy was shown to be a misleading measure and that unsupervised techniques are still behind supervised models in performance when labelled data is available [18].

2.5 Supervised vs. Unsupervised learning

2.5.1 Supervised learning

Supervised algorithms are models that are mainly used in both classification and regression problems [5]. Labels are provided and the construction of the model is based on the learnt patterns and assumptions deduced. The confidence of the prediction is based on accuracy and strength of the learnt truth (training data). The disadvantage of supervised learning is that most dataset have unbalanced class size and misclassifications, these causes skewing and degrades the model's performance [51, 57]. The models include logit, ANN, random forest etc. The model construction considers bias-variance trade off and model complexity. Highly complex models tend to overfit, not generalise well [27].

A comparative study in [29] evaluated supervised and unsupervised algorithms using numeric data transformed using PCA due to confidentiality issues. They were able to show that imbalance data degenerate the performance of supervised models. A sampling technique must be used to eliminate the data skewness bottleneck [29]. Supervised models perfomed slightly better than unsupervised models, AUROC of 0.989 for XGBoost compared to 0.960 for Generative Adversarial Networks (GAN).

2.5.2 Unsupervised learning

Unsupervised algorithms do not need training data to be able to learn patterns. They are used in clustering, exploratory analysis and dimensionality reduction [5]. They explore datasets to isolate the most dissimilar patterns from the normal behaviours in the data [29]. The isolated entries can be further analysed with outliers being the non-standard observations [1, 5]. Unsupervised learning allows models to extract patterns that have not been explored yet as they are not restricted by a training data. They learn the inherent structure and pattern of the dataset without classification

nor labels. Most unsupervised learning methods' performance cannot be compared when working on unlabelled data. Unsupervised models are optimal in exploration methods as they discover structures in data automatically [1, 18]. The comparative study in [29] showed that unsupervised algorithms were still worth exploring as they achieved AUROC above 0.90 for the four unsupervised models, the Restricted Boltzmann Machine (RBM), Generative Adversarial Networks (GAN), One-Class SVM (OCSVM) and Auto-Encoder (AE).

Comparative work done to evaluate supervised and unsupervised models showed that there is still a gap in the knowledge area (which our work aims to address). The studies in [18, 29] used numeric variables and there exist an opportunity to investigate the results of the comparison when mixed data type or categorical datasets are used.

2.6 Summary

The survey of the methodologies and techniques has formed a strong foundation for this research work based on insight gathered. It was discovered that feature creation and transformation techniques do not serve as magic tools, the dataset still need to have usable information for the models to be effective. The data in-depth analysis for our work will include feature visualisation that will enable the application of the outlier handling technique as outlined in [35]. Given that the implementation of the models in our proposed work are not compatible with categorical data, all categorical data will be encoded before modelling [32]. The values in each column will be resolved into its most basic format. This tidies up the dataset and improve performance than using the previously condensed feature [16, 35]. Dates are key aspects of the dataset and it is critical that they are in formats that the models can comprehend [16].

Adasyn is an improvement of SMOTE and borderline SMOTE in terms of generalisation of synthetic instances [17]. Using density distribution to assign weights to the data points before compensating the skewness based on impurity ratios [17].

The traditional models described above depend on the calculation of distances between objects. A neighbourhood approach, peer groups, sum of distances, similar-coefficients and clustering aprroaches are detailed in [2, 14, 23, 53]. Our work will use mixed data type dataset and as such, distance-based algorithms cannot be used.

Models such as GBM and XGBoost are able to handle missing values. Logit algorithms predict better in thin datasets and tends to scale badly with big data [15, 37]. Complex models like deep learning performs better than boosting machines at the expense of explainability [29]. Hybrid models are becoming popular and work well when the data is fed in real time during production. Feature engineering, selection and hyper-parameter tuning is amongst the most important aspect of building the fraud detection models [10, 37]. It was shown that domain-expertise created features improve the

predictive power of a model much better than those created by autoencoders by a scale greater than ten.

iForest work provided lessons on the settings of the number of trees and sub-sampling size for optimal performance [25, 46]. The biggest challenge remains the ability to extract dynamic and time-series patterns to depict the true characteristics of fraudulent transactions [25]. They also found that the performance is influenced by the sampling size, features selected, the model, and the analysis of the card spending behaviour. iForest does not need pre-training procedures. Accuracy was shown to be a misleading measure and that unsupervised techniques are still behind supervised models in performance when labelled data is available [18].

It was learnt that supervised learning models are usually restricted by the types of instance patterns that have been seen in the data before. It is critical that this training data is reliable for effective learning and the success of future predictions. The disadvantage of supervised learning is that most datasets have unbalanced class size and misclassification [51, 57]. Unsupervised learning allows models to extract patterns that have not been explored yet as they do not need training [29]. They are thus not restricted by a historic training data. Unsupervised models are optimal in exploration methods as they discover structures in data automatically [18, 29]. Comparative work done to evaluate supervised and unsupervised models show that there is still an opportunity to investigate how the results of the comparison when mixed data type or categorical datasets are used instead of numeric variables.

Chapter 3

Research Methodology

3.1 Introduction

In the previous chapter, the survey of the methodologies and techniques that formed a foundation for this research work were presented. The traditional and machine learning models were described and previous work that employed these algorithms presented. The differences between supervised and unsupervised algorithms were also highlighted.

In this chapter, the research framework is presented in Section 3.2 and the source of the data is outlined in Section 3.3. The techniques that will be used to prepare the data features will be discussed in Section 3.4. Models will be chosen and presented in Section 3.5 and lastly, the evaluation metrics will be discussed in Section 3.6.

3.2 Research Framework

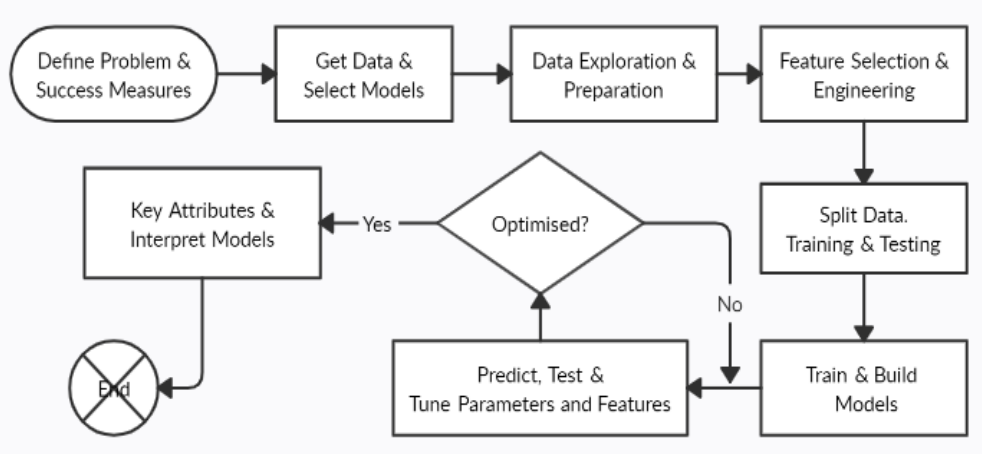

FIGURE 3.1: Model development process.

The framework that this work will follow is outlined in Figure 3.1 and shows the individual components in the sequence they will be tackled. The problem has been defined and the success measures have been explicitly stated in Chapter 1. In this chapter, the data source is outlined, the technical methods and algorithms are then detailed. The data exploration and feature engineering is presented

in Chapter 4. The iterative process of achieving an optimal fraud detection system is highlighted where parameters and features will be tuned. The model will then be interpreted by discussing the predictive features and their strength.

The credit card fraud model process is shown in Figure 3.2. The diagram shows where the model to be built will fit in the fraud detection process at a financial institution. When an input transaction is received, this will trigger a matching algorithm that will scan the legal and fraudulent pattern database to correctly classify the incoming request. A score will be assigned to the transaction and the transaction will be classified into an allowable or fraudulent request. The outcome will be fed back to the algorithm as a new example in the database. The model will continuously learn whilst in production. For a fraudulent alert, the financial institution will investigate the transaction and grade it accordingly. A customer may also notify the financial institution in an event where they have been a victim of impersonation and the investigation team will grade the affected transactions appropriately.

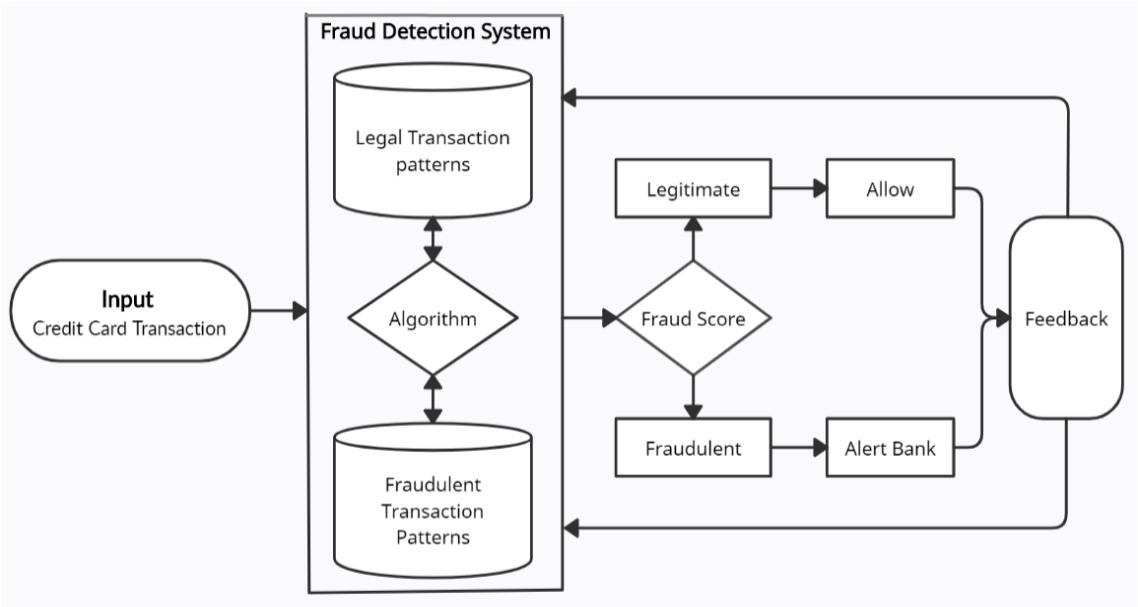

FIGURE 3.2: Credit card fraud detection process.

3.3 Data Sources

The dataset chosen for this work was downloaded from Kaggle and it is a simulated credit card transaction dataset that contains fraud and legitimate transactions of 999 customers across 693 merchants. The dataset is available in two sets, the training set (1 296 675 data points) and testing set (555 719 data points). The data was generated using Sparkov Data Generation and 99,42% of the transactions were non-fraudulent.

- Geographic features - There are 10 features that gives information on the merchant name, its location, the credit card owner's address and the population size in their neighbourhood.

- Demographic features - There are 5 features that gives insights on the credit card owner's name, last name, age, gender and the type of work they do.
- Transactional features - There are 5 features that gives information on the transaction amount, date, time, credit card number and unique transaction codes.

3.4 Data Pre-processing and Feature Selection

The correct selection of attributes eliminates overfitting and the curse of dimensionality in models. A feature correlation matrix will be built to determine the relationship between attributes using the correlation equation 3.1 where, the covariance is weighted by the product of the individual variances [9].

$$r = \frac{\sigma_{xy}}{\sigma_x \sigma_y} \tag{3.1}$$

The covariance, a measure of the joint variability is defined in 3.2:

$$\sigma_{xy} = \frac{\sum_{n=1}^{N}(x_i - \mu_x)(y_i - \mu_y)}{N} \tag{3.2}$$

A pairwise distribution analysis of the data will also be done to analyse the data visually. The features that were not considered for modelling and the qualifications are presented below:

- Unix_time - The time of the transaction can be extracted from trans_date_trans_time.
- Trans_num - This feature was dropped as it just identified the transaction uniquely, no general pattern could be mined from such a feature.
- Street - This feature will not add any value to the model since coordinates are provided.
- First and last - The features were not considered for this experimentation, only the general pattern from all transactions were considered without individualised database.

Missing feature values create a problem in modelling by reducing the sample size and statistical models assume data completeness [45]. The data will therefore be remedied by padding the missing entries with the mode, the most occurring value in that feature. XGBoost is equipped to handle missing values in its internal modelling process [32].

XGBoost is not compatible with categorical data [32]. These categorical features will be transformed using one hot encoding technique. This technique encodes the entries of a categorical attribute and transform them into dummy attributes constituted by the entries themselves. Each entry becomes a new feature represented by a binary 0 or 1. One attribute from the newly created dummy features can be dropped as it can be inferred from the others.

The features described below were split into meaningful parts and other used to extract other explicit fields:

- trans_date_trans_time
  - This feature was transformed to create quarter, weekday, week number, month-day (MMDD), day of month (DD), and 24 hour ranges.
- long, lat, merch_long, merch_lat
  - these features were used to calculate the harvesine distance between the customer's home and the merchant they transacted at, in kilometres.
- dob - the feature was used to calculate the customer's age at the time of the transaction.

All categorical features were converted into dummies using one hot encoding and most resulted in more than 50 unique categories. Binning was used to reduce the effective features that will be fed into the model. Continuous features were not binned to prevent the unintended loss of valuable information.

Fine classing is a well-established process to convert discrete and continuous features into categorical dummies based on their properties [42]. These initial categories rarely matter as we would bundle them further (coarse classing). It is critical however to be able to convert these arbitrary categories to usable dummies. This is achieved by computing or determine the ability of each category to predict the dependent feature (label). The established method for determining the rates of good and bad class across categories is the weight of evidence [16].

Weight of evidence is the extent at which an independent feature is able to predict a dependent feature (target label). It quantifies how well a predictive feature explains the target label [16, 42]. This is computed by the natural logarithm of the ratio of the proportion of observations of the first type of outcome of the dependent feature that fall into the respective category of the independent feature.

$$WoE_i = \left(\frac{\%(y=1)_i}{\%(y=0)_i}\right) \tag{3.3}$$

$$= \left(\frac{\%good_i}{\%bad_i}\right) \tag{3.4}$$

$$= \left(\frac{\%non\_fraudulent_i}{\%fraudulent_i}\right) \tag{3.5}$$

The natural logarithm of the ratio of the proportion of observations of the first type of outcome of the dependent feature that fall into the respective category of the independent feature.

Marital status has three categories, married, single and divorced. In Table 3.1 there are 4 600 clients who are single, 4 000 are good whilst 600 are bad. Of the 15 400 clients who are married, 12 000 are

TABLE 3.1: Example: Weight of evidence calculation for the Marital Status category on the Label.

Feature Categories	Number of Good	Number of Bad	Proportion Good	Proportion Bad	Weight of Evidence
Single	4 000	600	21.05%	12%	0.56
Married	12 000	3 400	63.16%	68%	-0.07
Divorced	3 000	1 000	15.79%	20%	-0.2
	19 000	5 000	100%	100%	

good whilst 3 400 are bad. Of the 4 000 clients who are divorced, 3 000 are good whilst 1 000 are bad. Using the weight of evidence computation, the categories single, married and divorced have 0.56, -0.07 and -0.2 respectively as shown in Figure 3.3. The further away from zero a category is, the better it is at differentiating between the two-class label. The category, *single*, explains the labels better than the other two categories (by a magnitude of five).

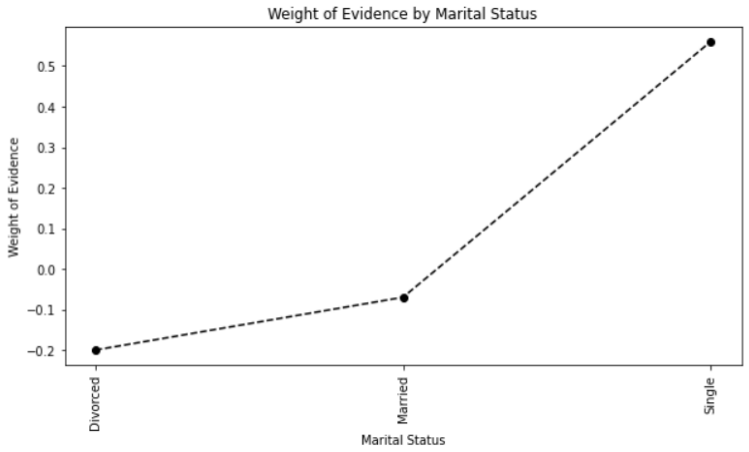

FIGURE 3.3: Weight of Evidence by Marital Status on the Label.

These calculations help us to do the final classing, coarse classing. Coarse classing is the notion of classing new categories from the initial ones based on their weight of evidence [16, 42]. We combine these into bigger categories based on similarities of weight of evidence. This approach of bundling lowers the number of dummies and improve the predictive power of a model. In the example, we may chose to combine categories *divorced* and *married* together as they has similar weight of evidence. This will reduce marital status to only have two categories, *single* and *divorced_married*. When we use one hot encoding, we will then drop the category with low weight of evidence. For our example, marital status will only contribute the category, *single*, into the model builder as *divorced_married* will be dropped.

3.5 Model Definition and Algorithms

XGBoost, a supervised algorithm was chosen for this work as it has proven its worth over the years; it is simple and trivial yet able to model complex interactions and still yield interpretable results. It has also formed part of winning solutions in Kaggle competitions over the years because of its scalability and algorithmic optimization [11].

The model will be validated using the k-fold cross validation with k=5 to eliminate overfitting [29]. This is a simple well-established method that is used to predict the test error accurately [9]. The method iteratively trains the model with $k-1$ partitions of the dataset and the remainder to test it. This is done k times whilst changing the testing partition each time [9].

The grid search method wass employed to tune and find the optimal parameters to use for building the models. Insights from previous models as described in 2.4.2 informed this work on the array of settings to feed onto the search to limit the possible values and combinations to choose from.

An iForest model, a unsupervised algorithm was also chosen in this work because it is a linear time complex ensemble method that explicitly isolates anomalies without profiling the normal instances first [25, 54]. This allows the research to compare two models that use ensemble technique and two opposite training methods, supervised and unsupervised.

Adasyn and SMOTE sampling techniques were compared in the building of the models to address the unbalanced data classes. The traditional under sampling technique will not be considered as this approach reduces the dataset significantly and over sampling would merely duplicate entries and not add significant value in training the model [17]. The synthetic generation of data points will avail data points that mimics the minority class closely.

The credit card fraud models was built using Python 3 on a machine that has the following specifications: Intel(R) Core(TM) i7-9750H CPU@2.60 GHz and 16.00 GB RAM, and windows 64-bit operating system.

3.6 Evaluation Metrics

The metrics that evaluated the performance of the designed models are listed with their corresponding mathematical formulae in Table 3.2.

The format and structure of the confusion matrix to be used in the analysis is shown in Table 3.3. A confusion matrix is the summary of the results of a classification problem. The number of correct and incorrect predictions are categorised by each class. It shows the level of confusion of the model when making predictions. The confusion matrix enabled the derivation of the performance measures

described in Table 3.2. As a final measure, a comparison against similar work in this knowledge area was be done to identify a position in the research field where the system developed fits.

TABLE 3.2: Table of formulae for evaluation metrics.

Metric	Formulae
Recall	$\frac{True\ Positive}{True\ Positive + False\ Negative}$
Precision	$\frac{True\ Positive}{True\ Positive + False\ Positive}$
Cohen's Kappa	$\frac{Observed\ Accuracy - Expected\ Accuracy}{1 - Expected\ Accuracy}$
Gini Coefficient	2AUROC - 1
Accuracy	$\frac{Number\ of\ features\ correctly\ classified}{Number\ of\ features}$
F_1-score	$2\left(\frac{Precision\ \times\ Recall}{Precision + Recall}\right)$

TABLE 3.3: Format of the confusion matrix.

	PREDICTED NO	PREDICTED YES
ACTUAL NO	True Negative	False Positive
ACTUAL YES	False Negative	True Positive

Where:

- True Positive - A prediction is positive when the observation is positive.
- False Positive - A prediction is positive and the observation is negative.
- True Negative - A prediction is negative and the observation is negative.
- False Negative - A prediction is negative and the observation is positive.

The accuracy of the model is a measure of how well the model predicts every class of the problem. The accuracy is optimal in cases where the classes are of equal importance. The accuracy measure is not a sufficient measure when the strength of predicting true positives is of great interest than merely the error rate [22]. The accuracy is only reliable when the classes are balanced as a highly imbalanced dataset can achieve good accuracy by always predicting the dominant class [22].

The precision measures what percentage of the target classifications are correct. The ratio between the true positive classifications and all positive classifications. In the context of our research study, the measure of transactions that were correctly classified as fraudulent out of all fraudulent transactions.

F_1-score is the harmonic mean of the precision and recall of the model. It is formulated by getting the reciprocal of the arithmetic mean of the reciprocals of the given observations.
$F_1 = \frac{2}{recall^{-1} + precision^{-1}}$. The f_1-score is a a more reliable measure when the data is unbalanced [7].

The Receiver Operating Characteristic (ROC) curve - This is a two dimensional graph that summarizes all the possible confusion matrices with varying classification threshold [12]. The graph plots the true positive rate (sensitivity) on the y-axis and the false positive rate ($1 - specificity$) on the x-axis. The graph that cuts the points {0,0} and {1,1}, diagonal line represent a ROC that is as good as a random guess whilst the prediction gets better as the curve approaches point {0,1}.

The Area Under the Receiver Operating Characteristic Curve (AUROC) is a measure that evaluates which ROC is better than the other by calculating the total area under a graph.

The Kolmogorov Smirnov (KS) statistics is the measure of the maximum difference between the true positive rate and the false positive rate (target and not target) in statistical models [22]. It validates the predictive power of models. A higher KS statistic value indicates great separation power of the binary classes. The separation would be read vertically on the plot (y-axis). On the x-axis, threshold axis of the plot shows the data percentage threshold. This is the percentage of the data that is predicted with the greatest separation power.

Cohen's kappa is the metric that evaluates the agreement of two raters [4]. In the context of the fraudulent transaction classification, we would be comparing the predictions of the models built against the target labels. The Cohen's kappa is also calculated from the confusion matrix whilst taking imbalance in the class distribution when calculating accuracy. The mathematical definition is thus:

$$\kappa = \frac{\rho_0 - \rho_e}{1 - \rho_e} \tag{3.6}$$

Where:
ρ_0 is the overall accuracy and ρ_e is the measure of agreement.

The gini coefficient is the measure of the discriminatory power of the model to predict a fraudulent transaction from a legitimate one. It is the measure of how the model exceeds random predictions. It is thus evaluated by calculating the area between a random line and the model curve.

3.7 Summary

The methodology to build and compare XGBoost and an iForest credit card fraud detection models is described. The objective of the work is to determine features that have the greatest impact in detecting fraudulent credit card transactions, to determine optimal aggregates and ratios that improve

model's performance; and to determine which model between the XGBoost machine and iForest models has superior fraud prediction strength. The dataset used is sourced from Kaggle and it is a simulated credit card transaction dataset from Sparkov Data Generation. The dataset contains fraud and legitimate transactions of 999 customers across 693 merchants. The dataset is available in two sets, the training set (1 296 675 data points) and testing set (555 719 data points) and 99,42% of the transactions being non-fraudulent. Imputing was used to handle missing values and one hot encoding to transform unusable categorical features into useful ones. The feature split and extract were used to convert features that had multi-dimensional insights. The binning technique used was based on the weight of evidence as it quantified how well a predictive feature explained the target. Adasyn and SMOTE sampling techniques were compared in the building of the models. The model is built on Python 3 and evaluated by the well-established methods; precision, recall, f1-scores that are based on the confusion matrix. The other statistical evaluation metrics were Cohen's kappa, gini coefficient, KS statistics and the AUROC measures.

Chapter 4

Exploratory Data Analysis and Feature Engineering

4.1 Introduction

In the previous chapter, the supervised and unsupervised models to be compared for our work were listed and justified. The model building and data preparation techniques were described. The source of the data was outlined, the overview of the data features and label were also described. The evaluation metrics are defined and softwares declared.

In this chapter, the exploration and preparation of the dataset is outlined. The statistical characteristic of the features, their correlations and selection are described in Section 4.2. The description and analysis of the features engineered is presented in Section 4.2.3.

4.2 Statistical Characteristics of the Dataset

4.2.1 Original Dataset

There are 1 852 394 instances in the dataset, with 70% of that dedicated for the training set. In the training set, only 0.58% are fraudulent transactions compared to 0.387% in the testing dataset. The dataset is highly skewed towards the majority class.

TABLE 4.1: Data description of the continuous numeric features.

	Count	Mean	Standard Deviation	Min	25%	Median	75%	Max
Amount	1 852 394	70	159	1	9.64	47.4	83.1	28 948
City population	1 852 394	88 643	301 487	23	741	2 443	20 328	2 906 700

The dataset has only two continuous numeric features, the amount (*amt*) spent on a transaction and the city's population (*city_pop*). The two continuous features showed that the features were not centred around the mean as depicted by the high standard deviation values in Table 4.1. City population had the highest variations, showing that these transactions occurred in various cities with different

populations.

Multicollinearity is a dataset property on which a magnitude change of one variable affects the other's magnitude in the same or opposite direction (positive and negative correlation) [39]. Features that are correlated causes model instability, overfitting and difficulties in interpretation [39]. The correlation matrix was interpreted by Table 4.2. One of the pair on features that are strongly correlated (≥ 0.7) must be dropped to eliminate overfitting and model instabilities.

TABLE 4.2: Correlation Coefficient Interpretation.

Magnitude of Correlation Coefficient	Interpretation Deviation
0.0 - 0.10	Negligible correlation
0.10 - 0.39	Weak correlation
0.40 - 0.69	Moderate correlation
0.70 - 0.89	Strong correlation
0.90 - 1.00	Very strong correlation

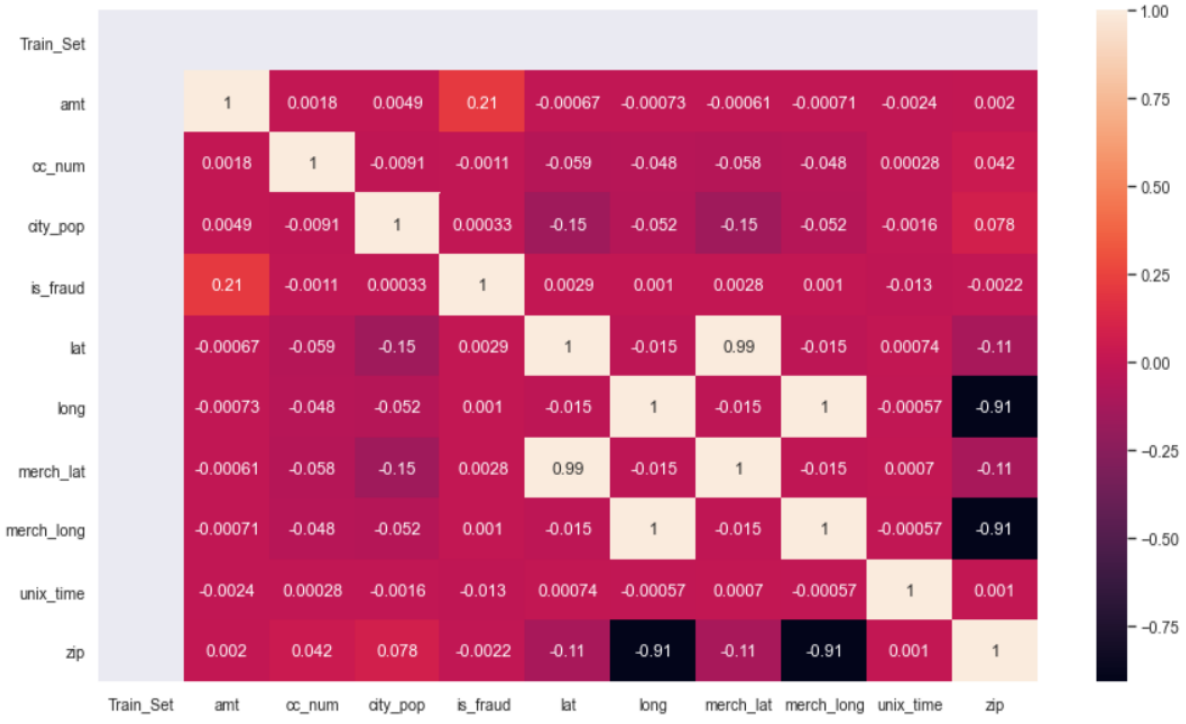

FIGURE 4.1: Correlation matrix for numeric features of the dataset.

The correlation matrix in Figure 4.1 revealed the following key notable relationships. The longitude and latitude of the merchant's physical location and the home address of the credit card holder are strongly correlated. This may be an early indication that the credit card transactions are processed in regions closer to the card owner's home. The two features described need to be used creatively to extract other information from it like distance between the two coordinates.

The longitude of the merchant's physical location and the home address of the credit card holder are strongly correlated to the zip code that represent the card owner's address. We expect the zip code to have the same information as the longitude or latitude position as this represent the same area. Depending on how the features are processed, one of the features would need to be removed.

The amount transacted has a 21% correlation to the target label as compared to the 1% correlation of the feature, time. We expect the amount of transaction to have more weight of evidence in predicting the target label than any other feature in the current dataset.

4.2.2 Characteristics of the processed dataset

4.2.2.1 Categorical features

The features were explored and their unique entries were grouped appropriately using the weight of evidence of each entry. Below, the feature *category* processing technique is shown. Figure 4.2 shows the distribution of each category entry. Most transactions are in the *gas and transport* category followed by *grocery* and *home* items. *Travel* constitute the least number of transactions in the dataset.

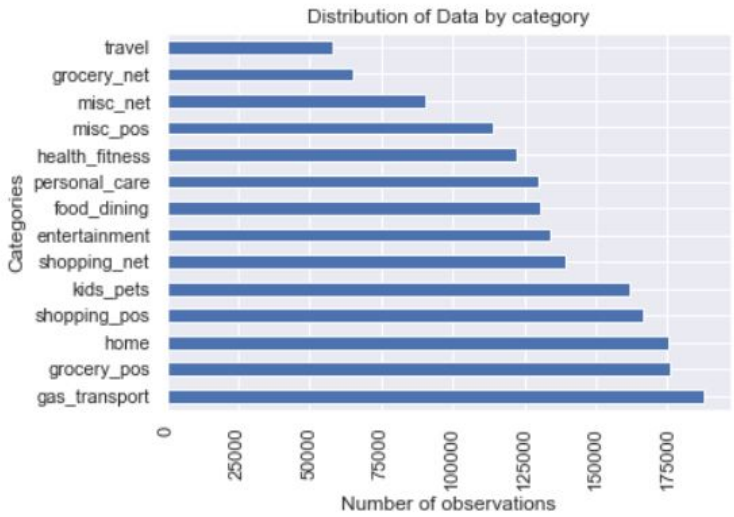

FIGURE 4.2: Distribution of entries in the feature:*category*.

Figure 4.3 shows the WoE of each entry in determining whether the transaction in these category of merchant is legitimate or not.

The entry with the lowest WoE value is dropped to eliminate dummy feature trap as caused by a feature or entry that can be inferred by others. New categories are created by merging those entries with equal or similar WoEs. In Figure 4.3, *category* : *health_fitness_home_food_dining* is created by merging *category* : *health_fitness*, *category* : *home*, and *category* : *food_dining*. The other groups encircled are also merged whilst the unique WoE entries are left as is.

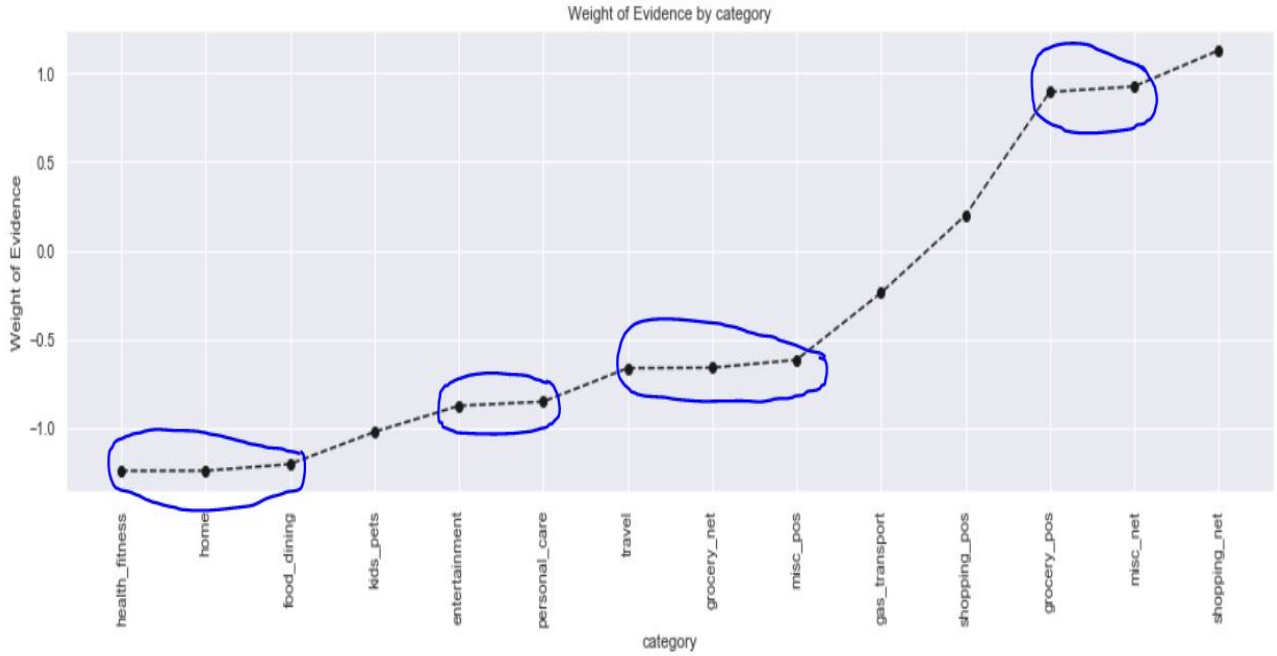

FIGURE 4.3: WoE for the entries of the feature:category.

Most discrete features had more than 100 unique entries and as such the dummy entries created were named and binned based on the values of the WoE the entries belonged to. In such cases, the binned class will be used to infer the group of dummies it was extracted from when needed.

4.2.2.2 Continuous features

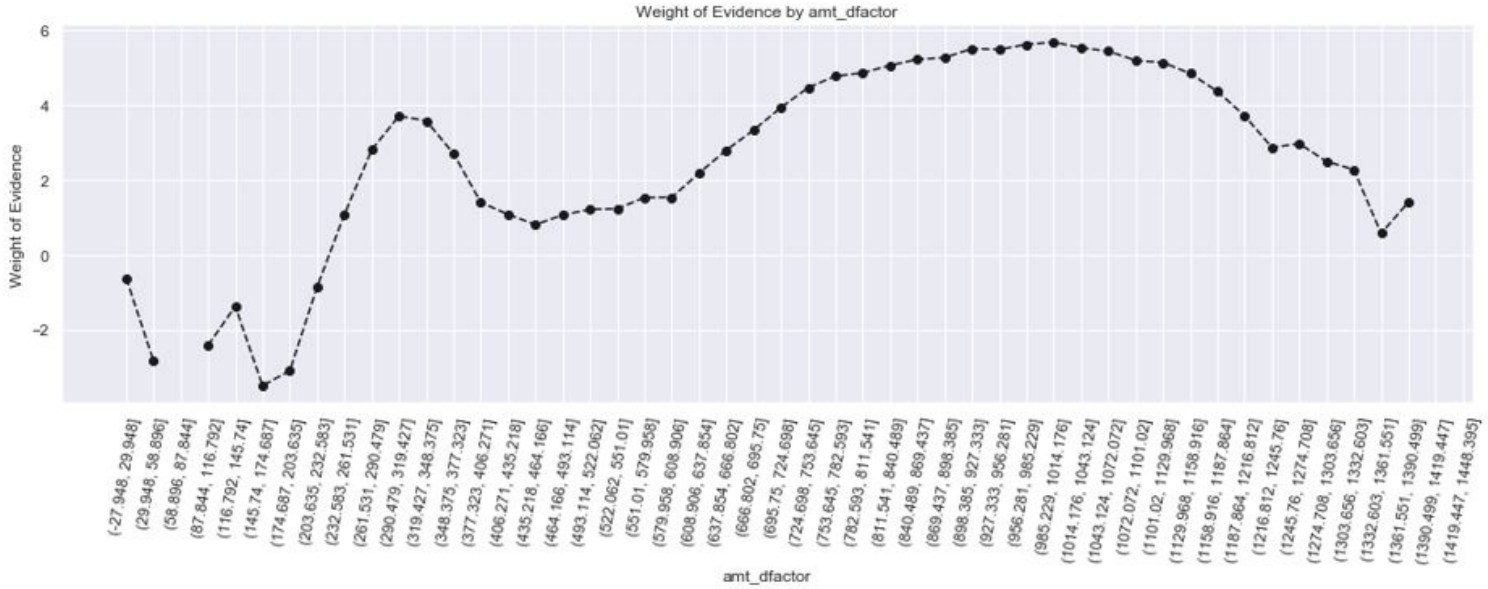

FIGURE 4.4: Amount spent at merchant.

All numeric and continuous features exhibited characteristics of uniqueness and randomness. The distribution of the WoE did not reveal much information. Many values were unique and there was no clear relationship of the WoE to the target. Figure 4.4 shows the amount feature and its WoE plot.

The plot shows that there is a minimal pattern in the distribution of the WoE and as such all numeric continuous features were not altered. These became part of the model input as is.

The column zip was dropped as a result of being strongly correlated to the longitude and latitude of the merchant and also to the credit card owner's location. The pre-processing included the extraction of temporal information from date time stamp, age from date of birth and distance from coordinates. All categorical features were then converted into dummies using one hot encoding resulting in 80 features which were then used to train the data-balanced and hyper-parameter tuned model.

4.2.3 Engineered Features

New features were created from existing ones in an attempt to improve the prediction power of the model created from the data-balanced and tuned hyper-parameter. Only continuous features were used as shown in Figure 4.5. The features, amount of transaction, the distance from the merchant, the customer's home and the customer's age were used. There are two sets of new features, ratios where the features were divided and products of which the features were multiplied. A correlation matrix was done and the new features, amt_x_age and amt_x_dist were dropped due to a strong correlation to other features. It was also evident that *amt*, *age* and *harvesine_dist* had to be removed as new features were strongly correlated to these original continuous features.

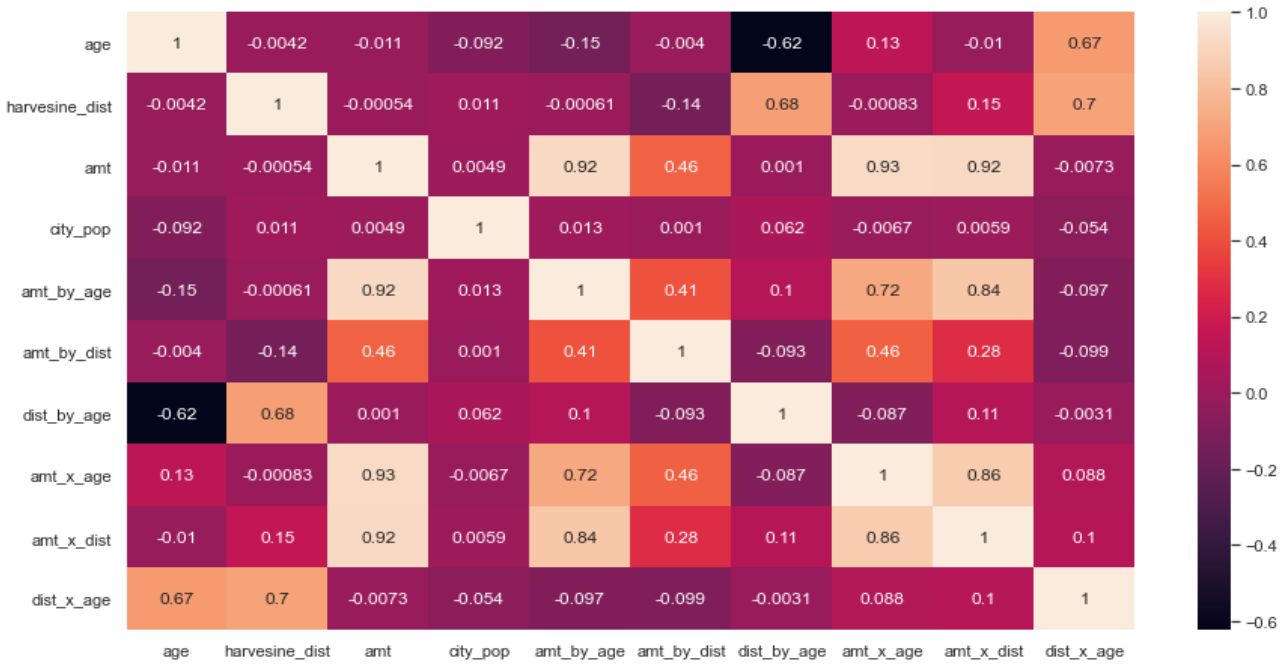

FIGURE 4.5: Six new features created from ratios and products of continuous features.

The final model was built using 84 features.

4.3 Summary

The column zip was dropped as a result of being strongly correlated to the longitude , latitude of the merchant and credit card owner's location. Strong correlation was determined by the score $\geq$ 0.7 as outlined in Table 4.2 and one feature of the correlated pair dropped to eliminate overfitting and model instabilities [39]. Temporal information, age and distance were extracted from original features. All categorical features were then converted into dummies using one hot encoding resulting in 80 features which were used to train the data-balanced and hyper-parameter tuned model. Six new features were created from ratios and products of continuous features to improve and enhance the performance of the data-balanced and hyper-parameter tuned model. Amt_x_age and amt_x_dist were dropped due to strong correlation to other features. The final model was built using 84 features.

Chapter 5

Results and Discussion

5.1 Introduction

In the previous chapter, the characteristics of the original features of the dataset were described. Examples were shown on how categorical and continuous features were analysed and prepared. Features that were dropped due to multicollinearity were stated and new features were extracted from existing ones. New features were engineered in an attempt to enhance the performance of the model to achieve the optimal credit card fraud detection model.

In this chapter, two sets of results are presented. The results of a balanced-data tuned hyper-parameter model are presented in Section 5.2.1 and the enhanced, final model in Section 5.2.2. A comparison of an iForest and XGBoost model is presented in iterative stages and the best model is established. The key features enabling the superior model's performance are also presented. The position of this work in relation to previous work is determined and the presented results are summarised in Section 5.3.

5.2 Model Results

5.2.1 Default and Optimised hyper-parameter models - Balanced Dataset

The dataset was balanced using the SMOTE and Adasyn sampling techniques. The two machine learning models were then trained and tested. The results of the XGBoost and iForest models using a balanced dataset are presented below.

Balancing the dataset using SMOTE and Adasyn sampling techniques resulted in similar performance across all metrics as represented by a single table, Table 5.1. Sampling the dataset reduced the accuracy slightly, by 0.2% for the training set compared to 3.3% for the testing set. This comparison is done against the baseline model (imbalanced dataset and default models) shown in Appendix A Table A.9. For what we lost in accuracy, we gained in the overall improved predictive power of the model. The recall, precision and the f1-score significantly increased to values above 92% for both training and testing sets when identifying fraudulent transactions.

TABLE 5.1: Evaluation of the SMOTE and Adasyn-balanced dataset - XGBoost default classifier.

	Confusion Matrix	Accuracy	Class	Precision	Recall	f1 Score	Cohen Kappa	Gini Coeff
Train	$\begin{bmatrix} 1284065 & 2701 \\ 5104 & 1286468 \end{bmatrix}$	99.6%	Class-0 Class-1	100% 100%	100% 100%	100% 100%	0.99	0.99
Test	$\begin{bmatrix} 550941 & 35583 \\ 2633 & 517991 \end{bmatrix}$	96.5%	Class-0 Class-1	100% 94%	94% 99%	97% 96%	0.93	0.93

More fraudulent examples have been synthesised and the model is able to learn patterns to identify fraud. The Cohen's kappa and the gini are significantly higher showing effects of decreased overfitting. The training and testing gini co-efficient for the balanced dataset with a default XGBoost classifier was 99% compared to 93%, a 6% difference. On imbalanced dataset (see Appendix A Table A.9), this was 83% versus 65%, 18% difference. The big difference showed an existence of overfitting.

The Kolmogorov-Smirnov statistic plot showed a visual improvement of the ability of the model to separate the two classes in Figure 5.1. 0.994 prediction confidence at a threshold of 0.563 for training and 0.955 prediction confidence at a threshold of 0.122 for the testing set.

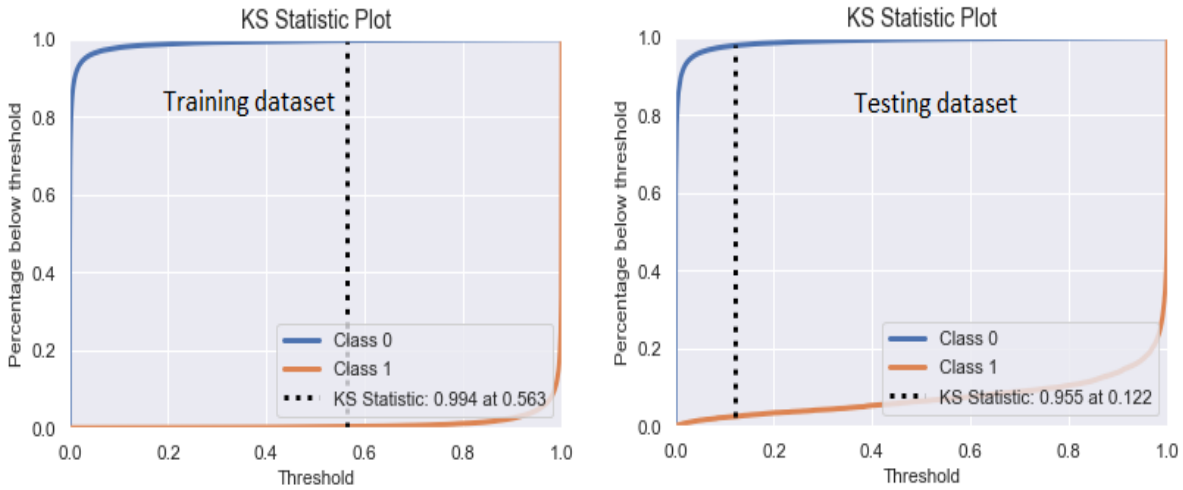

FIGURE 5.1: KS statistic plot for the SMOTE-balanced - XGBoost default classifier.

For the training set, there is 99.4% confidence in predicting 43.7% of fraudulent instances. The confidence reduces slightly to around 95% for the 56.3% of the data instances on the left of the threshold line (dotted line) due to confusion in predictions. There was however a visual representation of how different the training and testing performance is. In the testing set, the confidence of predicting fraudulent transactions is 95.5% at 12.2% of the data, the confidence dropped until 80% for the next

85% of the data and then reduce sharply towards total confusion for the remainder. There is a significant performance drop between the training and testing results showing evidence of overfitting and an opportunity to enhance the model built.

The results for the iForest model are shown in Table 5.2. Only the class 1 results are shown. The results reveal how the iForest models struggle to identify the target class, class 1 in training and testing set. The model failed to identify the negative class as well, evident from low accuracy scores in all three *n_estimators* values evaluated. The unsupervised algorithm showed evidence of struggles in dealing with a multi-dimensional mixed data type dataset.

TABLE 5.2: Evaluation of training and testing results on the SMOTE dataset- iForest with different *n_estimators* classifier for the target class only.

	n estimators	Accuracy	Class	Precision	Recall	f1 Score	Cohen Kappa	Gini Coeff
Train	20	35%	Class-1	45%	45%	45%	-20%	-0.3
	50	33%	Class-1	33%	33%	33%	-39%	-0.33
	300	34%	Class-1	34%	34%	34%	-34%	-0.34
Test	20	24.4%	Class-1	24%	24%	24%	-51%	-0.51
	50	23.9%	Class-1	24%	24%	24%	-52%	-0.52
	300	21.6%	Class-1	22%	22%	22%	-56%	-0.56

The iForest model was again tested when dataset dimensions were reduced (From 80 features to 30 random features) and the results are shown in Figure 5.3. The results showed an improved predictive power of the model with the reduction in input features. However, the performance was still poor and incomparable to any XGBoost model variants. The results of the Cohen's kappa and gini co-efficient showed a great disagreement between raters. It was concluded that the iForest model was not the right fit for the dataset (multi-dimensional mixed data type) in our work. It failed to scale up to any comparable measure to its supervised counterpart. The Adasyn sampling technique performance was worse than that of the SMOTE technique on the iForest model. The insights gathered from this experimentation however were that reducing the number of features and estimators improved the performance of an iForest model in a multi-dimensional mixed data type dataset.

TABLE 5.3: Evaluation of training results on the SMOTE and Adasyn-balanced dataset- iForest with different *n_estimators* classifier for the target class only with reduced features.

n estimators	Accuracy	Class	Precision	Recall	f1 Score	Cohen Kappa	Gini Coeff
20	45%	Class-1	45%	45%	45%	-20%	-0.23
50	43%	Class-1	43%	43%	43%	-39%	-0.33
300	44%	Class-1	44%	44%	44%	-23%	-0.34

The XGBoost model was optimized by identifying three critical hyper-parameters to tune. The three hyper-parameters and the values explored are listed in Table 5.4. A grid search was done to determine the optimal hyper-parameters for the model. The number of the grid array were limited to

optimise computational time. The grid search results shown in Table 5.5, ranked by the most optimal, showed that the optimal parameters were: *learning_rate* = 0.1, *max_depth* = 10, *n_estimators* = 40.

TABLE 5.4: Hyper-parameter values for grid search optimisation.

XGBoost Parameter	Description	Values
learning_rate	Boosting learning rate	0.1
max_depth	Maximum tree depth for base learners	8, 9, 10
n_estimators	Number of trees to fit	40, 50, 75

TABLE 5.5: Grid search results ranked by the mean test score.

Mean test score	learning_rate	max_depth	n_estimators
0.99789	0.1	10	40
0.99788	0.1	9	40
0.99787	0.1	10	50

The tuned hyper-parameter XGBoost model in Table 5.6 showed that the training and testing gini co-efficient for the balanced dataset with tuned hyper-parameter XGBoost classifier was 98% compared to 94%, a 4% difference. This showed that optimising the model improved the robustness of the model as overfitting was reduced. The SMOTE balanced dataset produced a better model that Adasyn balanced model in this instance as seen in Table 5.7. The training and testing gini co-efficient was 99% compared to 93%, a 6% difference. The model trained well but a bigger difference than the SMOTE balanced model showed that the Adasyn model overfits more in this instance.

TABLE 5.6: SMOTE-balanced dataset - Tuned hyper-parameter XGBoost.

	Confusion Matrix	Accuracy	Class	Precision	Recall	f1 Score	Cohen Kappa	Gini Coeff
Train	$\begin{bmatrix} 1277120 & 10888 \\ 12049 & 1278281 \end{bmatrix}$	99.1%	Class-0	99%	99%	99%	0.98	0.98
			Class-1	99%	99%	99%		
Test	$\begin{bmatrix} 557978 & 27214 \\ 5596 & 526360 \end{bmatrix}$	97%	Class-0	99%	95%	97%	0.94	0.94
			Class-1	95%	99%	97%		

TABLE 5.7: Adasyn-balanced dataset - Tuned hyper-parameter XGBoost.

	Confusion Matrix	Accuracy	Class	Precision	Recall	f1 Score	Cohen Kappa	Gini Coeff
Train	$\begin{bmatrix} 1284065 & 2701 \\ 5104 & 1286468 \end{bmatrix}$	99.6%	Class-0	100%	100%	100%	0.99	0.99
			Class-1	100%	100%	100%		
Test	$\begin{bmatrix} 550941 & 35583 \\ 2633 & 5179991 \end{bmatrix}$	96.5%	Class-0	100%	94%	97%	0.93	0.93
			Class-1	94%	99%	96%		

5.2.2 Final XGBoost Model - Enhanced Dataset

The results of the tuned hyper-parameter XGBoost model with new features enhanced the performance of the model as seen in Table 5.8.

TABLE 5.8: Balanced dataset with new features and optimised hyper-parameters XGBoost classifier: SMOTE and Adasyn results.

	Confusion Matrix	Accuracy	Class	Precision	Recall	f1 Score	Cohen Kappa	Gini Coeff
SMOTE								
Train	$\begin{bmatrix} 1276697 & 16137 \\ 12472 & 1273032 \end{bmatrix}$	98.89%	Class-0	99%	97%	98%	97.78%	0.9778
			Class-1	97%	99%	98%		
Test	$\begin{bmatrix} 548065 & 536975 \\ 5509 & 536975 \end{bmatrix}$	98%	Class-0	99%	97%	98%	96%	0.96
			Class-1	97%	99%	98%		
Adasyn								
Train	$\begin{bmatrix} 1276758 & 12411 \\ 15795 & 1271701 \end{bmatrix}$	98.90%	Class-0	99%	99%	99%	97.81%	0.9781
			Class-1	99%	99%	99%		
Test	$\begin{bmatrix} 548229 & 16581 \\ 5345 & 536075 \end{bmatrix}$	98.01%	Class-0	99%	97%	98%	96.03%	0.9603
			Class-1	97%	99%	98%		

On SMOTE-balanced dataset, the average of the precision, recall and f1-score is 99% for the training and 97% for the testing phase. The training and testing gini co-efficient for the balanced dataset with tuned XGBoost classifier were 97.78% compared to 96%, a 1,78% difference. The model was more robust and generalised well on training set to enable improved prediction strength on the testing set. On Adasyn-balanced dataset, the training and testing gini co-efficient were 97.81% compared to 96.03%, a 1,78% difference. The difference between training and testing was the same as that observed on SMOTE-balanced results but the Adasyn model predictive power is slightly higher, by

0.03%.

The results of both sampling techniques showed a reduction in overfitting and further improved prediction power on testing set when new features were added. This was achieved due to a slight decrease in training bias which improved generalisation in training to improve prediction strength.

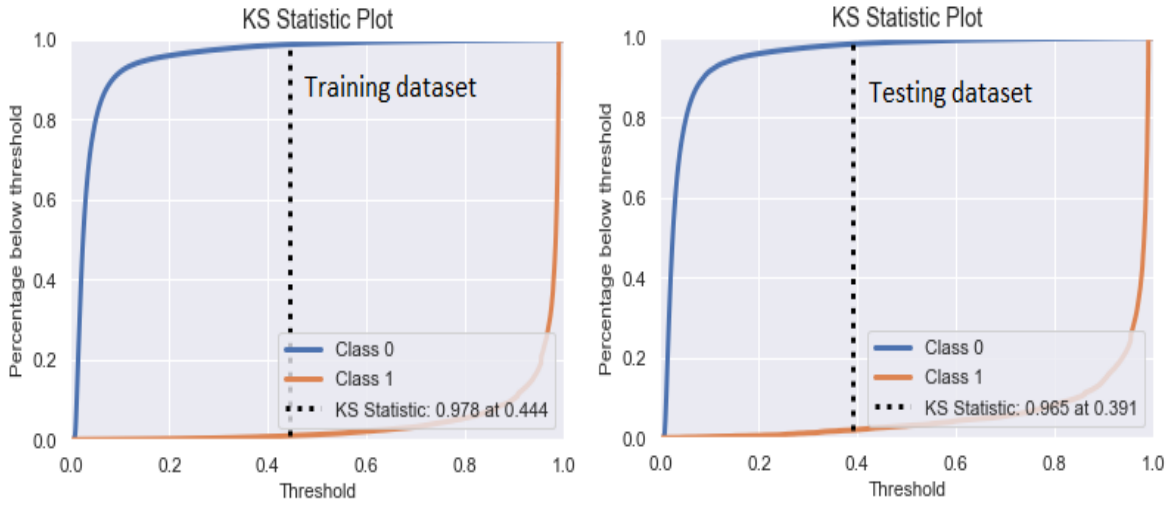

FIGURE 5.2: KS Statistic plot for the SMOTE-balanced with new features - Tuned hyper-parameter XGBoost classifier.

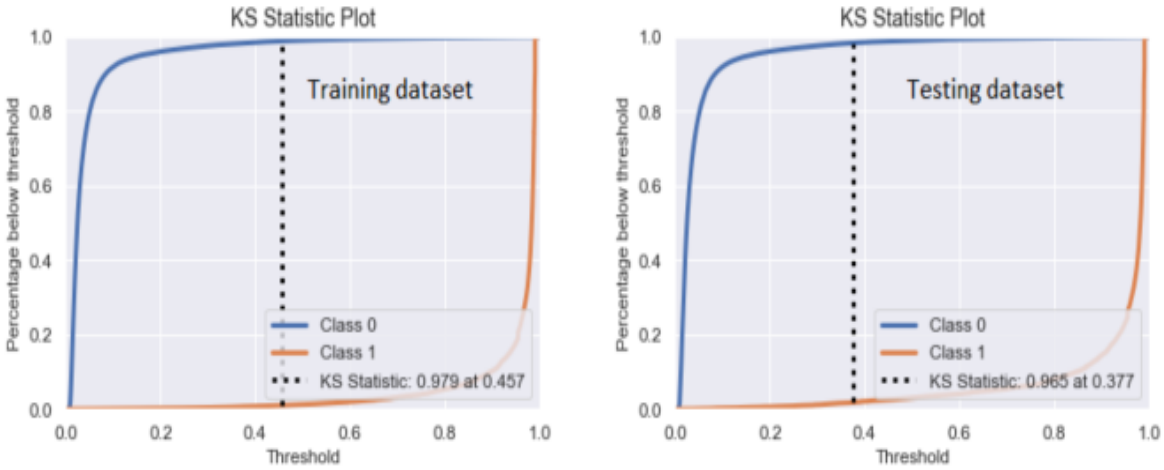

FIGURE 5.3: KS Statistic plot for the Adasyn-balanced with new features - Tuned hyper-parameter XGBoost classifier.

The SMOTE-balanced, tuned hyper-parameter XGBoost model is represented graphically by the KS statistic plot in Figure 5.2. The separation power of the classes for the training and testing sets are closely related, 0.978 prediction confidence at a threshold of 0.444 on training set compared to 0.965 at 0.391 on testing. A difference of 0.013 on the separation threshold gap and a 0.053 improvement on the cut-off threshold. The training results meant that for 55.6% of the remaining dataset, the model

has a 97.8% prediction confidence on the desired label. The confidence reduced to 90% as the prediction tends towards 90% of the dataset and then there is evidence of a huge dip for the last 10% of the dataset. The model only gets very confused for only 10% of the dataset.

The Adasyn-balanced results are better than the SMOTE-balanced model. The model showed a more improved generalisation. The separation power of 0.979 prediction confidence at a threshold of 0.457 on training set compared to 0.965 at 0.377 on testing set. A difference of 0.014 on the separation threshold gap and an improvement of 0.080 on the cut-off threshold. For a similar separation power, the testing set predict well for 62.3% of the dataset as compared to 54.3% in the training set. A significant improvement when predicting on testing data. Adasyn-balanced model is chosen as the superior model as it generalises well and is more likely to be consistently stable.

The final model has an AUROC of 0.998. The model is able to separate the two classes very well (0.998≈1).

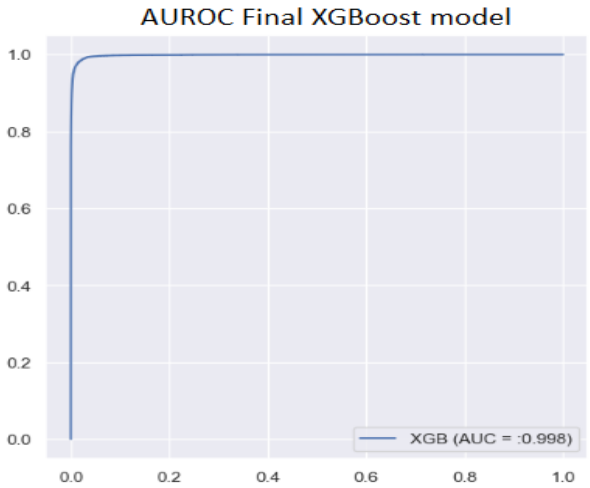

FIGURE 5.4: AUROC - Final model.

TABLE 5.9: Evaluation of the SMOTE-balanced dataset with enhanced dataset - iForest classifier.

	Confusion Matrix	Accuracy	Class	Precision	Recall	f1 score	Cohen Kappa	gini coeff
Train	$\begin{bmatrix} 158362 & 1130807 \\ 1130807 & 158362 \end{bmatrix}$	12.2%	Class-0	12%	12%	12%	-0.75%	-0.75
			Class-1	12%	12%	12%		
Test	$\begin{bmatrix} 87388 & 466188 \\ 87388 & 466188 \end{bmatrix}$	15.7%	Class-0	16%	16%	16%	-68%	-0.68
			Class-1	16%	16%	16%		

The performance of the iForest model got worse when new features were created as shown in Table 5.9. The precision, recall and f1-score were very poor and incomparable to any XGBoost models.

Worse than a random guess. It was shown in Table 5.3 that the iForest model improved when features were reduced and not added. This test was a further elaboration of how the iForest model was not the right fit for the multi-dimensional mixed data type dataset in our work.

The final and optimal model is the XGBoost classifier with a *learning_rate* of 0.1, *max_depth* = 10 and *n_estimators* = 40. The dataset for the optimal model is sampled using Adasyn. In Table 5.10, the most important features that gives the superior model its predictive strength are outlined.

TABLE 5.10: Top predictive features: Adasyn-balanced and optimised hyper-parameter XGBoost classifier.

Feature	Dummy/ Extracted feature	Entries	Predictive Power	Total Predictive Power
amt	*amt_by_age*		8.5%	8.5%
age				
merchant		$-0.5 \leq WoE \leq 0$	3%	8%
		$-2 \leq WoE \leq -1.5$	5%	
trans_date_trans_time	*weekday*	*Saturday*	3%	18%
		Wednesday	4%	
	month_day	$0.5 \leq WoE \leq 1$	3.7%	
		$-1.5 \leq WoE \leq -1$	4.3%	
	dayofmonth	1_6	3%	
job		$0.5 \leq WoE \leq 1$	3.9%	13.7%
		$0.1 \leq WoE \leq 0.5$	3.8%	
		$-1.5 \leq WoE \leq -1$	6%	

The top eleven predictors had a total predictive power of 48.2%. The newly created feature *amt_by_age*, had a predictive power of 8.5% compared to other features. From the original dataset, temporal information from the feature *trans_date_trans_time*, constituted 18% of total predictive power. This is evidence that temporal information is critical in fraud detection patterns as stated in previous research papers. The type of job was the second highest for the original feature with dummy features constituting 13.7% predictive power.

The dummy entries in the top predictors Table 5.10 that are represented in terms of the WoE and other non-intuitive symbols are described in Appendix A. The job entries are detailed in Table A.1, A.2, and A.3. The day of the month feature is detailed in Table A.4, the merchant entries in Table A.6 and A.7 whilst the *month_day* feature is represented in Table A.8.

This work established that the unsupervised model failed to scale up with multi-dimensional mixed data type dataset and transactions that are dynamic. We could not replicate the effectiveness of iForest as the data for this work was not purely numeric as in [3, 29]. Fraudulent transactions mimic legitimate transactions so closely that an unsupervised model failed to discriminate the two classes. This agreed with work from statistical models and theoretical discussion that unsupervised and/or

traditional statistical models are mainly exploratory and are inconclusive [21]. These findings have thus further reinforced lessons from other previous work.

Temporal information is a key aspect of the dataset and needs to be extracted for a good model to be built. In our model, the date and time stamp contributed the highest percentage to the predictive power.

It was shown that indeed, Adasyn sampling technique is a better version of the synthetic oversampling techniques [17]. Adasyn out-performed SMOTE in our work and this was due to its ability to learn minority instances better and replicate them well.

Isolation forest did not have a KS statistic metric functionality on Python and reinforced the idea that unsupervised and statistical models were not vigorously evaluated by complex metrics as they prioritise interpretability than accuracy [21]. The failure of this outlier-based method to scale up to the supervised method was proof that the anomaly approach is best applicable in dataset that are either purely numeric or has a lot of successive samples to draw distinct patterns from.

The use of feature creation and selection methods, weight of evidence to bin and class features, Adasyn sampling were critical in creating a robust XGBoost model that measured, 98% f1-score, Cohen's kappa of 0.9603 and 96.03% gini coefficient for the testing dataset. The superior model achieved less than 2% difference in training and testing across all significant metrics (precision, recall, f1-score, Cohen's kappa and gini co-efficient). Our model performance scaled well amongst other fraud detection systems descibed in Section 2.4.2. It is better than any ensemble method and scales well with complex neural network models.

5.3 Summary

The iForest model was shown to not being the right fit for the multi-dimensional mixed data type dataset in this work as it failed to scale up to any comparable measure to its supervised counterpart. This was also shown to be because the data was not purely numeric. The insights gathered from this experimentation however was that an unsupervised model work best when data is of a lower dimension.

Adasyn sampling technique was shown to be superior than SMOTE technique. The model built using Adasyn sampling technique was able to reduce overfitting and was shown to generalise well. The optimal hyper-parameters for the final model, the XGBoost classifier for this dataset were: $learning\_rate$ = 0.1, max_depth = 10, $n\_estimators$ = 40. It had a 97.9% separation power of the classes during training at 45.7% of the dataset and 96.5% at 37.7% of the dataset on testing. A difference of 0.012 on the separation threshold gap and an improvement of 0.080 on the cut-off threshold of the dataset. The testing results meant that for 62.3% of the dataset, the model has a 96.5% prediction confidence on

the desired label. The confidence reduced to 90% as the prediction tends towards 90% of the data and then there is evidence of a huge dip for the last 10% of the dataset. The model only gets very confused for the last 10% of the dataset.

The model had an average of 98% f1-score, Cohen's kappa of 0.9603 and 96.03% gini coefficient for the testing dataset as compared to 98% f1-score, Cohen's kappa of 0.9781 and 97.81% gini coefficient for the training set. The model had on average, less than 2% difference in training and testing across all significant metrics. The created aggregates introduced a variation and more generalisation on the top eleven predictive features hence the improved, reliable, and robust model.

The top eleven features constituted only 48.2% prediction power. The top eleven predictive features were, *amt_by_age*, two *merchant* dummy groups, two *weekday* entries, two *month_day* entires, one *dayofmonth_day*, and three *job* dummy groups. These were made up by five original dataset features. A new feature which is the product of the amount spent on the transaction and the age of the customer improved the predictive power of the model by enhancing generalisation and reducing overfitting. No single feature contributes more than 18% total predictive power in the original features. This showed that the model is stable, and that the absence of any single feature cannot destabilise the performance of the model. The model built was thus reliable.

Chapter 6

Summary, Conclusion and Recommendations

6.1 Summary

The purpose of the research work was on the limitation and elimination of financial losses by designing a credit card fraud detection system that is based on the misuse of credit card. The fraud detection system type was limited by the synthetic data used which did not have successive sample data to characterise the personalised transactional behaviours. The richness of data, censored real data and the dynamic nature of fraud characteristics are the limitations of credit card fraud detection systems.

Extreme gradient boosting machine and isolation forest techniques were compared, where an in-depth analysis of their predictive power was explored. This work enhanced the knowledge bank of work that employed XGBoost, isolation forest or a comparison of the two well-established machine learning techniques. This was achieved by extensively investigating the effect of large multi-dimensional mixed data type on models, sampling and feature selection and creation techniques.

The dataset was sourced from Kaggle, a simulated credit card transactional dataset from Sparkov Data Generation. It was available in two sets, the training set (1 296 675 data points) and testing set (555 719 data points). 99.42% of the transactions were non-fraudulent. Imputing was used to handle missing values, one hot encoding to transform unusable categorical features into useful ones and the feature split and extract were used to convert features that had multi-dimensional insights. The binning technique used was based on the weight of evidence as it quantified how well a predictive feature explained the target. Adasyn and SMOTE sampling techniques were compared in the building of the models. The model is built on Python 3 and evaluated by the well-established methods; precision, recall, f1-scores that are based on the confusion matrix. The other statistical evaluation metrics were Cohen's kappa, gini coefficient, the auroc, and the KS statistic plot.

6.2 Conclusion

The iForest model was incomparable to any XGBoost model variants due to multi-dimensions and mixed data types. It was concluded that the iForest model was not the right fit for the dataset in this work as it failed to scale up to any comparable measure to its supervised counterpart. The complex nature of this work favoured a model that learnt from labels as many fraudulent transactions mimic legitimate transactions so closely. It was gathered however, that reducing the number of features and estimators improved the performance of an iForest model in a multi-dimensional mixed data type dataset.

Adasyn sampling technique was shown to be superior than SMOTE technique. The model built using Adasyn sampling technique was able to reduce overfitting and was shown to generalise well. The optimal parameters for the final XGBoost classifier for this dataset were: *learning_rate* = 0.1, *max_depth* = 10, *n_estimators* = 40. It had a 97.9% separation power of the classes during training at 45.7% of the dataset and 96.5% at 37.7% of the dataset on testing. A difference of 0.012 on the separation threshold gap and an improvement of 0.080 on the cut-off threshold of the dataset. The testing results meant that for 62.3% of the dataset, the model has a 96.5% prediction confidence on the desired label. The confidence reduced to 90% as the prediction tends towards 90% of the data and then there is evidence of a huge dip for the last 10% of the dataset. The model only gets very confused for the last 10% of the dataset.

The top eleven predictive features were constituted from five original features. The features constituted only 48.2% prediction power. One new feature, *amt_by_age*, which is the product of the amount spent on the transaction and the age of the customer. A ratio that combine a transactional and demographic feature with 8.5% predictive power. Three dummy features from a demographic original feature *job*. The three dummy feature had a 13.7% predictive power. Two *merchant* dummy groups with 8% predictive power. The remainder of the top features are made up by five transactional features (temporal information) with 18% total predictive power. This showed that temporal dimension of fraudulent transactions provides a rich source of insights in the pattern of credit card fraud.

No single feature contributes more than 18% total predictive power in the original features. This shows that the model is stable, and that the absence of any single feature cannot destabilise the performance of the model. The created aggregates introduced a variation and more generalisation on the top eleven predictive features hence the improved, reliable, and robust model.

6.3 Recommendations

After an extensive investigative work on building a robust fraud detection system, these are the recommendations:

Use supervised techniques as they yield higher predictive power and reliability for dynamic, multi-dimensional mixed data type problems. The work presented in this research showed that an outlier-based method does not scale well in a multi-dimensional mixed data type dataset that has dynamic patterns that mimics normal behaviour. XGBoost is the recommended model over an isolation forest model for fraud detection systems.

Temporal information is a key aspect of the dataset and needs to be extracted for a good model to be built. The model built shows that the date and timestamp contributed the highest percentage to the predictive power.

Use an Adasyn sampled dataset as it learns better and is able to synthesise samples using a weighted score that enables robustness and generalisation for future predictions.

For reliability and stability, chose a model that does not rely heavily on a single feature. Models that rely heavily on selected features tend to be unstable when the feature is missing from the dataset.

6.4 Future Work

Source real data with successive samples. This will expand the research to tackle behavioural trends and to improve anomaly results. Individualised patterns would be studied and enhance the database on which the model derive its detection algorithm.

Explore the use of principal component analysis in mixed data type datasets to reduce the dimensionality of modern dataset to improve the performance of unsupervised models.

Appendix A

Graphs, tables, formulae and Definitions

A.1 Main Section

The section below contains all diagrams referred to in the main report. Some of the results are quoted in the main report but not shown graphically in the main body.

A.1.1 Pre-Processing

A selected categorical and continuous variables are represented below:

A.1.1.1 Weekday

The distribution of the weekday in Figure A.1 shows that most transactions happen on Sundays and Mondays. There are 10000 more examples in the two days than the other days.

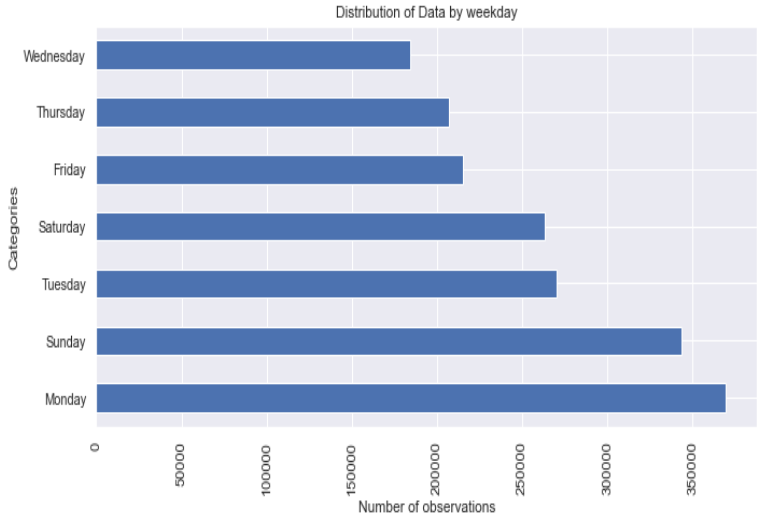

FIGURE A.1: The distribution of entries on the variables: *weekday*.

The table that shows how the WoE of the weekday has been evaluated. Figure A.2 shows the computed results ranked by the lowest WoE value. Figure A.3 graphically shows the computed results and is a clear aid in grouping entries with similar WoE.

	weekday	n_obs	prop_good	prop_n_obs	n_good	n_bad	prop_n_good	prop_n_bad	WoE	diff_prop_good	diff_WoE	IV
0	Monday	369418	0.004017	0.199427	1484.0	367934.0	0.153766	0.199666	-0.261213	NaN	NaN	0.030508
1	Sunday	343677	0.004626	0.185531	1590.0	342087.0	0.164750	0.185640	-0.119382	0.000609	0.141831	0.030508
2	Tuesday	270340	0.004683	0.145941	1266.0	269074.0	0.131178	0.146018	-0.107175	0.000057	0.012207	0.030508
3	Saturday	263227	0.005672	0.142101	1493.0	261734.0	0.154699	0.142035	0.085408	0.000989	0.192583	0.030508
4	Wednesday	183913	0.006117	0.099284	1125.0	182788.0	0.116568	0.099193	0.161405	0.000445	0.075997	0.030508
5	Thursday	206741	0.006370	0.111607	1317.0	205424.0	0.136463	0.111477	0.202229	0.000253	0.040824	0.030508
6	Friday	215078	0.006398	0.116108	1376.0	213702.0	0.142576	0.115970	0.206547	0.000027	0.004318	0.030508

FIGURE A.2: The calculated table for the WoE and Information Value for the entries on the variables: *weekday*.

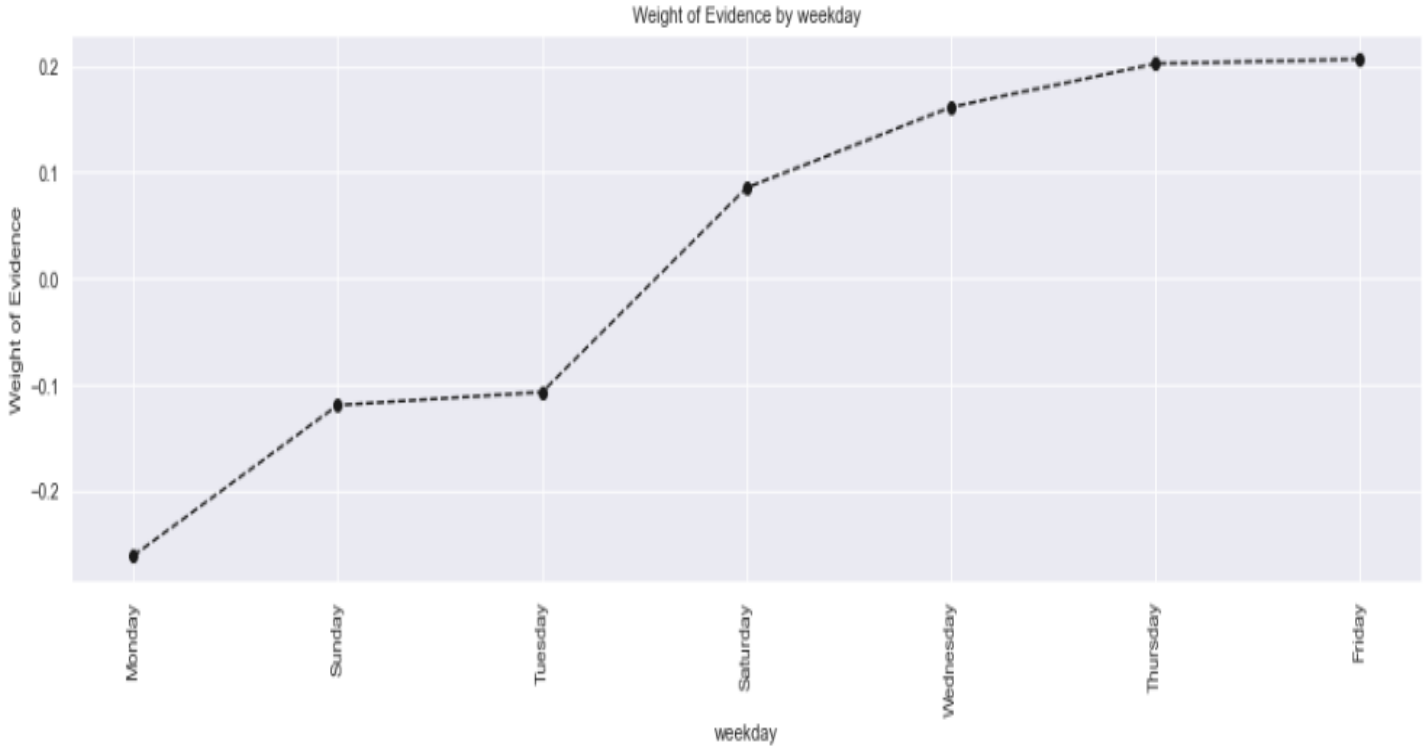

FIGURE A.3: A graphic representation of the WoE for the entries on the variables: *weekday*.

A.1.1.2 State

The distribution of the state in Figure A.4 shows that most transactions happen in ten states. These states have the highest populations and as such more transaction occur at the states.

The table that shows how the WoE of the state in Figure A.5 graphically shows that the variable state does not provide distinctive prediction. The entries of state are similarly informative.

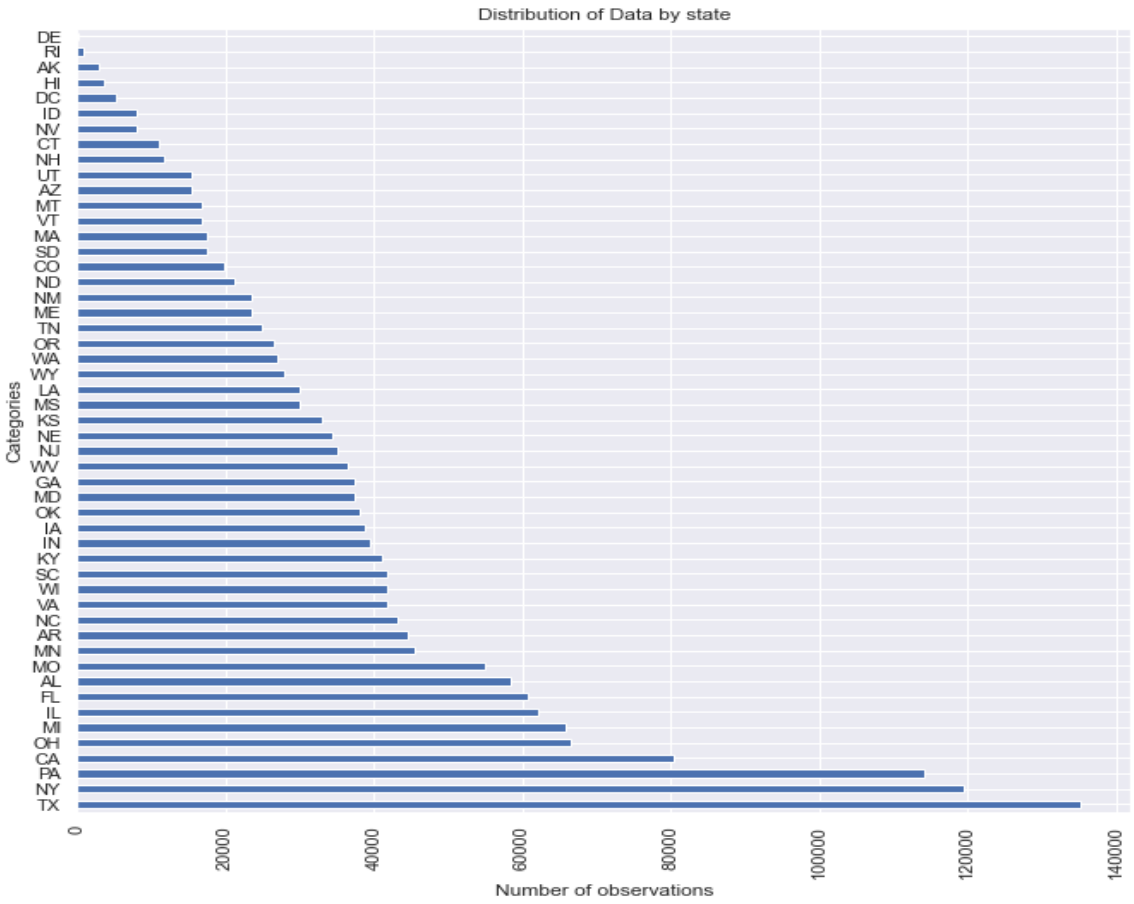

FIGURE A.4: The distribution of entries on the variables: *state*.

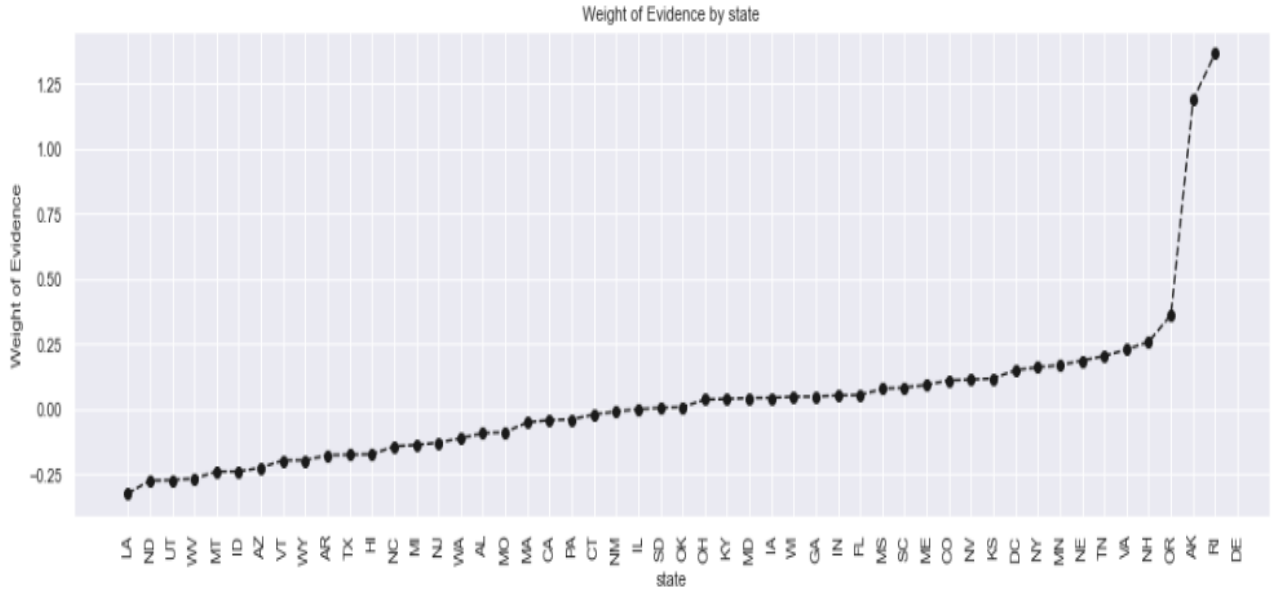

FIGURE A.5: A graphic representation of the WoE for the entries on the variables: *state*.

A.1.1.3 Age

The ranking of WoE in Figure A.6 shows that the variable age cannot be grouped by WoE values. There is no pattern in how the WoE is structured. Age was one of the continuous variables that were used in the model as is because of the indiscriminate WoE.

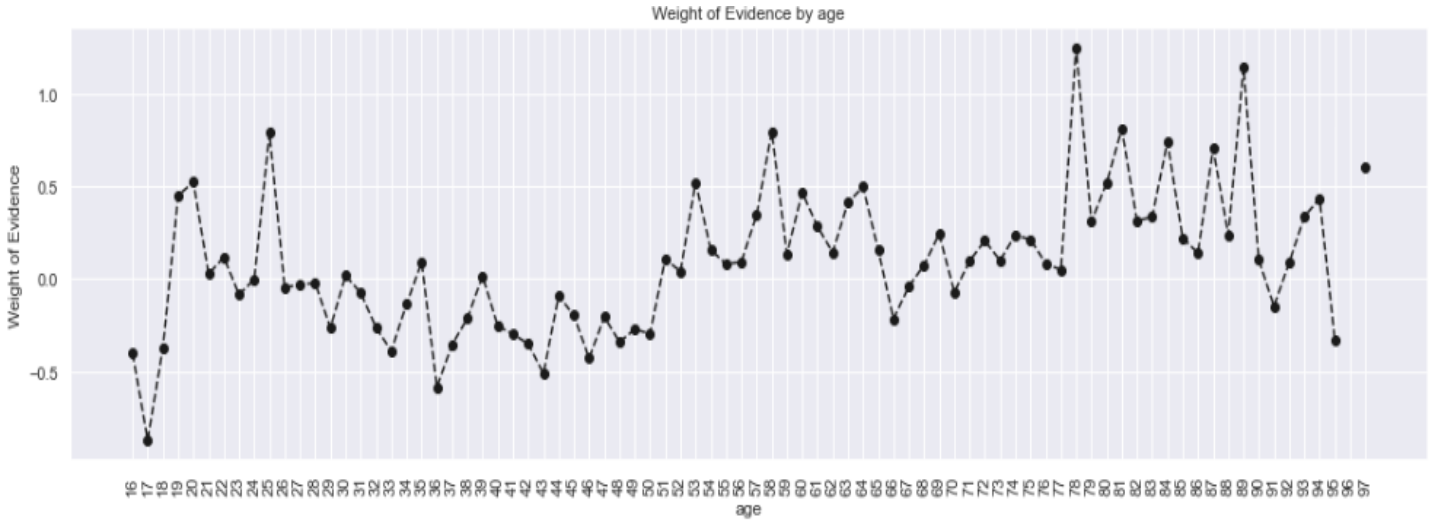

FIGURE A.6: A graphic representation of the WoE for the entries on the variables: *age*.

A.1.1.4 Harvesine distance

The indiscriminate WoE is evident in all continuous variables as seen in Figure A.7 that represent the distance variable.

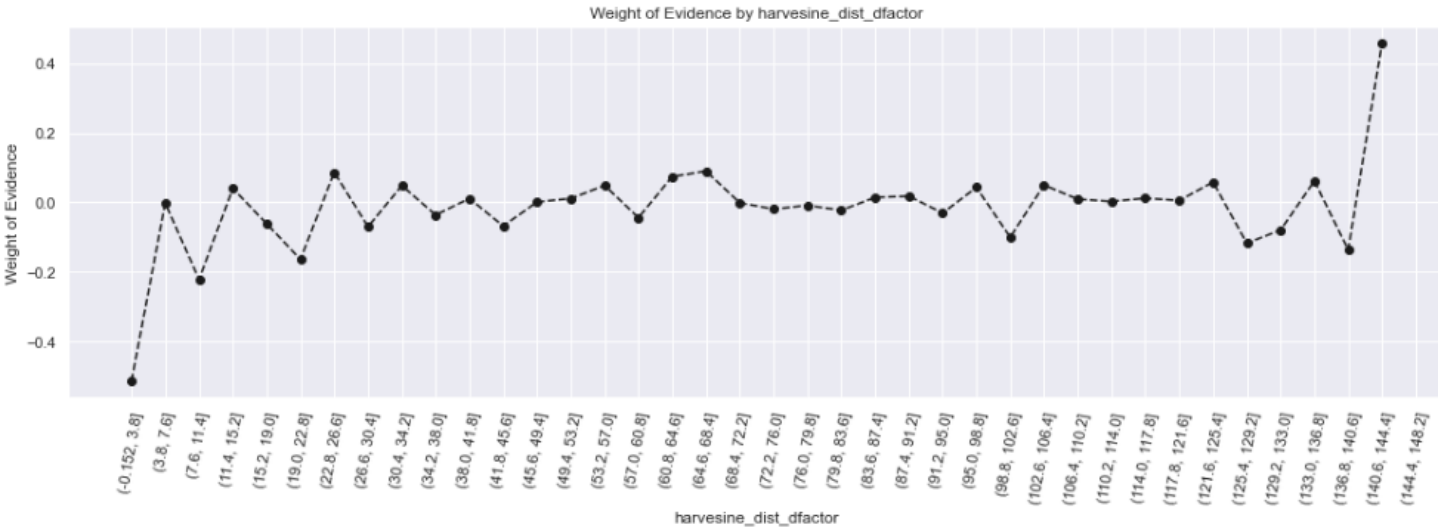

FIGURE A.7: A graphic representation of the WoE for the entries on the variables: *harvesine distance*.

A.1.1.5 Amount by Age

The indiscriminate WoE is also on the created continuous variables. Figure A.8 shows the WoE of the *amount_by_age* variable. There is no relationship on the WoE.

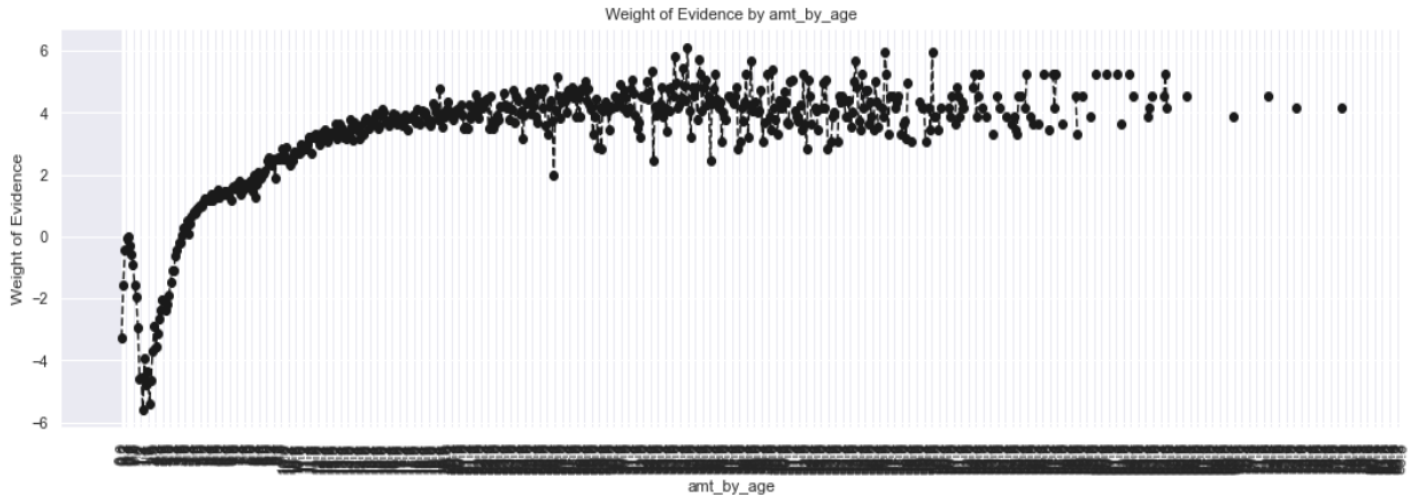

FIGURE A.8: A graphic representation of the WoE for the entries on the variables: *amount_by_age*.

A.1.2 Dataset Exploration

A snippet of the description of the dataset is shown in Figure A.9.

	count	mean	std	min	25%	50%	75%	max
age	1296675.0	47.678537	17.391256	16.0	34.00	46.00	59.00	97.0
harvesine_dist	1296675.0	76.066879	29.098666	0.0	55.30	78.20	98.40	152.0
amt	1296675.0	70.351035	160.316039	1.0	9.65	47.52	83.14	28948.9
city_pop	1296675.0	88824.440563	301956.360689	23.0	743.00	2456.00	20328.00	2906700.0
merchant:-2_WoE_-1.5	1296675.0	0.079071	0.269850	0.0	0.00	0.00	0.00	1.0
...	...	...	...	...	...	...	...	...
job:WoE=0	1296675.0	0.166922	0.372907	0.0	0.00	0.00	0.00	1.0
job:0.1_WoE_0.5	1296675.0	0.228753	0.420030	0.0	0.00	0.00	0.00	1.0
job:0.5_WoE_1	1296675.0	0.051580	0.221177	0.0	0.00	0.00	0.00	1.0
job:1_WoE_1.5	1296675.0	0.009601	0.097512	0.0	0.00	0.00	0.00	1.0
job:1.5_WoE_2	1296675.0	0.002044	0.045169	0.0	0.00	0.00	0.00	1.0

88 rows × 8 columns

FIGURE A.9: A snippet of the description of the dataset.

A.1.3 Top predictors in imbalanced dataset

The experiment with imbalanced dataset shows how the models are dependent on specific variables. In both the default and optimised XGBoost the model depends heavily on *hour* : 23_22, *category* : *travel_grocery_net_misc_pos*, *category* : *gas_transport* with a collective predictive power of 35% as shown in Figure A.10. The outcome is worse on the optimised model as *category* : *gas_transport* has

a predictive power more than 35% on its own.

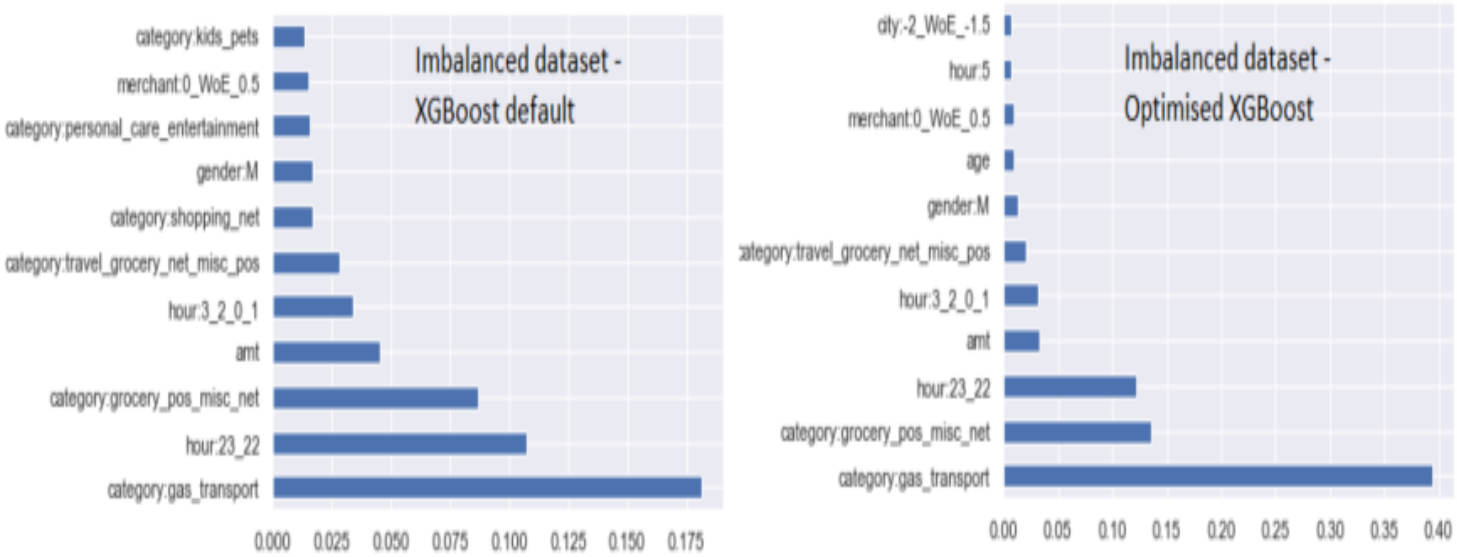

FIGURE A.10: Top predictors in imbalanced dataset.

A.1.4 Definition of Dummy Features in the Top Predictors

The most important features that gives the superior model its predictive strength are outlined in Figure A.11.

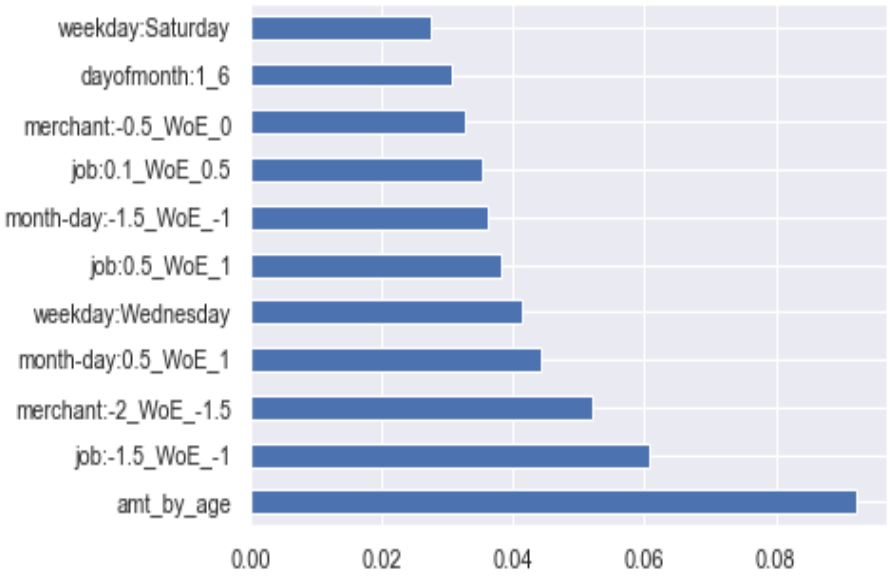

FIGURE A.11: Top predictors in the final model.

A.1.4.1 Job Features

TABLE A.1: *job* entries represented by the $0.5 \leq WoE \leq 1$ in the top predictive features

Entries		
Archaeologist	*Teaching laboratory technician*	*Broadcast presenter*
Race relations officer	*Editor, film/video*	*Hydrologist*
Tree surgeon	*Insurance risk surveyor*	*Maintenance engineer*
Media planner	*Television/film/video producer*	*Audiological scientist*
Art therapist	*Forensic psychologist*	*Occupational hygienist*
Animal nutritionist	*Lecturer, higher education*	*Licensed conveyancer*
Psychiatrist	*Administrator, education*	*Systems analyst*
Producer, radio	*Chief Technology Officer*	*Art gallery manager*
Prison officer	*Commercial horticulturist*	*Pharmacologist*
Chief Financial Officer	*Psychologist, sport and exercise*	*Associate Professor*
Teacher, early years/pre	*Public affairs consultant*	*Merchandiser, retail*
Mechanical engineer	*Further education lecturer*	*Soil scientist*
Designer, textile	*Television production assistant*	*Pathologist*
Company secretary	*Scientist, physiological*	*Education administrator*
Sports administrator	*Research officer, trade union*	*Furniture designer*
Neurosurgeon	*Armed forces logistics/support*	*Loss adjuster, chartered*
Chemical engineer	*Administrator, local government*	*Air broker*
Retail buyer	*Trading standards officer*	*Barrister*
Buyer, industrial	*Exercise physiologist*	*Charity fundraiser*
Ecologist	*Private music teacher*	*Plant breeder/geneticist*
Product manager	*Emergency planning/management officer*	*Special effects artist*
Curator	*Environmental education officer*	*Diagnostic radiographer*
Building control surveyor	*Sport and exercise psychologist*	*Therapist, music*
Mental health nurse	*Therapist, horticultural*	*Transport planner*
Engineer, petroleum	*Community pharmacist*	*Quantity surveyor*
Psychiatric nurse	*Scientist, research (medical)*	*Hospital doctor*
Psychologist, clinical	*Teacher, English as a foreign language*	*Police officer*
Animator	*Armed forces training and education officer*	*Freight forwarder*
Therapist, occupational	*Estate manager/land agent*	*Landscape architect*
Geochemist	*Conservation officer, historic buildings*	*Scientist, clinical*
Engineer, maintenance	*Designer, interior/spatial*	*Academic librarian*
Gaffer	*Chartered legal executive*	*General practice doctor*
Surveyor, mining	*Environmental health practitioner*	*Scientist, biomedical*
Science writer	*Information systems manager*	*Chartered accountant*
Chiropodist	*Museum/gallery exhibitions officer*	*Field seismologist*
Bookseller	*Development worker, international aid*	*Glass blower/designer*
Embryologist, clinical	*Outdoor activities/education manager*	*Chief Marketing Officer*
Production assistant, radio	*Training and development officer*	*Theme park manager*
Insurance claims handler	*Investment banker, corporate*	*Clinical biochemist*
Communications engineer	*Civil engineer, contracting*	*Fitness centre manager*
Education officer, museum	*Equality and diversity officer*	*Hospital pharmacist*
Programmer, applications	*Engineer, broadcasting (operations)*	*Hydrographic surveyor*
Engineer, aeronautical		

TABLE A.2: *job* entries represented by the $0.1 \leq WoE \leq 0.5$ in the top predictive features.

job	Entries	
Broadcast engineer	*Colour technologist*	*Higher education careers adviser*
Textile designer	*Network engineer*	*English as a foreign language teacher*
Biochemist, clinical	*Catering manager*	*Education officer, community*
Visual merchandiser	*Nurse, children*	*Advertising account executive*
Analytical chemist	*Marketing executive*	*Restaurant manager, fast food*
Copy	*Chief of Staff*	*Occupational therapist*
Retail manager	*Designer, multimedia*	*Surveyor, hydrographic*
Optometrist	*Horticultural therapist*	*Producer, television/film/video*
Architect	*Aeronautical engineer*	*Engineer, manufacturing*
Medical secretary	*Environmental manager*	*Senior tax professional/tax inspector*
Politician	*Engineer, materials*	*Manufacturing systems engineer*
Oncologist	*Conservator, furniture*	*Investment banker, operational*
Hotel manager	*Presenter, broadcasting*	*Commissioning editor*
Theatre director	*Water quality scientist*	*Engineer, civil (consulting)*
Wellsite geologist	*Human resources officer*	*Civil Service fast streamer*
Buyer, retail	*Public relations officer*	*Designer, television/film set*
Administrator, arts	*Primary school teacher*	*Teacher, adult education*
Set designer	*Air cabin crew*	*Museum education officer*
Sales executive	*Tour manager*	*Museum/gallery conservator*
Data scientist	*Purchasing manager*	*Research scientist (medical)*
Hydrogeologist		*Production assistant, television*

TABLE A.3: *job* entries represented by the $-1.5 \leq WoE \leq -1$ dummy.

Entries		
Interpreter	*Intelligence analyst*	*Community education officer*
Firefighter	*Child psychotherapist*	*English as a second language teacher Engineer, land*
Water engineer	*Pension scheme manager*	*Medical sales representative*
Tax inspector	*Retail banker*	*Corporate investment banker*

A.1.4.2 Day-of-Month Features

TABLE A.4: dayofmonth:1_6 entries in the Top predictors.

dayofmonth entries
1
6

A.1.4.3 Weekday Feature

TABLE A.5: *weekday* entries in the Top predictors

weekday entries
Saturday
Wednesday

A.1.4.4 Merchant Features

TABLE A.6: merchant entries for $-2 \leq WoE \leq -1.5$ in the top predictive features.

merchant	Entries	
$-2 \leq WoE \leq -1.5$	*fraud_Durgan, Gislason and Spencer*	*fraud_Denesik, Powlowski and Pouros*
	fraud_Rau − Grant	*fraud_Reichel, Bradtke and Blanda*
	fraud_Lubowitz − Walter	*fraud_Fadel − Hilpert*
	fraud_Wilkinson PLC	*fraud_Medhurst Inc*
	fraud_Hermiston, Russel and Price	*fraud_Torphy − Kertzmann*
	fraud_Bins − Tillman	*fraud_Quitzon − Goyette*
	fraud_Johns − Hoeger	*fraud_Pollich LLC*
	fraud_Johns Inc	*fraud_Hackett Group*
	fraud_Connelly − Carter	*fraud_Gutmann, McLaughlin and Wiza*
	fraud_Cummerata − Hilpert	*fraud_Crist, Jakubowski and Littel*
	fraud_Stark − Koss	*fraud_Romaguera and Sons*
	fraud_Armstrong, Walter and Gottlieb	*fraud_Kohler, Lindgren and Koelpin*
	fraud_Schulist Ltd	*fraud_Collier LLC*
	fraud_Connelly PLC	*fraud_Wiza, Schaden and Stark*
	fraud_Champlin − Casper	*fraud_Turner LLC*
	fraud_Mueller, Gerhold and Mueller	*fraud_Stiedemann Ltd*
	fraud_Kihn − Schuster	*fraud_Hagenes, Kohler and Hoppe*
	fraud_Padberg − Sauer	*fraud_Labadie LLC*
	fraud_Mante, Luettgen and Hackett	*fraud_Becker, Harris and Harvey*
	fraud_Kulas Group	*fraud_Kihn − Fritsch*
	fraud_Fritsch and Sons	*fraud_Volkman PLC*
	fraud_Graham and Sons	*fraud_Kemmer − Reinger*
	fraud_Eichmann − Kilback	*fraud_Jakubowski Group*
	fraud_Heller PLC	*fraud_Morar Inc*
	fraud_Osinski Inc	*fraud_Ortiz Group*
	fraud_Beier and Sons	*fraud_Lubowitz, Terry and Stracke*
	fraud_Kihn, Brakus and Goyette	*fraud_Brown, Homenick and Lesch*
	fraud_Weber and Sons	

TABLE A.7: merchant entries represented by the $-0.5 \leq WoE \leq 0$ in the top predictive features.

merchant	Entries	
$-0.5 \leq WoE \leq 0$	*fraud_Brekke and Sons*	*fraud_Christiansen − Gusikowski*
	fraud_Heller − Abshire	*fraud_Streich, Hansen and Veum*
	fraud_Hilpert − Conroy	*fraud_Monahan, Hermann and Johns*
	fraud_Lind, Huel and McClure	*fraud_Torp − Labadie*
	fraud_Corwin − Romaguera	*fraud_Boyer − Haley*
	fraud_Block − Hauck	*fraud_KassulkeInc*
	fraud_Ledner, Hartmann and Feest	*fraud_Smith − Stokes*
	fraud_AbshirePLC	*fraud_Schaefer, McGlynn and Bosco*
	fraud_Bins, Balistreri and Beatty	*fraud_BuckridgePLC*
	fraud_Hagenes, Hermann and Stroman	*fraud_HarrisInc*
	fraud_FritschLLC	*fraud_Bruen − Yost*
	fraud_EmardInc	*fraud_MorissettePLC*
	fraud_Stamm − Rodriguez	*fraud_Dickinson − Rempel*
	fraud_KlingInc	*fraud_Turner, Ruecker and Parisian*
	fraud_Crooks and Sons	*fraud_Nicolas, Hills and McGlynn*
	fraud_Gibson − Deckow	*fraud_Reynolds − Schinner*
	fraud_Hermann − Gaylord	*fraud_HermanInc*
	fraud_LangworthLLC	*fraud_ThielPLC*
	fraud_ShieldsInc	*fraud_Gutmann − Upton*
	fraud_CollierInc	*fraud_Satterfield − Lowe*
	fraud_HalvorsonGroup	*fraud_GislasonGroup*
	fraud_Corwin − Collins	*fraud_Olson, Becker and Koch*
	fraud_RodriguezGroup	*fraud_Adams, Kovacek and Kuhlman*
	fraud_Metz − Boehm	*fraud_Mraz − Herzog*
	fraud_Stroman, Hudson and Erdman	*fraud_Connelly, Reichert and Fritsch*
	fraud_Wisozk and Sons	*fraud_Ratke and Sons*
	fraud_Hudson − Grady	*fraud_Hills − Boyer*
	fraud_StiedemannInc	*fraud_SpinkaInc*
	fraud_Ledner − Pfannerstill	*fraud_Huels − Nolan*
	fraud_CummingsLLC	*fraud_MurrayLtd*
	fraud_Stark − Batz	*fraud_MarksInc*
	fraud_Cremin, Hamill and Reichel	*fraud_GutmannLtd*
	fraud_Ziemann − Waters	*fraud_EffertzLLC*
	fraud_Erdman − Kertzmann	*fraud_Jaskolski − Dibbert*
	fraud_Bartoletti − Wunsch	

A.1.4.5 Month-Day Features

TABLE A.8: *month_day* entries in the top predictive features.

month_day	Entries												
$0.5 \leq WoE \leq 1$	26	56	525	49	112	426	331	511	424	516	928	1025	1013
	35	88	925	97	711	811	620	917	824	428	106	1212	1020
	37	101	48	812	211	119	630	722	712	116	313	1128	1010
	32	57	218	430	105	719	523	827	317	831	325	1114	1012
	31	83	913	103	727	914	529	519	912	323	15	1019	1024
	29	63	114	610	113	710	316	54	522	830	124	1016	1211
	47	95	825	120	223	927	322	617	922	420	321	1115	1125
	24	94	619	513	514	118	517	425	310	110	524	1119	
	73	59	813	919	826	102	109	614	512	117	108	1011	
$-1.5 \leq WoE \leq -1$	531	625	422	915	822	814	1210	1218	1022	1215	1225	1116	

A.1.5 Baseline model - Imbalanced Dataset

The baseline models are built using default classifier hyper-parameters and skewed dataset with minimal processing (only converting features into formats which the ensemble models used can interpret). The baseline model's purpose is to contextualise and benchmark the iterative development of the optimised models.

The results of the default XGBoost and imbalanced-data model show that the default model fails to identify the fraudulent transactions in the test dataset. It is heavily biased towards the legitimate transactions as seen by the precision averaging around 37%, recall at 65% and f1-score at 47% for the test dataset as shown in Table A.9. The recall measures the identification of true positives and false negatives and hence can be seen to be measuring well than the precision and f1-score. The Cohen's kappa and the gini are significantly lower for the test dataset than the training dataset, 0.75 and 75% as compared to 0.89 and 89% respectively.

TABLE A.9: Evaluation of the default XGBoost classifier on imbalanced dataset.

	Confusion Matrix	Accuracy	Class	Precision	Recall	f1 Score	Cohen's Kappa	Gini Coeff
Train	$\begin{bmatrix} 1288957 & 1243 \\ 212 & 6263 \end{bmatrix}$	99.8%	Class-0	100%	100%	100%	0.89	0.83
			Class-1	83%	97%	90%		
Test	$\begin{bmatrix} 553400 & 748 \\ 174 & 1397 \end{bmatrix}$	99.8%	Class-0	100%	100%	100%	0.75	0.65
			Class-1	37%	65%	47%		

The KS statistic plot in Figure A.12 shows the separation power of the default XGBoost model for the training dataset. The figure evidently shows that the model is not clearly discriminating the two classes. These results are a clear indication that there is a need to balance the dataset to train

and achieve a more discriminative model. The separation power reduces drastically as more data is classified as seen by the shape of the class 1 (orange line) graph.

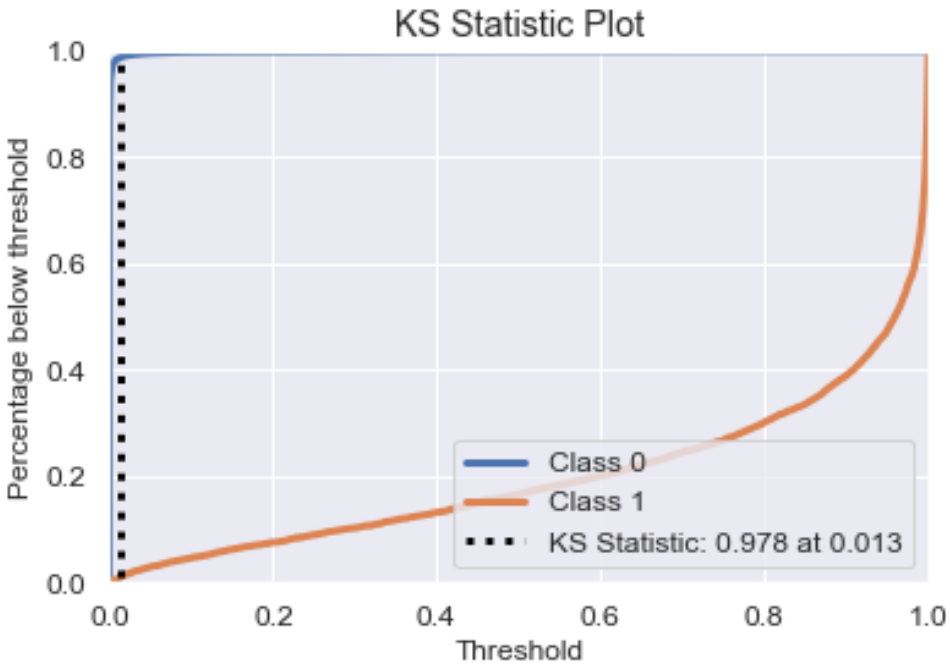

FIGURE A.12: KS Statistic plot for the default XGBoost classifier - Training.

The results of the default iForest model shows how the model fails to train on imbalanced dataset as shown in Table A.10. The model performs poorly while learning the properties of the minority class, class 1. The model cannot learn enough from the minority class to be able to classify the minority class.

TABLE A.10: Evaluation of the default iForest classifier on imbalanced dataset.

	Confusion Matrix	Accuracy	Class	Precision	Recall	f1 Score	Cohen Kappa	Gini Coeff
Train	[1216691 4776 ; 72478 2730]	94.1%	Class-0	94%	100%	97%	5.6	0.03
			Class-1	36%	4%	7%		

A.2 Appendix-A Conclusion

The results of the modelling process are shown and a detailed summary of the development process has been detailed. The definition of the original entries of features in the top predictors is detailed. The baseline models results using default classifier hyper-parameters and skewed dataset with minimal processing were also shown.

Bibliography

[1] O. Adepoju, J. Wosowei, and S. Lawte. "Comparative Evaluation of Credit Card Fraud Detection Using Machine Learning Techniques". In: *Global Conference for Advancement in Technology (GCAT), IEEE* (2019).

[2] M. Aryuni, E.D Madyatmadja, and E. Miranda. "Customer Segmentation in XYZ Bank using K-Means and K-Medoids Clustering". In: *International Conference on Information Management and Technology, IEEE* (2018).

[3] J. Awoyemi, A. Adetunmbi, and S. Oluwadare. "Credit card fraud detection using Machine Learning Techniques: A Comparative Analysis". In: *IEEE* (2017).

[4] M. Bland. "Cohen's kappa". In: *University of York Department of Health Sciences* (2008). URL: https://www-users.york.ac.uk/~mb55/msc/clinimet/week4/kappa_text.pdf.

[5] R. Bolton and D. Hand. "Statistical Fraud Detection: A Review". In: *Statistical Science, IEEE* 17 (2002).

[6] L. Breiman. "Random Forests". In: *Kluwer Academic,* (2001), pp. 5–32.

[7] C3.ai. "What is the F1 Score?" In: *C3 AI* (2022). URL: https://c3.ai/glossary/data-science/f1-score/.

[8] T. Celik. "Adaptive Computation and Machine Learning Supervised Learning: Part II". In: *School of Computer Science and Applied Maths, University of the Witwatersrand* ,Unpublished (2020), pp. 1–58.

[9] W. Chagwiza. "Bagging and Boosting Models - Theory and Practice". In: *Standard Bank / EOH Information Services, Unpublished* (2019), pp. 1–12.

[10] P. Charoen-Ung and P. Mittrapiyanuruk. "Sugarcane Yield Grade Prediction using Random Forest and Gradient Boosting Tree Techniques". In: *IEEE 15th International Joint Conference on Computer Science and Software Engineering (JCSSE)* (2018).

[11] T. Chen and C. Guestrin. "XGBoost: A Scalable Tree Boosting System". In: *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining* (2016), pp. 785–794.

[12] G. Christodoulakis and S.Satchell. "The Analytics of Risk Model Validation". In: *A volume in Quantitative Finance* (2008). URL: https://www.sciencedirect.com/book/9780750681582/the-analytics-of-risk-model-validation.

[13] Experian. "Exploring the links between customer recognition, convenience, trust and fraud risk". In: *The 2018 Global Fraud and Identity Report* (2018). URL: https://www.experianplc.com/media/3415/global-fraud-and-identity-report-2018-final.pdf.

[14] Z. Ferdousi and A. Maeda. "Unsupervised Outlier Detection in Time Series Data". In: *Proceedings of the 22nd International Conference on Data Engineering Workshops* (2006).

[15] D. Ge, S. Chang, and J. Gu. "Credit Card Fraud Detection using Lightgbm Model". In: *International Conference on E-Commerce and Internet Technology IEEE* (2020).

[16] I. Good. "Weight of Evidence: A Brief Survey. Bayesian Statistics." In: *Bayesian Statistic 2, Available Online* 2 (1985), pp. 249–270. URL: http://www.swrcb.ca.gov/water_issues/programs/tmdl/docs/303d_policydocs/207.pdf.

[17] S. Goswami. "Class Imbalance, SMOTE, borderline SMOTE, ADASYN - Class Imbalance can put our algorithm off balance". In: *Towards Data science* (2020). URL: https://towardsdatascience.com/class-imbalance-smote-borderline-smote-adasyn-6e36c78d804.

[18] V.T. Gowda. "Credit Card Fraud detection using Supervised and Unsupervised Learning". In: *AIRCC Punlishing Corporation* (2021). URL: https://aircconline.com/csit/papers/vol11/csit111107.pdf.

[19] D. Hand and N Adams. "Plastic card fraud detection using peer group analysis". In: *Advances in Data Analysis and Classification* (2008). URL: https://www.researchgate.net/publication/220490538_Plastic_card_fraud_detection_using_peer_group_analysis.

[20] T. Hastie, R. Tibshirani, and J. Friedman. "The elements of statistical learning: data mining, inference and prediction". In: *Springer Series in Statistics* 2 (2001), pp. 352–370.

[21] J. Hurwitz and D. Kirsch. "Machine Learning For Dummies, IBM Limited Edition". In: *John Wiley & Sons, Inc* (2018). URL: https://www.ibm.com/downloads/cas/GB8ZMQZ3.

[22] K. Jolly. "Machine Learning with Scikit-learn Quick Start Guide: Classification, Regression, and Clustering Techniques in Python". In: *Packt Publishing - Available online on: Oreilly Media* (2018). URL: https://learning.oreilly.com/library/view/machine-learning-with/9781789343700/95a3272b-7a0f-429d-a992-2aa582df605e.xhtml.

[23] C. Ju and N. Wang. "Research on Credit Card Fraud Detection Model Based on Similar Coefficient Sum". In: *First International Workshop on Database Technology and Applications, IEEE* (2009).

[24] G. Ke, Q. Meng, and T. Finley. "LightGBM: A highly efficient gradient boosting decision tree". In: *31st Conference on Neural Information Processing Systems* (2017).

[25] F. Liu, K.Ting, and Z. Zhou. "Isolation Forest". In: *Eighth IEEE International Conference on Data Mining* (2008).

[26] A. Mishra and C. Ghorpade. "Credit Card Fraud Detection on the Skewed Data Using Various Classification and Ensemble Techniques". In: *IEEE International Students' Conference on Electrical, Electronics and Computer Science* (2018).

[27] T. Mitchell. "Machine learning - Artificial Neural Networks". In: *Mcgraw-Hill Science/Eng/Math* (Mar. 1997), Chapter 4.

[28] A. Natekin and A. Knoll. "Gradient boosting machines - a tutorial". In: *Methods article - Frontiers in Nurorobotics* (Dec. 2013), Volume 7.

[29] X. Niu, L. Wang, and X. Yang. "A comparison study of credit card fraud detection: Supervised versus unsupervised". In: *arXiv* (2019). URL: https://arxiv.org/pdf/1904.10604.pdf.

[30] C. O'Neil. "Weapons of Math Destruction". In: *Harlow, England: Penguin Books* (2017). URL: https://edisciplinas.usp.br/pluginfile.php/4605464/mod_resource/content/1/%28FFLCH%29%20LIVRO%20Weapons%20of%20Math%20Destruction%20-%20Cathy%20ONeal.pdf.

[31] C. Pan, J. Tan, and D. Feng. "Identification of Power Quality Disturbance Sources Using Gradient Boosting Decision Tree". In: *IEEE* (2018).

[32] M. Pathak. "Using XGBoost in Python Tutorial". In: *Datacamp* (2018). URL: https://www.datacamp.com/tutorial/xgboost-in-python.

[33] G. Press. "Cleaning Big Data: Most Time-Consuming, Least Enjoyable Data Science Task, Survey Says". In: *Forbes* (2016). URL: https://www.forbes.com/sites/gilpress/2016/03/23/data-preparation-most-time-consuming-least-enjoyable-data-science-task-survey-says/?sh=88feadf6f637.

[34] J. Raymaekers, W. Verbeke, and T. Verdonck. "Weight-of-evidence 2.0 with shrinkage and spline-binning". In: *arXiv Computer Science* (2021). URL: arXiv:2101.01494.

[35] E. Rencberoglu. "Fundamental Techniques of Feature Engineering for Machine Learning". In: *Towards Data Science* (2019). URL: https://towardsdatascience.com/feature-engineering-for-machine-learning-3a5e293a5114.

[36] A .Rogozhnikov. "Gradient Boosting explained [demonstration]". In: *Towards Data Science* (2016). URL: http://arogozhnikov.github.io/2016/06/24/gradient_boosting_explained.html.

[37] G. Rushin, C. Stancil, and M. Sun. "Horse Race Analysis in Credit Card Fraud-Deep Learning, Logistic Regression and Gradient Boosted Tree". In: *IEEE* (2017).

[38] Y. Sahin and E. Duman. "Detecting Credit Card fraud by ANN and Logistic Regression". In: *IEEE* (2011).

[39] P. Schober and C. Boer. "Correlation Coefficients: Appropriate Use and Interpretation". In: *Anesthesia and Analgesia* (2018). URL: https://www.researchgate.net/publication/323388613.

[40] Scikit-learn. "sklearn.decomposition.PCA". In: (2007-2022). URL: https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html.

[41] C. Seiffert, T.M. Khoshgoftaar, and J.V. Hulse. "A Comparative Study of Data Sampling and Cost Sensitive Learning". In: *IEEE International Conference on Data Mining Workshops* (2008).

[42] D. Sharma. "Evidence in Favor of Weight of Evidence and Binning Transformations for Predictive Modeling". In: *SSRN* (2011). URL: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=1925510.

[43] V. Sharmila, K. Kumar, and R. Sundaram. "Credit Card Fraud Detection Using Anomaly Techniques". In: *IEEE Xplore* (2019).

[44] J. Shlens. "A Tutorial on Principal Component Analysis". In: *Institute for Nonlinear Science* (2005). URL: https://www.cs.cmu.edu/~elaw/papers/pca.pdf.

[45] M. Soley-Bori. "Dealing with missing data: Key assumptions and methods for applied analysis". In: *Boston University - School of Public Health* (2013). URL: https://www.bu.edu/sph/files/2014/05/Marina-tech-report.pdf.

[46] Y. Song, X. Liu, and J. Tian. "Isolation Forest based Detection for False Data Attacks in Power Systems". In: *IEEE PES Innovative Smart Grid Technologies Asia* (2019).

[47] A. Surbhi and S. Kumar. "Fraud Detection During Money Transaction and Prevention". In: *2nd International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT), IEEE* (2019).

[48] K. Torizuka, H. Oi, and F. Saitoh. "Benefit Segmentation of Online Customer Reviews Using Random Forest". In: *IEEE, Industrial Engineering and Engineering Management (IEEM)* (2018).

[49] L. Tran, L. Fan, and C. Shahabi. "Distance-based Outlier Detection in Data Streams". In: *Proceedings of the VLDB Endowment, IEEE* 9 (2016).

[50] O. Vynokurova, D. Peleshko, and O. Bondarenko. "Hybrid Machine Learning System for Solving Fraud Detection Tasks". In: *IEEE Third International Conference on Data Stream Mining and Processing* (Aug. 2020).

[51] K. Wang, J. Wan, and G. Li. "A Hybrid Algorithm-Level Ensemble Model for Imbalanced Credit Default Prediction in the Energy Industry". In: *Energies* (2022). URL: https://doi.org/10.3390/en15145206.

[52] H. Wickham. "Tidy data". In: *Journal of statistical software* VV, Issue II (2016). URL: https://vita.had.co.nz/papers/tidy-data.pdf.

[53] W.Yu and N. Wang. "Research on Credit Card Fraud Detection Model Based on Distance Sum". In: *International Joint Conference on Artificial Intelligence* (2009).

[54] H. Xu and H. Wang. "Identifying diseases that cause psychological trauma and social avoidance by Xgboost". In: *IEEE International Conference on Bioinformatics and Biomedicine (BIBM)* (2019).

[55] S. Xuan, G. Liu, and Z. Li. "Random Forest for Credit Card Fraud Detection". In: *IEEE* (2018).

[56] T. Yiu. "Understanding Random Forest - How the Algorithm Works and Why it Is So Effective". In: *Towards Data Science* (2019). URL: https://towardsdatascience.com/understanding-random-forest-58381e0602d2.

[57] C. Zhen, J. Duan, and L. Kang. "A hybrid data-level ensemble to enable learning from highly imbalanced dataset". In: *Elsevier - information Sciences* (2021). URL: https://doi.org/10.1016/j.ins.2020.12.023.