Cross Channel Fraud Detection Framework in Financial Services using Recurrent Neural Networks

This dissertatin is submitted for the degree of

Doctor of Philosophy

BY

Yogesh Patel

Date: 17/09/2019

School of Computing and Digital Media

London Metropolitan University

166-220 Holloway Road, London, N7 8DB

Acknowledgements

I would like to take a moment to say thanks to many people, without them it would not have been possible to complete my journey. I would first like to thank my supervisor, Prof. Karim Ouazzane who provided unprecedented time, support and guidance, throughout this tenure. Always motivated me when things got hard, always pointed me in the right direction and taught me to never give up even in moments of doubt. I would further like to thank my collaborators Dr. Vassil Vassilev, Dr Jun Li and Thekla Polykarpou, who have also been instrumental with their feedback. This project would not have been as good without your help.

I would like to extend my thanks for my friends Peter Foster, George Walker, Gabriel Dominguez Conde and others who helped me learn some very complicated machine learning models and maths behind them. Whom I spent countless hours debating, brainstorming and experimenting various solutions and having lots of fun along the way. Over the last few years, I have also been privileged to work with many great people at Callsign, who have helped me grow and given me the space to complete my thesis.

Finally, I would also like to thank my family, especially my wife, Shital for been extremely supportive. My children Riana and Darshan to provided core motivation to finish my degree with expediency, so I can spend some more time with them. My mum and my brother deserve a special thank too, for their continued encouragement.

Thank you very much, everyone!

Yogesh Patel

London, September 16, 2019.

I would like to dedicate this thesis to my loving family…

Abstract:

The reliability and performance of real time fraud detection techniques has been a major concern for the financial institutions as traditional fraud detection models couldn't cope with the emerging new and innovative attacks that deceive banks. The problems are further exacerbated with evolving customer behaviour as existing fraud detection models unable to cope with class imbalance problem and longer feedback loop. This thesis looks at the holistic view of fraud detection and proposes a conceptual fraud detection framework that can detect anomalous transaction quickly and accurately, as well as dynamically evolve to maintain the efficiency with minimum input from subject matter expert. The framework is used to analyse Internet Banking (IB) transactions and contextual information to reduce the false positives and improve fraud detection rates. Based on the proposed framework, Long Short-Term Memory (LSTM) based Recurrent Neural Network model for detecting fraud in remote banking is implemented and performance is evaluated against Support Vector Machine (SVM) and Markov models.

The main research element is to model events as state vectors so that sequence-based learning can be applied, followed by a weak classifier to deal with noise. Firstly, the study focuses on Feature Engineering where along raw attributes such as IP Address, Amount and other, two novel features for remote banking fraud are evaluated, i.e., the time spend on a page and the time between page transition. The second focus is on modelling which is performed on an anonymised real-life dataset, provided by a large financial institution in Europe. The results of the modelling demonstrate that given the labelled dataset all models can detect payment fraud with acceptable accuracy.

Various tests proved that the LSTM model achieves a F1 score of 97.7% whereas the SVM and Markov model achieve 93.5% and 95.0% respectively. As the time elapsed, the LSTM model performance significantly improves as the sequence of events became larger. As the dataset increases that time it takes to train traditional models becomes a bottleneck. This proves the hypothesis that the events across banking channels can be modelled as time series data and then sequence-based learners such as Recurrent Neural Network (RNN) can be applied to improve or reduce the False Positive Rate (FPR) and False Negative Rate (FNR).

List of Publications

- *Remote Banking Fraud Detection Framework using Sequence Learners.* **Yogesh Patel, Karim Ouazzane, Vassil Vassilev, Jun Li.** Journal of Internet Banking and Commerce. April 2019. Vol: 24 Issue 1.
- *Keystroke Dynamics using auto encoders.* **Yogesh Patel, Ouazzane Karim, Vassil Vassilev, Ibrahim Faruqi, Geroge Walker.** International Conference on Cyber Security and Protection of Digital Services (IEEE). June 2019.
- *Adaptive business rules framework for workflow management.* **Kanana Ezekiel, Karim Ouazzane, Vassil Vassilev, Yogesh Patel.** Business Process Management Journal. October 2018.

Table of Contents

List of Figures:

Dictionary:

Payment Fraud	Making unauthorised payments completed by a cyber criminal to deceive banks and their customers. For the purpose of this report, this involves intercepting and altering payee details and amounts on Payable Orders via online account takeover as well as compromise of the credit and debit card details.
Channel	A proxy for a bank to interact with its customers. This is typical via the portal, telephone, branch, e-commerce site, the point of sale (POS) and mobile.
Cross Channel Fraud	Cyber criminals making use of multiple channels to defraud a bank. i.e. customer changes the address via bank's portal and calls the bank to re-order a new credit/debit card.
False Positive Rate	The ratio between the number of geneiun events marked as fraud event.
False Negative Rate	The ratio between the numnber of fraud events marked as geneiun events.
I.I.D	Independently and identically distributed variable origin from the same probability distribution as the others but mutually independent to others.
Modus Operandi (MO)	A method that fraudster will use to continuously defraud the banks.
Transaction	The term transaction in this thesis is referred to as banking transaction that involves money movement such as transferring money from one account to

	another or payment of goods. In any other context, the term will be explicitly stated.
Data Model	Data model defines the how data is connected to each other and their relationships. It provides a logical structure of the database.
Neural Network Model	This model will be used to classify events as fraudulent or legal based on the neural network algorithm that has been trained over a set of historical events.
Fraud Model	Generic term used to classify events as fraudulent or legal based on different machine learning algorithms that has been trained. In this report, the term will be often referred to as just model
Customer Behavioural Modelling	Term used to represent a set of customer activities performed on the bank's channels by the customer.

Glossary:

POS	Point of Sale
CNP	Card Not Present
PSD2	Payment Services Directive
UK	United Kingdom
FPR	False Positive Rate
FNR	False Negative Rate
FFA	Financial Fraud Action
ATM	Automated Teller Machine
OTP	One Time Password
KNN	K-Nearest Neighbour
NB	Naïve Bayes
SVM	Support Vector Machine
ANN	Artificial Neural Networks
ML Lib	Machine Learning Library for Big Data
OLTP	Online Transaction Processing
RFM	Recency, Frequency, and Monetary
SMOTE	Synthetic Minority Over-Sampling Technique
PCA	Principle Component Analysis
RIDIT	Statistical scoring method used to analyse ordered qualitative measurements
TRSGM	Transaction Risk Score Generation Method

IJIRCCE	The International Journal of Innovative Research in Computer and Communication Engineering
NN-BP	Neural Network Back Propagation
HMM	Hidden Markov Models
CA	Cluster Analysis
MLP	Multi-Layered Perceptron
EM	Expectation Maximisation
QRT	Questionnaire Responded Transaction
IOT	Internet of Things
AI	Artificial Intelligence
EMV	Euro Pay, MasterCard and Visa
Fraud MO	Fraud Modus Operandi
RNN	Recurrent Neural Network
ROC	Receiver Operating Curve

1.

Introduction

Fraud losses continue to increase year by year, it has been reported that globally businesses are losing an average of 6.8% of their total expenditure as a direct result of fraud. Fraud has always been a major concern for the financial institutions. Fraud in financial services is defined by the deceitful activities of the fraudsters that result in financial profits or other benefits to them, as emphasised in the Oxford dictionary:

"Wrongful or criminal deception intended to result in
a financial or personal gain."

Since, the dawn of the internet, cyber criminals have been increasingly perpetrating financial institution's defences what now seems like a relatively straightforward manner. This has resulted in a more fine-grained definition of fraud provided by Van Valsselear et al. (7) as:

"Fraud is an uncommon, well-considered, imperceptibly concealed, time-evolving and often carefully organised crime which appears in many types of forms."

Examples are, criminals either stole credit cards or compromise merchants' point of sale (POS) devices to carry out fraudulent transactions. Similarly, in online banking, fraudsters compromise; customers' device (personal computer or mobile) and move money out of customers' accounts using the bank's e-commerce sites. These are also referred to as cybercrime, where criminal targets computer, network or devices. Some examples of cybercrimes include fraud and phishing.

The National Institute of Standards and Technology (NIST) provides a flexible way to address cybersecurity by a framework for managing cyber threats (145). The five core functions are defined as:

- Identify: understanding the business context in which any organisations are being defrauded.
- Protect: once a threat such as fraud is identified, provide appropriate measures to protect an organisation to any further damage.

- Detect: provide an ability to detect a threat such as online banking fraud through the use of anomaly detection and event correlation.
- Respond: provides a list of action to be taken when a threat is detected and may include detail analysis, provide mitigations and future improvements.
- Recover: provides a set of activities to bring services back to the normal status that were impacted during a cybersecurity incident, with emphasis on availability and resiliency.

While all of the above core functions are essential parts of effective fraud risk management strategy, this research is primary focused on detection especially on fraud detection function. For example, ability to detect fraud in real-time when fraudsters make unauthorised payments from the customer's account via the bank's channels such as an online portal.

Fraud via an internet and e-commerce contributes to over 30% of overall fraud according to Fraud Action UK (2). The fast development in the payment ecosystem has led to new threats as cyber criminals have established new and innovative attacks across multiple banking channels to deceive banks and their genuine customers. The innovation in digital banking such as Apple Pay and Android Pay coupled with regulatory changes such as Payment Service Directive 2 provides a bigger landscape for the fraudster to operate on (1). In 2018, innovations in machine learning and openness from industry to invest in advanced security stopped more than £1.6 billion of unauthorised fraud. However, fraudsters still successfully managed to steal £1.2 billion.

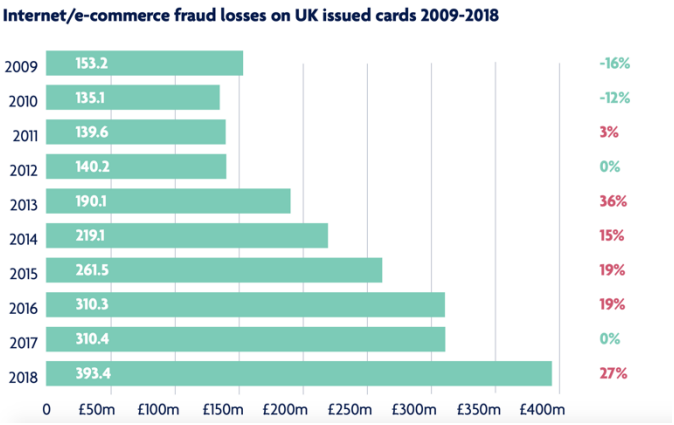


Figure 1.1. Fraud Losses in the UK (2)

As seen in Figure 1.1 above, in 2018; there is an increase of 27% of an internet and e-commerce fraud. Through the use of social engineering tactics, phishing and sophisticated malware, fraudsters were able to compromise customers into revealing their online banking credentials. An example includes a phishing scam where an attacker uses fraudulent e-mail claiming to be from a bank in order to trick customers into disclosing confidential information such as bank account information. A more sophisticated fraud modus operandi (MO) will include an impersonation scam where fraudster often call their victim pretending to be a bank and claiming there has been a suspicious activity on their bank account, they would then claim to provide a fix by installing a software. They will then install a malware on the computer, which will be used to steal account details as well as security credentials. This type of fraud is referred to as Remote Access Takeover (RAT). If that fails, they will claim that account details need to be re-verified or money needs to be transferred to a “safe” account. These stolen details are then used to then make unauthorised payment.

Fraud management is getting a lot more attention both in the financial services as well as academia. White papers published by some of the commercial fraud detection companies highlights the existing methods are primarily rule-based and channel specific, with some making use of a neural network for profiling customer behaviour (3) (4). They are based on first detecting the fraud technique and then writing appropriate rules for subsequently mitigating that fraud technique. They are considered as inadequate and ineffective against zero-day attacks as they do not proactively identify fraudulent transactions, thus causing bad customer experience. Most researchers in academia have researched on increasing the accuracy of the fraud detection through multiple techniques. A lot of research has been

already carried out using machine learning algorithm to detect fraud, but almost all of them have been channel specific (5) and primarily focused on credit card fraud detection. Several classifiers such as linear regression, KNN, C5.0, NB, SVM and ANN among other have been used to train datasets in identifying fraud and have done reasonably well on the available data (6). While the existing techniques of machine learning provide insight into the difficulties involved in various learning environments and they have an ability to generalise whenever it is required to classify an instance, they do not provide a holistic approach to fraud detection. Also, they suffer from low detection accuracy and high false positives especially when fraudsters change their tactics and the emergence of the new fraud MOs are so frequent (7). Fraud expertise is required to handcraft the data and retraining of the fraud models which are often slow and full of errors. Furthermore, different types of anti-fraud measures such as management information (MI) reports are being used to prevent attacks such as Phishing (i.e. Anti-phishing working group). However, these measures lack the real-time element and lack malware protection against the growing number of phishing websites.

This research is focused on the possibility to create a novel real-time fraud detection framework to detect social engineering and malware fraud. The intent is to make use of the advanced machine learning techniques to reduce the false positive rate (FPR) and false negative rate (FNR) on online banking transactions. The overall summary diagram is included in Figure 1.4 that outlines the key research findings. Such as making use of point estimates over the probability distribution function where there is a huge variance in dataset and a need to quickly generalise model from observed data. An example of such is to consider events across multiple banking channels that can be modelled as time series data for an individual customer and just based on few samples applying machine learning algorithms that can model for time series data as sequences such as Recurrent Neural Network (RNN) can be evaluated (8).

1.1. Research Problem

The key research question which presents the scientific relevance of this investigation is to develop innovate framework based on the deep learning and machine learning techniques. Using point estimates over raw dataset to improve on sequence learning techniques to remove any uncertainty in classifying an event as either fraud or non-fraud in an online

banking context for financial institutions. Furthermore, to investigate and learn about the customer's behaviour patterns to make inference about the social engineering or malware attempt from a fraudster.

This study will address the research questions presented below:

- Can events across multiple channels be considered as time-series events in order to build a classifier that can detect inherent anomaly regardless of the skewness, noise in the data? (See Section 3.3.1)
- How to reduce uncertainty of the model prediction by reducing false negative rate when there are only a few labelled data, or the observed labels are noisy as a result of manual measurement imprecision during labelling? (See Section 4.2)
- Can a correct use of model and technique be used (i.e. sequence-based machine learning techniques such as HMM or RNN) to represent customer behaviour as they interact with the online channel? (See Section 4.3)
- Can a model structure be able to generalise to new abnormal behaviour not seen during training? Thus, using point estimates to model deterministic function that contains uncertainty information in order to reduce false negative rate. (See Section 5.2.1)

Initial statistical analysis on the dataset of known frauds indicated that event correlation analysis could potentially reveal patterns of activity within an internet banking session, which is a strong indicator of fraud.

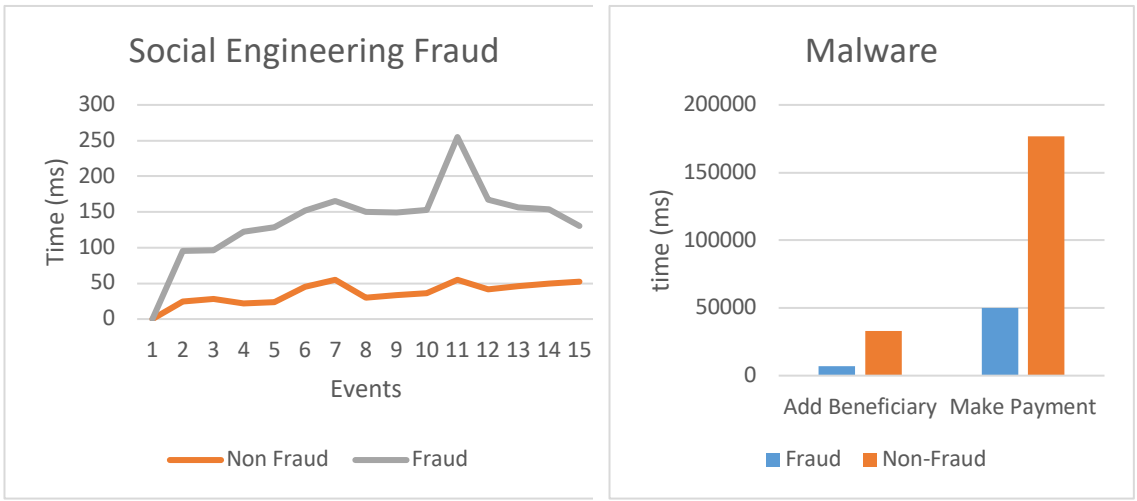


Figure 1.2. Example of Social Engineering and Malware Fraud in Real Dataset (expanded in Section 4.2.1)

Figure 1.2 above show social engineering journeys takes longer to complete than genuine journeys, this could be an example when customer may be on the phone to fraudster. Similarly, malware journeys appear to be quicker than genuine journeys when it comes to adding a new payeeand then making a payment to that beneficiary, which is may be indicator that a Botnet might be automating the journeys.

The outcome of this research will be used to provide better protection to the banking customer and improve the security posture of the banking security controls.

1.2. Aims and Objectives of this Research

The aim of this research is to develop cross channel fraud detection framework, that will help advance the field of payments. This will be achieved by creating a novel framework where the cross-pollination of transaction types and contextual information can be used to reduce the false positive and improve the detection rate. One will focus on the development of fraud framework that can obtain confidence using techniques and theoretical foundations of deep learning and machine learning techniques. Ensembling of these techniques will allow us to improve the fraud detection rate that was not possible until recently. The effectiveness of the fraud detection system is determined by alerts it generates, its accuracy of detecting fraudulent events and speed at which it can train model to capture new threats. Our research in working with several leading UK banks and showed that maintaining the accurate system, generating/working on alerts and quickly responding to new fraud MOs requires extensive efforts across multi-disciplined fraud team. The value chain diagram below outlines the general flow.

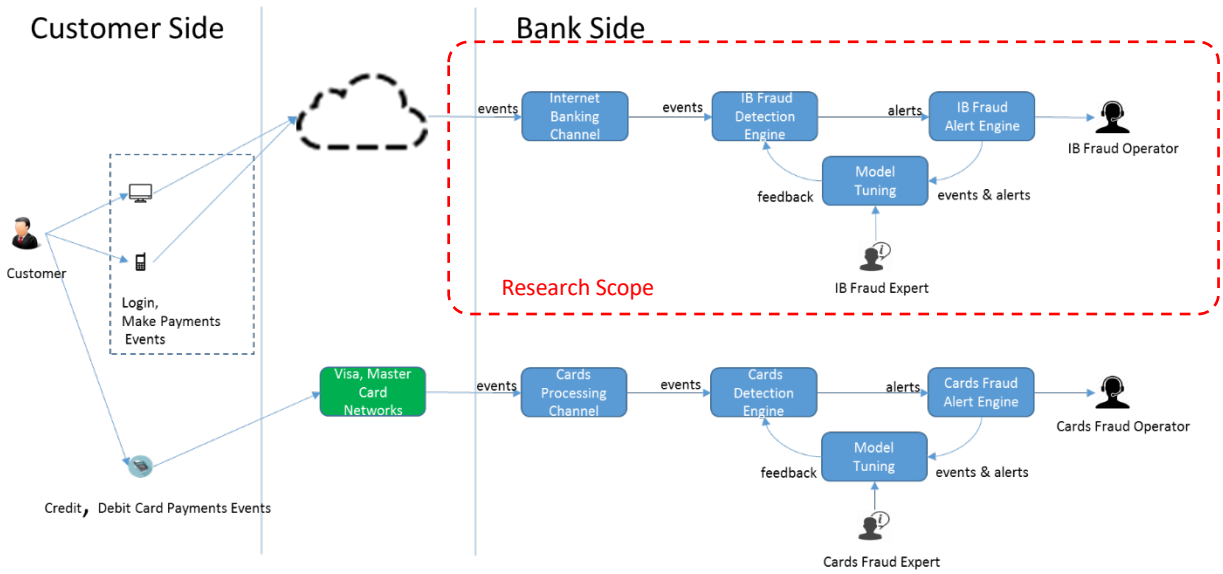


Figure 1.3. Value Chain Diagram for Fraud Detection (This is further expanded in Chapter 3)

Figure 1.3 describes fragmented soloed real-time fraud detecting systems split by channel and reliant on the manual intervention of the fraud expert to update the detecting engine by tuning the fraud rules. This is often referred to as fraud model tuning. Both the commercial systems and academic research tends to focus on fraud model tuning and fraud detection aspects for improving false positive rate based on profiling customer behaviour. The current research in the space showed neural network based solutions are the de-facto standards for detecting fraud (9) (10) (11). The above value chain diagram is for illustrative purpose only to show fraud detection is difficult, due to complex payments methods and complex user interaction channels. Covering entire ecosystem is outside the scope of this research, this research is focused on fraud detection on the internet banking channel.

Below are some of the reasons the current approaches fail to provide adequate detection:

- ***Evolving Customer Behaviour:*** Normal behaviour of the customer is constantly evolving as the flexibility grows in terms of how the customer manages their money. The fast rate at which the customer behaviour is changing, the traditional Neural Network based models no longer accurately represent the customer activities and their payment behaviour. This is due to the limitation of traditional neural network architecture as they can only deal with fixed length input and assumes data to be identically and independently distributed (i.i.d). (8) This research will work on

outlining a flexible algorithm that can operate over sequences of events and over a longer period of time to accurately model customer behaviour. (i.e. RNN).

- ***Class imbalance problem:*** the ratio between fraudulent transactions and non-fraudulent transactions is not equal, means identifying fraudulent transactions is hard to achieve using default model parameters. An expert is required to configure various parameters and settings. This thesis will investigate the use of synthetic data generating techniques as well as some of the sampling-based techniques (such as cost based sampling) to address this problem. Also, evaluate and fine tune the advanced machine algorithm that will provide optimal fraud model performance.
- ***Cross-Channel Fraud MOs:*** Categorisation of fraudulent events and non-fraudulent events are becoming even more difficult to detect by a human expert as the fraudsters utilising cross-channel tactics. Cross-channel fraud is the new reality as the sophistication of criminal activity rapidly increases. Fraudsters are exploiting the "blind spots" across various channels. Evaluate the cross-pollination of data and looking at the contextual data to improve the model performance.
- ***Longer Feedback Loop:*** fraudulent transaction required to be labelled manually and suspicious transactions required further investigation by an expert, before they can be labelled as actual fraud. This effort takes several days to months before data is available to re-tune the models. However, this is an expensive process and often the contextual information required to label data accurately is not feasible. The systems architecture will provide faster feedback to the fraud models to tag fraud that might have been missed during the real-time evaluation. This will reduce the expensive expert time requirement.

The main objectives of this research are:

- To investigate the current state of research in fraud detection and to identify the main problems, existing approaches and available methods for achieving fraud detection with improved performance. See Chapter 2 for further details.

- To investigation data availability across multiple channels to produce appropriate data sets and criteria that will form to be subject to the analysis. Also, to investigate

techniques to process raw transactional banking dataset for fraud detection. See Chapter 3 for further details.

- To research into the methods for machine learning for analysing of the multidimensional banking data represented as profiles and real-time transaction data. For example, anomaly detection on the customer spends pattern and behaviour could be analysed using techniques such as neural networks. See Chapter 4 for more details.

- To design ensemble technique and elicit the best ensemble classifier. To implement and execute fraud framework that ensembles deep learning and other deterministic models on a suitable platform in a context of fraud detection. Demonstrating that customer behaviour during a banking session can be used to detect social engineering and malware fraud. See Chapter 4 for more details.

- To analyse the use of the constructed fraud model, the efficiency of the selected methods and to create ensemble classifier. Evaluate the efficiency by performing validation, offline and online testing. Validation of the developed framework will be performed using both benchmark data and real-life data; offline testing will be carried out using historical data and then online testing will be performed with real-time data. See Chapter 5 for more details.

1.3. Research Methodology

The methodology of the proposed research combines several methods, which are needed to address the different objectives in an adequate way. Due to the complexity of the solution, the methodology will combine analytical, constructivist and empirical approaches. The analytical approach will mainly be used during the research stage when there is a need for critical analysis and comparison of alternative methods is investigated. The constructivist approach will be used for modelling, formulating building blocks of the framework and constructing software. The empirical approach will be used to compare the predictions with the actual data and to make cross-comparison between the different use-cases.

1.3.1. Analytical Approach

- Data acquisition – collate data from an array of data repository. The data from various sources will need to be collected to build a rich set of customer profiles. Making use of different acquisition functions that can model uncertainty about the unlabelled data points in order to decide on their potential informativeness. Three different categories of data will be collected:
  - Tracking customer interactions - page navigations, device usage.
  - Transactional information - transactions from online banking, credit, and debit card transactions.
  - Customer insight – details about geo-location, date of account opening, amount of outstanding loans.
- Data preparation – developing robust and automated techniques for data cleansing, quality grading, and assurance. This will also involve formalisation of sampling methods to factor in unbalanced nature of the dataset. In particular, the focus will be on using technique such as Synthetic Minority Over-sampling Technique (SMOTE) (105) to under-sample from larger label transactions and oversample from smaller label sample. Also, there will be also techniques developed to periodically refresh data.
- Data and model exploration - investigate the correlation between contextual information and it's relevant to the transaction events. This will provide deeper insight into fraudulent transactions. Furthermore, investigate which machine learning algorithm best fits the needs and build a fraud model around that.
- Analytical approach is further expanded in Section 4.2.

1.3.2. Constructivist Approach

- Model development and optimisation – firstly to obtain an appropriate machine learning toolset that can provide a basic platform to achieve our mentioned objectives. This will include the support for advanced machine learning algorithms such as deep neural networks, recurrent neural network,

a support vector machine and others. It will also provide a means for measuring performance. To build a model to cater for the event data that is non-stationary in nature. The model will continually learn and adapt to new fraud threats as they emerge. Followed by building a framework and run a series of experiments on the data set and compare the performance measure.

- Productionisation- Create a fraud management platform, to meet real-world working conditions. It can aggregate inputs from multiple channels and systems, which ingests information, enriches machine learning models, and provides consolidated reporting in addition to providing a holistic view of fraud. Demonstrate that preserving historical data is useful for quickly training a new model, but it is also important to remove outliers as they can skew the results.

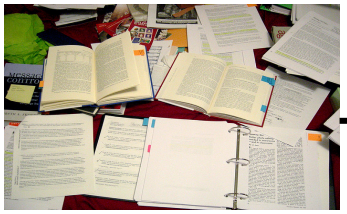
- Constructivist approach is further expanded in Sections 4.3

1.3.3. Empirical Approach

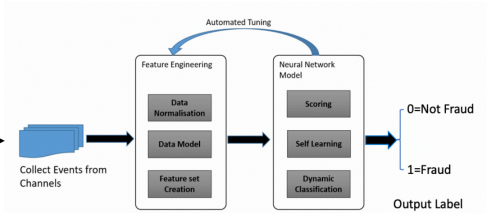
- Analysing and Reporting - both ML algorithm/s and performance of the system will be evaluated with respect to the predictions with the actual data. Various comparison between the different use cases across channel will be analysed and report on the finding will be created.

- Empirical approach is covered in detail in Chapter 5.

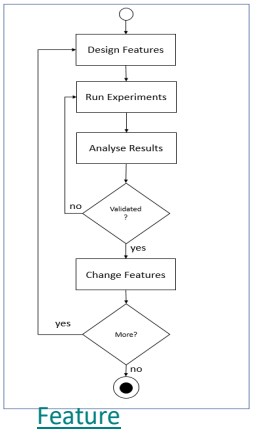
1.4. *Summary Overview Diagram*



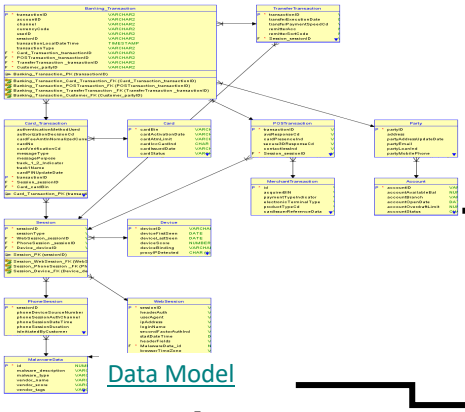
Literature Review



Conceptualisation of the cross-channel fraud framework

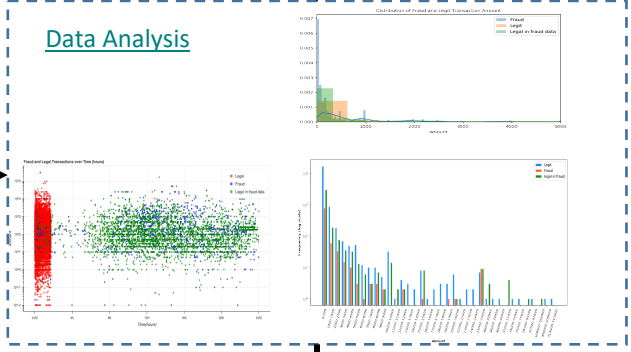


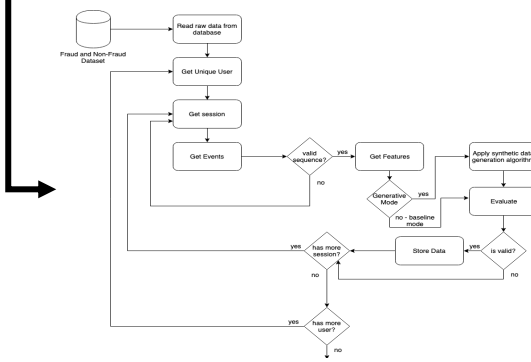
Feature Engineering



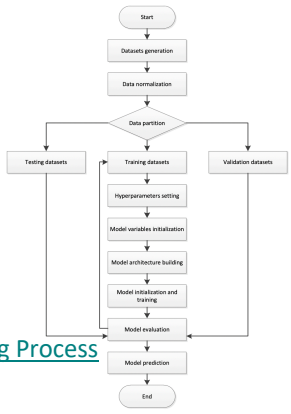
Data Model

Data Analysis

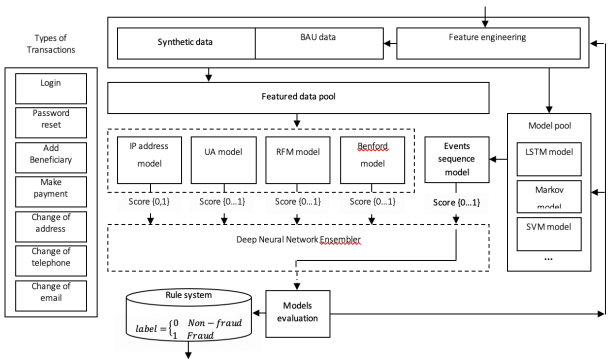




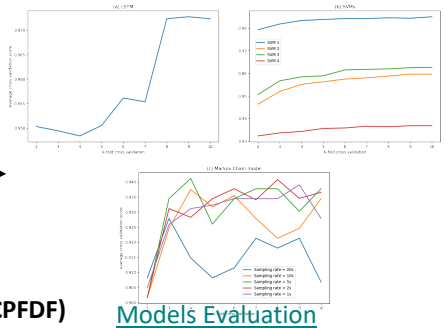
Synthetic Data Generation



Modelling Process



Cross Channel Fraud Detection System



Models Evaluation

Figure 1.4 - Cross Channel Payment Fraud Detection Framework (CCPFDF)

1.5. The Contributions to the Knowledge

The research conceptualises a cross channel fraud detection system that allows events to be modelled as state vectors and applied to learning algorithm, followed by a weak classifier to deal with noise in labelled data for training. It detects and discovers a possible hidden structure and regularity patterns associated with transactional and contextual data when modelled as time series of variable length. More specifically, the research contributions include:

- The intelligent payment fraud detection process by developing sequence learners based on real-life bank transactions.
- Comprehensive uncertainty analysis, comparative analysis and evaluation of the sequence learners
- Conceptualisation of fraud detection framework with multi-components and stages to facilitate the payment fraud detection process.
- A logical data model for cross channel fraud detection system that can cater for transactional data from different channel, namely remote banking and cards.
- Novel approach in synthetic data generation using Generative Adversarial Network (GAN) network.
- LSTM, SVM and Markov Chain models development for online fraud detection using feature engineering combining contextual and temporal data based on real-life bank transactions.
- Model evaluation technique to verify model performance and limitations, and justify the model results based on data, statistical and modelling techniques.

1.6. Structure of the Report

To report the findings of the research in detail, this document is organized as follows:

Chapter 2: Literature Survey

In this chapter, various research papers have been reviewed and studied in the context of the fraud detection domain.

The nature of the existing researches, methodologies and proposed solutions to problems closely related to machine learning models such as supervised and unsupervised learning are studied.

Chapter 3: Conceptualisation of fraud detection framework

This chapter forms the core of the thesis, provides details about the conceptual framework for cross channel fraud detection. The core focused been on feature engineering where normalisation and data model are discussed. Followed by a neural network model that provides insights into classification and self-learning capabilities.

Chapter 4: Model Implementation

Based on the conceptual framework discussed in Chapter 3, this chapter provides systems architecture of the framework and provides implementation of the Support Vector Machine, Long Short-Term Memory and Markov Models. Real banking transactional data is researched, and detail analysis is provided in exploratory analysis section.

Chapter 5: Evaluation

The chapter adopts a comprehensive approach to evaluate and compare the performances of the models based on well-defined indicators and datasets. The hyperparameters used by various models and its impact on the models are described in this chapter.

Chapter 6: Conclusion and Future Work

This final chapter provides conclusion on the thesis, describes what has been achieved, recaps on the research contribution and recommendations for the future work.

2.

Background and Literature Survey

This chapter discusses the various studies spanning across various researches carried out on fraud detection and related works. Research literature review was conducted by firstly collecting the literature based on research objectives and questions. Key words such as Financial Services; Payment Fraud Detection System (PFDS); Recurrent Neural Network (RNN); Long Short-Term Memory (LSTM); Support Vector Machines (SVM) where used to search the well-known databases of journals and articles. Articles and papers were selected the relevant literature that was pertinent to the topic of this thesis by looking at how an author has addressed the research question, key concepts, theories, methods and models used. This was followed by the results and evaluation of the results. Accordingly, it will account for the definitions of concepts and issues that affect the detection of fraudulent activities using different techniques and approaches by examining their accuracy and limitations. The chapter starts by presenting an introduction to fraud in the banking sector and latest emerging threats. Subsequently, fraud detection techniques such as classical rule-based expert systems, supervised and unsupervised will be studied. The chapter concludes with outlining the direction of this research.

2.1. Fraud in the banking sector

The latest Nelson report (12) shows the global fraud loss of $21.84 billion for the year ending 2015. That approximately 7 cents for every 100 dollars spent. A well-defined definition of the fraud is given by Van Vlasselaer et al. (7).

> "Fraud is an uncommon, well-considered, imperceptibly concealed, time-evolving and often carefully organised crime which appears in many types of forms."

When looking at the banking sector viewpoint, the fraud is uncommon in a sense that not many people commit fraud and it's a rare occurrence hence hard to detect. Fraud detection problem has often been dubbed as "needle in a haystack problem". Fraud is well-considered, well concealed and carefully organised, fraud is no longer been done by "script kids" but through a well-planned set of activities by organised cybercrime gangs (2). Fraud is time-

evolving as fraudsters try to defraud banks over multiple channels, exploiting the weak spot. Fraudsters change their tactics faster than the bank could respond. This scenario is often referred to as "cat and mouse game". Fraudsters use a wide range of approaches and patterns using cross-channel techniques and hence fraud appears in many types of forms. A list below provides an overview of some of the fraud in the banking sector.

Action Fraud UK defines fraud as (2):

> "Financial fraud includes 1st and 3rd party fraud on all core banking products/services (including credit and charge cards, current accounts and debit cards, savings accounts, cheques, overdrafts and loans): channels (including point of sale, remote purchases, online/telephone banking, branch counter) and customers (personal and business)."

The Table 2.1 below provides an overview of some of the fraud in the banking sector, but not limited to (A well-defined list can be found at (13)):

Fraud Type	**Description**
Credit card/debit card fraud	As defined by action fraud police UK, credit/debit card fraud is also referred to as plastic fraud that involves compromising of any personal information that is on the credit and debit cards that can be used to commit fraud. Fraudsters might use this information to purchase goods via card present or card not present channels (14). The subtypes under the plastic fraud are counterfeit card, online purchase (Cared Not Present), ID theft and lost & stolen card.
Insurance Fraud	There are many subtypes under the insurance fraud namely home insurance, car insurance, medical insurance and applicable to both the consumer and the seller. For example, the seller may include selling policies from non-existent companies to earn commissions. Consumer fraud includes falsified medical history to lower the premium or faked damage (for example, staged accident).
Money laundering	Deceitfully submitting a large sum of money that are obtained from or used for crimes such as terrorist activities, drug trafficking and other

	serious financial crimes. One of the popular subtypes under money laundering is the mule account creation.
Identity Theft	Unlawfully gaining important personal information in order to take over another person's identity. The identity theft could result in fraudster applying for a credit/debit card pretending to be a legitimate customer and carrying out card fraud.
Internet banking fraud / remote banking fraud	Remote banking fraud is considered to be theft committed using online technology to illegally remove money from a bank account or transfer it to a different bank account. Remote banking fraud occurs via the following channel: internet banking, telephone banking, and mobile banking.
Cheque fraud	There are mainly three subtypes of cheque fraud namely: counterfeit, forged and fraudulently altered. • Counterfeit: printed on a fake paper to look like the genuine cheque • Forged: genuine cheque stolen from the customer and their signature are forged to withdraw the funds. • Fraudulently altered: a genuine cheque that has been altered by a fraudster before it is paid.

Table 2.1 Types of Fraud in the banking sector

Various papers and books have been written that describe the motivation and psychological reasons behind committing fraud. One of the most popular books (15) by Donald Cressey where he developed a model for the occupational offenders referred to as the "fraud triangle" as shown in Figure 2.1, that provides inside into why people commit fraud.

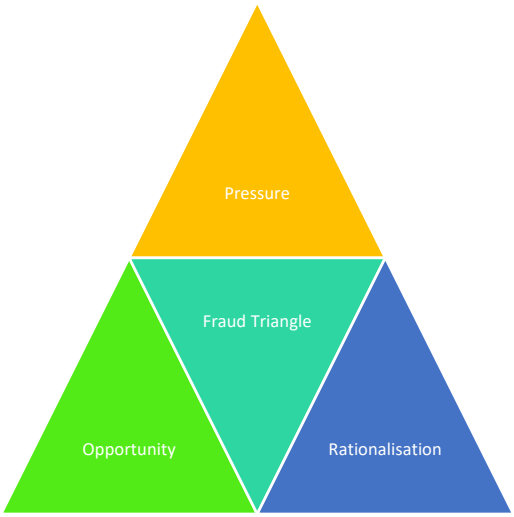


Figure 2.1. Fraud Triangle (15)

The original model developed for embezzlement occupational fraud, however, this could easily be extended to cover the fraud in general (7). The triangle describes three elements required for fraud to be attempted they are:

- Pressure: this outlines motivation of committing fraud. An individual is most likely to commit fraud when he or she is under pressure either due to financial, social or any other nature.
- Opportunity: this outlines misuse of trust. An individual creates an opportunity either due to weak security controls or in a position of trust. This concern results in a fraud opportunity to exists, fraud to be committed and concealed.
- Rationalisation: this outlines a psychological desire such as personal code of ethics or belief into committing fraud. This is concerns with a view of the fraudsters that they are acting legitimately.

Several theories and approaches have been identified that are based on the above model (16) (17). The common theme is on the basis that a conflict of interest may exist between an employee and employer which may lead to disgruntles amongst the employee(s) and employee (s) rationally commit fraud when the opportunity arsis.

Fraud in a banking sector is a multifaceted phenomenon. As discussed in the latest report by Action Fraud Police UK (2), while most of the fraud losses are due to plastic fraud there is a rise in remote banking fraud. Figure 2.2 below outlines financial fraud losses by type for 2015.

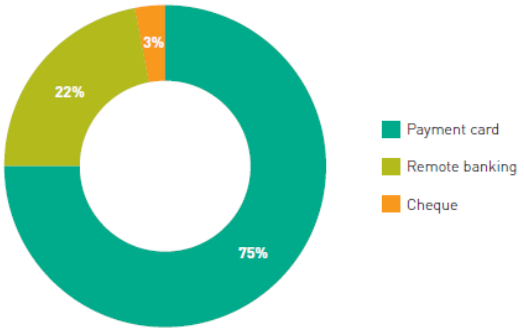


Figure 2.2. Fraud losses by Type (2)

The online banking fraud grew by 23% in the last year, this represents a percentage ratio which is higher than the growth of the plastic fraud. The growth contributes to the latest trends of phishing and social engineering attacks to carry out impersonation and deception scams. Phishing attacks happen when fraudsters pretending to be a genuine company such as a bank to trick their customers into disclosing sensitive information such as login credentials by redirecting customers to a fake website. Furthermore, plastic fraud is also on the increase for card presents and card not presents scams through skimming and lost and stolen cards (18).

The above evidential figures and trends are rough estimates however they do indicate the importance of efficient and effective fraud detection and prevention system. The subsequent section describes various research elements into the fraud detection system.

2.2. Rule-Based Expert Systems Approach

Traditionally rule-based expert systems are used for detecting fraudulent transaction (14). These systems are based on the intuition and expertise of the subject matter expert. Typically, this involves detail manual investigation of the suspicious cases, for example when a customer calls the bank to state they are seeing transactions on the bank statement that they don't recognise. Subsequent investigation may lead to new fraud mechanism used by the fraudster. Rules are then written or updated to limit the further fraud exposure.

An example of such rule for credit card fraud detection is shown in Figure 2.3 below:

```
IF:
```

- TRANSACTION AMOUNT is above threshold OR
- Use of PRIMARY ACCOUNT NUMBER (PAN) above the threshold (i.e. more than 3 times in an hour) with the same Merchant OR
- CREDIT CARD has been used with a RESTRICTED Merchant OR
- CREDIT CARD has been used in a RESTRICTED Country

THEN:

- MARK transaction as suspicious AND
- DECLINE transaction AND
- ALERT fraud investigation team

Figure 2.3. Rule-Based Expert System for Credit Card Fraud Detection

In the example above "thresholds" are determined by fraud subject matter expert (19). It is difficult to maintain, manage and implement such rules. It also requires a complex environment as the fraud schemes and techniques are constantly changing and rules need to be updated frequently.

One of the disadvantages of such rule-based system is that fraudsters can easily learn the thresholds as well as the rules by trial and error to quickly devise innovative workarounds. Furthermore, another limitation of the rule-based system lies in the nature on how they are developed based on previously known fraud cases which make it difficult for them to generalise to new emerging fraud patterns (20). This requires deciding when the existing thresholds and rules should be updated, deleted or when the new thresholds and rules should be added.

In the paper published in 1994 by Leonard (21), describes an implementation of a rule-based expert system to detect credit card fraud. The construction of the rule-based model is based on detecting deviation from the norm on the spend patterns using the expert system. They analysed 12,710 accounts of which 578 where accounts labelled as fraud and compared against 3 different model.

Classification results

1. *Naive model*			2. *Expert-system model*			3. *Holdout sample*		
	Actual nonfraud	*Actual fraud*		*Actual nonfraud*	*Actual fraud*		*Actual nonfraud*	*Actual fraud*
CNF	12,132	578	CNF	11,284	197	CNF	13,666	393
CF	0	0	CF	848	381	CF	990	951
Total	12,132	578	Total	12,132	578	Total	14,656	1344
Percentage correct: 95·45			Percentage correct: 91·78			Percentage correct: 91·35		
Cost: (0 + 20 × 578) *A* = 11,560 *A*			Cost: (848 + 20 × 197) *A* = 4788 *A*			Cost: (990 + 20 × 393) *A* = 8850 *A*		

CNF: classified nonfraud. CF: classified fraud

Figure 2.4. Classification Results (21)

The results in Figure 2.4 showed naïve model achieved 95.45% detection rate, however, the cost of 11,560 means a very high False Positive Rate of 95.18% meaning it misclassified majority of the legitimate account as fraudulent. In comparison, the expert system model correctly classified 65.92% correctly, providing a very good accuracy rate compare to naïve model. The holdout or the blind test will also have performed using the expert system model which showed the accuracy rate like expert system model test and the fraud classification has also improved slightly to 70.76%.

Several techniques exist to mitigate the need to handcraft the rules and thresholds one such technique makes use of genetic programming to evolve logical rules for detecting credit card fraudulent transactions (22). The system classifies data into 3 fuzzy sets: “LOW”, “MEDIUM” and HIGH for the column. The classification is based on the algorithms such as trapezoidal and arctangent to create fuzzy clusters (22). The idea is to determine which columns are a strong contributor for classification of fraudulent events. It then uses genetic programming for rule generation and rule evaluation process. The framework of such system is presented below:

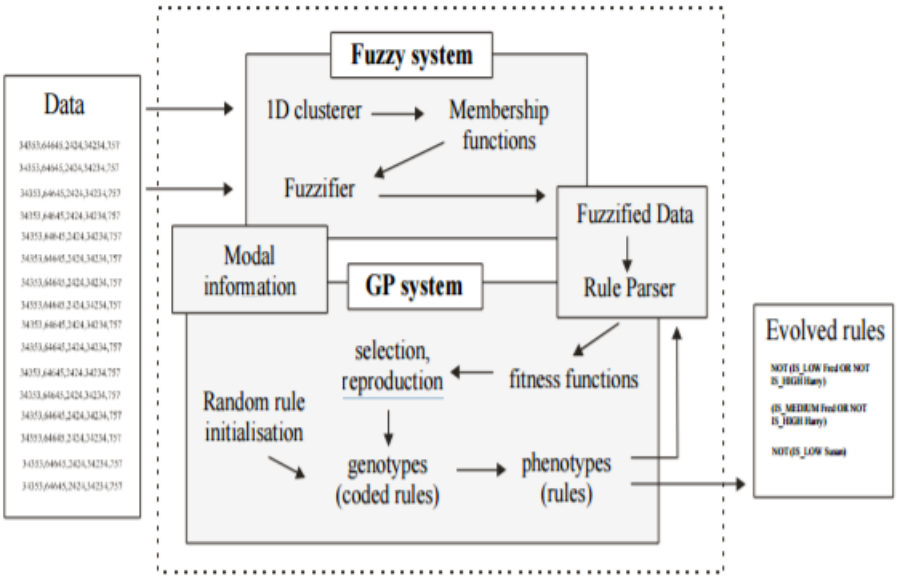


Figure 2.5. Evolutionary Fuzzy System framework (22)

Several experiments performed using such fuzzy system as shown in Figure 2.5, that validates their claims that such system can be used to accurately and intelligibly classify fraudulent transactions with the lower false positive rate. One of the perceived limitations of such system is that it assumes data to be static and structured while this is not the case in most real-life systems. Also, the evolved rules generated performs a basic function, which may not be suitable in a cross-channel scenario.

The rules-based expert system is not just limited to the credit card fraud detection. A lot of research is made on using such system for fraud detection in telecommunications (23). The research paper outlines a need for fraud detection in telecommunications to cater for fraud techniques such as social engineering, fraud due to loopholes in rules, fraud based on new regulation, masquerading as another user and others. The authors examined the rule-based approach and according to the authors of this paper, one of the main drawbacks of the system that is based on the thresholds is that once a threshold is violated it treats customers as fraudsters. This situation is reversed, if the fraudsters are always below the threshold, the violation is never triggered. They proposed an alternative method that is to monitor customer behaviour and compare the characteristics using the adaptive exponential weighted moving average (EWMA). The formula of EWMA is given by:

$$\mathbf{X}_n = \theta * \mathbf{D}_c + (1 - \theta) * \mathbf{X}_p \qquad \text{(eq. 2.1)}$$

Where X_p is the customer signature based on the timestamp n, X_n is a new value, X_p is an old value, D_c is the current information and θ is the non-linear function. They demonstrated the lack of a rule-based system and highlighted a need for a more scalable, robust and efficient fraud detection system that can cater for running large machine learning models rather than the simple rule-based system.

Almost all the research in this section showed that rule-based expert systems are quite efficient at detecting fraud however they are quite laborious in nature and requires manual interventions from a subject matter expert. One of the biggest disadvantages the research highlighted is to try to generalise to a new fraud pattern that has not been seen before. In general, there is a need for a more robust fraud detection system that requires less human expertise and provides faster feedback loop when anomalies in events are detected.

2.3. Un-Supervised Learning Approach

A wide variety of unsupervised techniques exists to detect fraudulent transactions across different fields as a credit card, remote banking, telecommunications and many others. The common theme across all these techniques is to find a deviation from norm (14). The norm can be defined as an average behaviour of a customer or behaviour of an average customer across different time period (7) and then flagging transactions as "fraud" to highlight the deviation away from the norm. From the fraud detection viewpoint, unsupervised techniques try to model the distribution of the normal behavior and flag any observation that shows the greatest deviation from this normal behaviour, this is sometimes referred to as outlier detection (23). This does not require any pervious labelled data and operates by creating clusters of observations that are similar.

2.3.1. Clustering

The goal of the clustering algorithm is to discover groups of similar observations within the data or to determine the distribution of the input space called *density estimation* (24) that from the similarity. In the banking sector, clustering can be used for plastic and non-plastic fraud detection. The density estimation can be derived from customer characteristics such as sociodemographic, behavioural or on account characteristics such as a transaction. Also, density estimation or similarity distance can be achieved through various distance metrics algorithms such as Minkowski distance as defined below (7):

$$D = \left(\sum_{i=1}^{n} |x_i - y_i|^p\right)^{1/p} \quad \text{(eq. 2.2)}$$

This equation can be used to calculate the distance for continuous data where n is the number of input variables, x and y are two observed variables. When p set to 1 the equation becomes Manhattan distance p set to 2 the equation becomes Euclidean distance. There are various different options to calculate the distance such as single-link, complete link, centroid-link and group-average link as shown in Figure 2.6 below (25):

(a) single-link: maximum similarity (b) complete-link: minimum similarity

(c) centroid: average inter-similarity (d) group-average: average of all similarities

Figure 2.6. Similarity between two observed clusters (25)

In general, minimum variance cluster value is calculated using Ward's method and it is shown below:

$$D(C_i, C_j) = \sum_{x \in C_i} (x - c_j)^2 + (x - c_i)^2 + (x - c_{ij})^2 \quad \textbf{(eq. 2.3)}$$

This equation provides distance between the two cluster where c_j, c_i, c_{ji} are the centroid. A number of the cluster can then be decided using the dendrogram.

2.3.1.1. Hierarchical Clustering Methods

Commonly used clustering technique in fraud detection is the hierarchical clustering methods such as divisive or agglomerative hierarchical clustering methods (26).

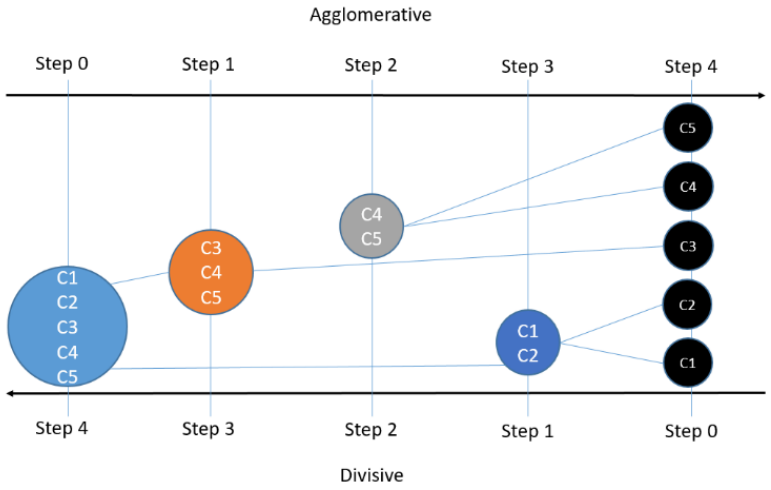


Figure 2.7. Hierarchical Clustering method using Agglomerative and Divisive Methods

In Figure 2.7 above, the agglomerative method starts by treating individual observation and tries to cluster them while the divisive method threats all the observation as a single cluster and they try to break into small sub-clusters. The ideal solution lies somewhere in the middle of the two extremes (7).

A paper published by L. Torgo and E. Lopes (27) makes use of the agglomerative clustering algorithm to detect fraud by creating an outlier ranking. The main motivation behind their paper is that the outliers cannot easily be part of "normal" cluster and it's reflected during the merging process. Their methodology provides a way to optimal use of the inspection resources when forming clusters by creating a ranking. By creating such ranking, the paper claims to efficiently generate the probability of observation been fraudulent. They create synthetic data to prove their point. V. Hanagandi et al. (28) use density-based clustering approach to producing a fraud score for credit card transactions. Their work makes use of radial basis function (RBF) along with a clustering algorithm to classify fraudulent transaction based on the historical data.

2.3.1.2. K-Means Algorithm

Another well-known algorithm to clustering for fraud detection is the K-Means algorithm. This algorithm is used to find hidden information to a large and diverse data by applying a non-linear transformation to the data. By treating the equation 2 as squared Euclidean distance the measure of dissimilarity can be calculated based on a non-linear function of the input data

for a given cluster centre. K-Means algorithm can be summarised as shown in Table 2.1 below (29):

1. Select K observations as the initial centroids (seed).
2. Allocate all the points that are closest to the centroid (using Euclidean distance).
3. Re-compute the centroid of each cluster.
4. Keep repeating steps 2 and 3 until the centroids don't change or a threshold in terms of a number of iterations is reached.

Table 2.1 K-Means pseudo-algorithm

The algorithm aims at minimising the sum of squared error objective function given by (30):

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} \| x_i^{(j)} - c_j \|^2 \qquad \text{(eq. 2.4)}$$

Where $\| x_i^{(j)} - c_j \|^2$ is the squared Euclidean distance between x and j, k is the number of clusters, n is the number of points in a cluster.

The paper (31) produced by P Chougule et al. describes combining genetic algorithms with k means for credit card fraud detection. They create 3 different clusters namely low risk, medium risk and high risk, the transaction will belong to one of the clusters. Once the stable centroids are achieved, a genetic algorithm was applied to improve the efficiency. A level of improvement was shown when applying the genetic algorithm. However, the paper describes the use of biometrics data and in their paper, they fail short of explaining any correlations of data sources such as biometrics in their findings. P Bhati et al. (29) makes combines K-means with Hidden Markov Model (HMM). K-means is used to create clusters and outliers within the clusters are determined by HMM. The results are then validated using Luhn Algorithm. Their paper concluded by highlighting the use of HMM improves their results significantly when compared with using K-Means only. The limitation of their paper is that only single attribute was used (i.e. credit card number), working with more attributes will make the system more complex to implement and evaluate.

2.3.1.3. Adaptive Methods

In an investigation lead by L. Zheng et al (32), a logical graph of behaviour profiles (LGBP) that is a total order-based model to represent the logical relation of attributes of transaction

records is proposed for fraud detection of online shopping. Each attribute in the transaction records are totally ordered and classified by their values. Then based on the LGBP, a path-based transition probability and an information entropy-based diversity coefficient are defined in order to characterize the user's diversity and transaction behaviours. Also, they defined a state transition probability matrix to capture temporal features of transactions and then build a behaviour profiles for each use. Based on these profiles, an incoming transaction was verified as fraud or not. The paper concluded that the proposed model overcome the weakness of Markov chain models since it characterized the diversity of user behaviours.

An adaptive algorithm that can detect credit card fraud as it occurs (real-time) was explored by John B (33). The solution is based on the use of an Artificial Neural Network, Hidden Markov Model and a One-Time Password. After the successful authentication of the One-Time password, the system extracts the user's profile from the bank database, which this profile classified using Artificial Neural Networks. And then using Hidden Markov Model, the financial profile of the user was generated. The combination of the three methods was used for fraud detection. The proposed method was tested using synthetic data and successfully detected and prevented fraud.

Paper produced by R. Bolton and D. Hand (14) presented two techniques known as Break Point Analysis and Peer Group Analysis to detect fraud on the credit card transactions. They analysed intra-account behaviour changes over the certain moving time period (24 transactions for Break Point Analysis and 17 weeks for Peer Group Analysis). Any deviation such as a burst of activity within that period is an indication of deviation from the norm. The comparison was done by ranking the accounts using t-statistic such as t-test and t-score. The behaviour changes are compared to the customer own past activities (for break point) and to the peers (peer group analysis) using distance algorithms such as Mahalonabis Distance and t-score is calculated using the below formula:

$$S_t = \frac{y_n - \overline{X_{n\in k}}}{\sigma} \qquad \text{(eq. 2.5)}$$

Where y_n target account, $\overline{X_{n\in k}}$ observations on the account (i.e. average amount per week or day) and σ is the standard deviation. In their experiments, more superior results were achieved using Peer Group Analysis then to Break Point Analysis. They also highlighted the limitation with the work which generalise well to local anomalies but not towards global

anomalies. Further research showed techniques such as breakpoint analysis are incorporated in the commercial software such as FICO's Falcon where it combines with feedforward Artificial Neural Networks trained using back-propagation training algorithm (34).

2.3.2. Self-Organising Maps (SOM)

Another class of unsupervised learning is Self-Organising Maps which is part of the neural network family. SOM makes use of the neural network architecture to create data clusters in high-dimensions and makes use of competitive learning (35), where only one neural will be active per group (winning neuron). SOM architecture is organised in a lattice structure where the full coordinated system, based on place-coded probability distribution is created around the location of the winning neurons. The motivation behind the SOM originated from how the brain is organised in a way that takes different sensory inputs by different parts of the brain and represented as computational maps in topological order. The most commonly used SOM is known as Kohonen networks as shown in Figure 2.8 below (36):

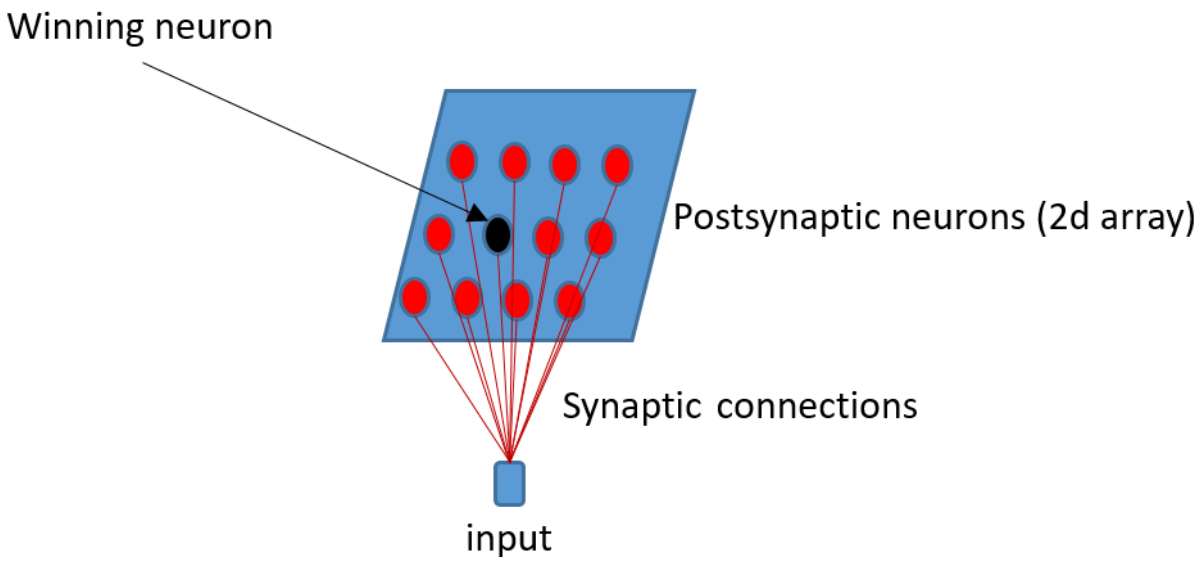


Figure 2.8. Kohonen Model

The process of creating SOM network such as Kohonen model can be summarised as (35):

- Competition: a discriminant function is used as for the basis for competition among the neurons. a neuron with the highest value of the discriminant function is the winning neuron. (i.e. using distance algorithm).
- Cooperation: the neurons around the winning are excited and forms a cluster around that neurons (i.e. using Gaussian distribution).
- Synaptic adaptation: re-evaluate the discriminant function of the all the excited neurons to update their synaptic weights in order to form a pattern based on the input.

Based on the above process, the algorithm can be summarised as shown in Table 2.3 below:

Initialise the weights randomly for each neuron in the lattice Create probabilistic based sample from the input space so that activation pattern can be created Find the best matching neuron at a certain timestamp by minimising the Euclidean distance between the input space and its associated weights by formula $$i(x) = \arg\min_j \|x(n) - w_j\|$$ Where i(x) input space at a certain time stamp, x is the input vector and w is the weight vector The below formula is used to update weights for all the excited neurons $$w_j(n+1) = w_j(n) + \mu(n)h_{j,i(x)}(n)(x(n) - w_j(n))$$ Where $\mu(n)$ is given as the learning rate and $h_{j,i(x)}(n)$ is defined as a cluster function centred around the winning neuron Repeat steps 2-4 until feature map becomes stable

Table 2.3 Types of Fraud in the banking sector

The inherently nonlinear characteristics of SOM make it useful for solving clustering problems within the fraud detection domain.

In a paper (37) from Martin H et al. proposes the use of the self-organizing neural network to attack the fraud detection problem in credit card transactions. The paper suggests the

adoption of SOM over K-means as their experiments highlight the SOM to be a superior at unsupervised cluster analysis the K-means. The paper concludes with outlining that the most important feature of SOM is their natural ability to find dense areas in the input space. A similar approach to using SOM is suggested in the article by Vladimir, et al. (38) where they enabled an automated creation of transaction monitoring system using historical data such as accounts and transactions. Through using these two datasets, customer profiles were created and constantly monitored for any deviation. They assumed that fraudulent transaction will reside in a different space then legal transaction. This can be achieved using SOM topology. D. Olszewski (39) makes use of the SOM for visualisation technique in order to detect fraudulent accounts. The paper utilises SOM in order to visualise account into 2D space for easy of analysis by a human expert and combine this with classification algorithm to improve detection rate. The account clusters are created using SOM's U-Matrix. U-Matrix provides a visualising technique by calculating the average distance between the neuron and its neighbours. This visualising technique highlights dark colours indicating a large distance and can be inferred as cluster boundaries (40). Their architecture consists of two-dimensional lattice and consists of a number of coherent partitions. Their proposed method generalised for telecommunications fraud, credit card fraud and intrusion detection system.

J Hollmen (41) in his thesis on "Probabilistic to fraud detection" for mobile calls. He describes the problem of fraud detection as a pattern recognition problem. Purposes algorithm such as HMM, SOM and EM for detection that is based on the call data of mobile phone subscribers.

2.3.3. Others approach to fraud detection

Relatively very few research papers exist that examines Graph Mining and Link Analysis as the approaches to fraud detection. Few white papers are provided by commercial organisations that outline the effectiveness of using graph mining and link analysis approaches. A white paper published by CGI Inc. (42) describes the use of social network analysis to provide deeper insight into the cross accounts, cross product lines for determining potential mule rings or group of account holders participating in fraudulent activities. They make use of the graph theory (nodes and vertices) to identify deep hidden relationships to identify fraud rings. The measurements are based on density (how deep is the network), centrality (nodes that are close to high activity nodes) and other measures such as cluster co-efficiency. Another

whitepaper by Detica Inc. (43) outlines their platform called NetReveal that performs entity centric profiling to create a unique fingerprint of an entity and then to flag any suspicious activity. The platform can detect hard links such as same address or soft links such as multiple withdrawals from the same ATM. Currently, both of the above commercial products support the investigation of fraud as a post event rather than real-time fraud detection.

Another unique technique of unsupervised neural network-based approach is suggested by Dorronsoro et al. (44) that makes use of non-linear discriminant analysis. The paper segments the account numbers into different geographical regions and then applies a discriminant function to lower the class variance between the different regions created. The paper outlines the achievement of high efficiency and low false positive rates.

A paper (45) by Anshul S et al. uses a Hidden Markov Model to detect credit card fraud. The approach described in this paper uses HMM to profile customer behaviour patterns on credit card spend. It then assigns category such as low, medium and high for each incoming transaction. If the incoming transactions violate a pre-defined threshold value, it marks that transaction as fraudulent. The results showed that HMM producing high false positives but good at detecting an outlier. Future research in HMM is to extend this technique to profile also human behaviour while they are navigating the banking portal's pages.

Fletcher Lu et al. (46) combines Benford's law with reinforcement learning to find new fraud patterns. In their paper, they describe this method as Adaptive Benford's Law. This novel approach works by removing the requirements of "no built-in minimum or max values" for numeric data and applies Benford law for by considering 1^{st}, 1^{st} and 2^{nd} and 1^{st}, 2^{nd} and 3^{rd} digit sequence of a dataset. It extends outlier detection method with Benford law and reinforcement learning component. Deviating from the expected Benford law distribution is an indicator of anomalous behaviour and then applying reinforcement learning to the underlying attributes deducting the anomalous behaviour. This idea can be further expanded by looking to combine supervised and unsupervised fraud detection for technique on a cross-channel data set.

In a research published by Yiyang B et al. (47) an ensemble approach was used for financial fraud detection combining the bagging technique, which can reduce the variance of the classification model by resampling the data set and the boosting technique in order to reduce

the bias of the model. The bagging process divides the data into two classes and then they are fitted into the boosting classifier. Oversampling and under sampling were used to balance the distribution of minority and majority classes for each sub-training data set. In order to reduce the variance and bias of the base learner, Adaboost was integrated into the model for each sub classifier. Then the final output was generated from the voting combination of each sub classifier. The drawback of this approach is the drop of the performance when the classifiers are combined. This is because of the use of the minority class in each base learner and the existence of noise data in the minority class.

In research paper (48) published by Mandeep S and Pravendir S outlined a hybrid approach called TRSGM (Transaction Risk Score Generation Method) to calculate a risk score for each incoming transaction. They combined Bayesian learning, k-means and DBScan algorithms to detect suspicious behaviour based on its deviation from the good pattern. Their results showed good accuracy based on the synthetic data they created. Their approach remains inconclusive for real-world data that is noisy and in large volumes. As suggested by many papers (49) (50) (51) K-means clustering algorithm is a simple and efficient algorithm for credit card fraud detection. In addition, as pointed out in research paper (52), one of the major disadvantages of using Bayesian networks is that it requires significantly higher computational power to produce similar outputs as a threshold-based system with low computational power requirements.

Paper (53) issued in IJIRCCE magazine another use of applying Bayes theorem with DBScan but this time using a heuristic to eliminate the conflict of existing Dempster-Shafer theory. Dempster has given the rule to combine evidence coming from different independent sources. However, Dempster's rule of combination has been criticised as sometimes it gives some illogical results. Therefore, the above-proposed model is susceptible to failures. Dempster-Shafer theory, that allows for combining evidence from multiple and heterogeneous data sources and get to a degree of belief that takes into account all the available evidence.

Paper produced by Jianyun Xu et al. (40) provides a framework in the form of association rule and pattern matching to evaluate anomalies in new transactions. It proposes FP-Tree Based Pattern Matching algorithm and alerts accumulating algorithm to accurately differentiate the

anomaly behaviour from profile users' behaviour. The results published shows ROC of this proposed framework outperforming other well-known classifiers such as Naïve Bias, SVN, NN-BP, C4.5 as shown in Figure 2.9 below.

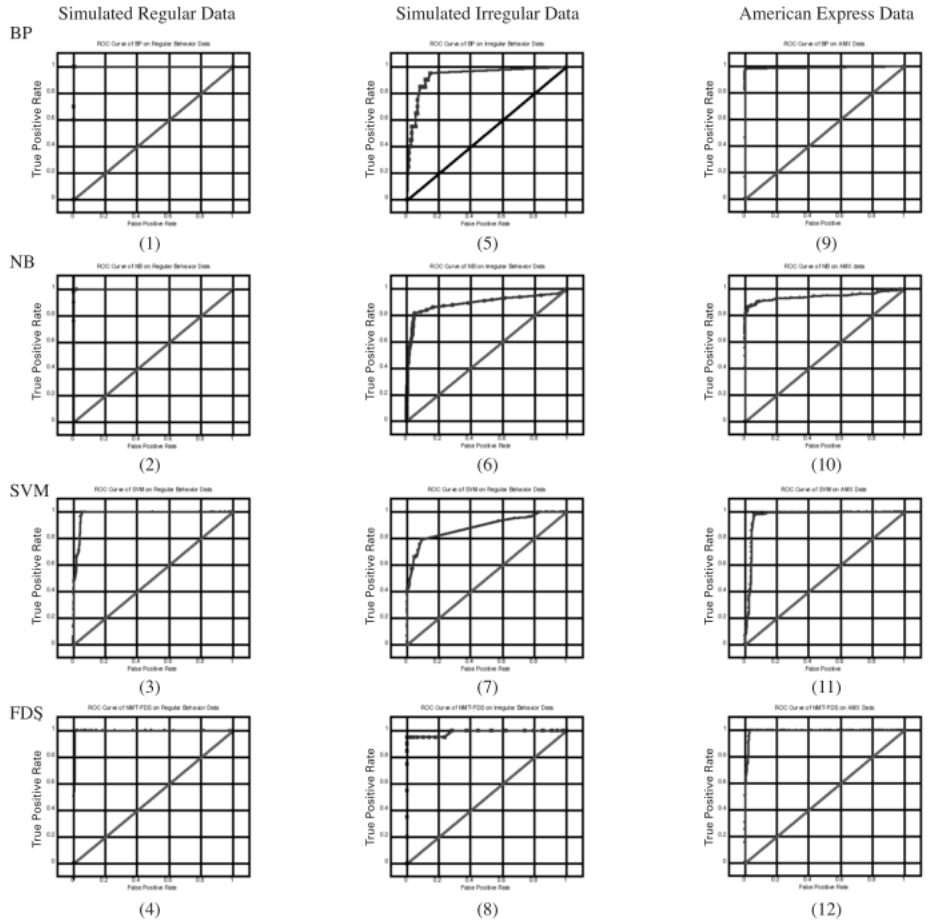


Figure 2.9. ROC curve comparison among the different algorithms [8]

In (54) investigation lead by Li Z, et al. a Kernel-based Supervised Hashing (KSH) model is used to detect credit card fraud. KSH is based on the idea of approximate nearest neighbour that can provide the most similar existing fraud samples for a transaction when the transaction is predicted to be fraud. The KSH had the advantages of being suitable for large and high-dimensional datasets and time consumption especially in the training stage which is much lower than other models.

In paper (55) from Duman, introduces a classification problem of credit card fraud detection with variable misclassification costs function as the available limit of the card. The aim was to minimise the misclassification cost. The solution of such a problem is a combination of genetic algorithms and scatter search, which is based on the behaviour of the fraudsters as well as of

the customers. The suggested algorithm followed the steps of genetic algorithm, but it had some components from scatter search such as the recombination operator instead of crossover operator of genetic algorithm. The performance of the model was based on experimenting with different values of the parameters. The authors made a conclusion that the most effective variable in detecting fraud is the regions for a credit card holder and that the obtained findings might not be generalized to the global fraud detection problem as some variables were missing.

In research produced by Shamil M et. al. (56) considers methods of data analysis and machine learning based on social and transaction graphs for fraud detection. The suggested algorithms for detecting fraudulent transactions are a local outlier factor algorithm based on local sensitive hashing method for finding normal situations or anomalies. Developed algorithm for feature calculation and specific sub-graph patterns by identifying “Volcanoes” and “Black Holes”. “Black hole” refers to a sub-graph, which has only incoming edges from the vertices of the graph not included in this sub-graph. “Volcano” refers to a sub-graph which has only outgoing edges to the vertices of the graph not included in this sub-graph. The task of identifying “volcanoes” and “black holes” is a combinatorial problem. For feature calculation the algorithm is described using vertex-centric approach and is iterative with the maximum number of transactions. The developed algorithm demonstrates quality classification results comparable with a Random Forest when applied on a real transaction graph.

A whitepaper produced by a leading analytics engine provider, SAS (57) provides a framework for detecting, preventing and managing financial crimes, which are cross-channel. The framework includes components for detection, alert management and case management along with capabilities for building advanced analytics. However, this is a commercial offering and requires in-depth knowledge of the SAS software, which is rare in the research and academia. In addition, there are no known cases where the cross-channel fraud detection capabilities been utilised by any of the financial institutions.

Recently there has been a lot of activity in the research area of unsupervised feature learning often referred to as deep learning. As examined in the fraud literature review the existing machine-learning techniques have several limitations in their ability to process raw data and it requires domain expertise (58) to engineer and create a set of features relevant for fraud

detection and machine learning modelling. Recently some work has been carried out (94) (96) that shows this limitation can be addressed to an extent using the principles of deep learning. Since extracting features from raw data and processing through efficient machine learning algorithm is one of the crucial tasks of fraud detection, the next section outlines some of the research been done in a very active field of Deep Learning.

2.3.4. Critique of the Unsupervised Learning Approach

The research highlighted the key outlier detection approach is the clustering approach. The non-parametric nature of creating clusters and sub-clusters makes this a viable approach to fraud detection and quite commonly use in ensembled approaches. Using this approach, the need for a prior knowledge about fraud patterns can be minimised and the existing knowledge can be incorporated in a semi-supervised clustering approach. The research highlighted that the advantage of using hierarchical clustering is that does not require clusters to be defined upfront. However, the disadvantage is that it does not scale well with large data set. While the k-means algorithm can scale well, it does require clusters to be defined upfront. Many key papers highlighted the use of k-means over hierarchical clustering is preferred.

The research into SOM highlighted their usefulness as a visualisation technique and making use of neural network concepts to create coherent clusters as an alternative to k-means. Kohonen's topology has been studied to address the fraud detection space with limited success, the limitation (35) highlighted are:

1. Correctly estimating the probability density function for its input
2. There is no objective function to optimise

Considering, the non-linear stochastic nature of the fraud detection, the above limitations makes SOM harder to converge. There is a need to study SOM topology such as kernel-SOM in the fraud detection space. There is also a need to experiment with the break point analysis and peer group analysis (14) to cater for a wide range of cross channel transactions to determine their effectiveness.

2.4. *Supervised Learning*

2.4.1. Artificial Neural Networks (ANN)

A lot of real-life problems associated with pattern recognition and classification problems are solved using an artificial neural network. The motivation behind the neural networks is adapted from the human brains and its analogy of parallel computation to adapt the neural connection accordingly (36). The idea behind an ANN is to use learning algorithm by modelling how the information is processed by the brain's nervous system called Neuron. An Artifical neuron receives an input signal combines that with a weight. An output is produced by an activation function if the output signal is strong enough as depicted in Figure 2.10 below (36):

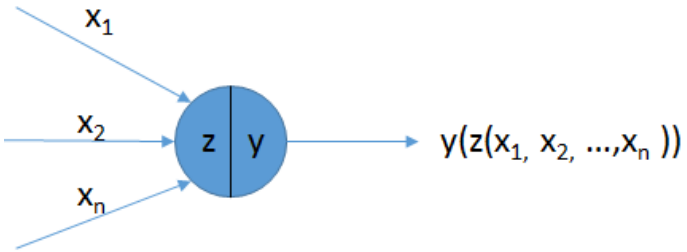


Figure 2.10. Artificial Neural Network Implementation

The first implementation of such algorithm was proposed by McCulloch Pitts (59) known as binary threshold neuron given by the formula below:

$$z = b + \sum_i w_i x_i \qquad \text{(eq. 2.6)}$$

$$y = \begin{cases} 1 \; if \; z \geq 0 \\ 0 \; otherwise \end{cases} \qquad \text{(eq. 2.7)}$$

Where, w_i is the weight of the i^{th} neuron and x_i is the input into the i^{th} neuron and b is bias. The output is given by the sum of all the neurons greater given as 1 and 0 otherwise.

A more general model called (59) perceptron where introduced by Frank Rosenblatt in 1985. A report published by Minsky and Papert (60) highlighted that complex pattern recognition cannot accurately be modelled by a single perceptron and multiple layers are required for universal approximation theorem to sustain. Rosenblatt model was further generalised by adding more layers and more neurons as shown below (60):

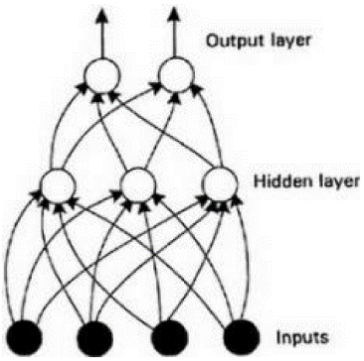


Figure 2.11. Multi-Layer Perceptron (Feed Forward Neural Network) (89)

The above Figure 2.11 consists of an input layer, a hidden layer and an output layer. Hidden layer is used to produce a set of features based on the input and given to output for prediction. In the hidden layer feature transformation based on the input, layers are based on a non-linear function such as:

- Logistic Sigmoid Function:

$$f(z) = \frac{1}{1 + e^{x}}, y: 0 \to 1 \qquad \text{(eq. 2.8)}$$

- Hyperbolic Tangent function:

$$f(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}, y: -1 \to 1 \qquad \text{(eq. 2.9)}$$

In the output layer, the transformation is generally based on the linear function:

- Linear function:

$$f(z) = z, \qquad y: -\infty \to +\infty \qquad \text{(eq. 2.10)}$$

The recent advancement in the artificial neural network has used the properties of the binary / linear threshold function to improve the performance of the ANN is called Rectified Linear Neuron (ReLU) that outputs nonlinear function of the total input. The ReLU function can be defined as:

$$f(z) = \max(0, z) \qquad \text{(eq. 2.11)}$$

Since, the discovery of the ReLU several research papers (61) (62) have highlighted with empirical evidence of the ReLU over performing other activation functions. The points below provide certain properties of ReLU that make this function beneficial (62):

- non-differentiable nature of ReLU precisely at zero, which provides a sparse representation.
- Faster to converge.
- Avoids vanishing grading problem (63).

In the financial services, an ANN is widely used in application such as credit scoring (64), (65) and fraud detection [6,7, 13, 22, 66]. A paper published by Ghosh et al. (67) made use of ANN to show an improvement of 20% to 40% on a real data for Mellon Bank. They made use of the P-RCE neural network that consists of RBF and feed forward neural network to train their network to output fraud scores.

Emanuel Mineda Carneiro et al. presented a paper (68) that showed good detection rate on credit card fraud by combining cluster analysis with the multi-layer perceptron (MLP) neural network. Iterative Naïve Bayesian Inference Agglomerative Clustering (INBIAC) algorithm was used to perform Cluster Analysis (CA) to automatically normalise qualitative data.

Ganesh K, et al. (69) made use of Artificial Neural Networks (ANN) in a supervised manner for fraud detection. Fraud label data was presented to the Artificial neural network (ANN). Their paper showed ANN to be a good technique even when the data is noisy since the weights of the neurons in the hidden layers can correlate with other neurons.

Recently, the paper produced by Syeda et al. (70) suggests the use of parallel granular neural networks for fast credit card fraud detection. It outlines a process for speeding up the data mining and knowledge discovery process. Similarly, a paper by Maes et al. (71) outlines a credit card fraud detection technique that combines Artificial Neural Network with Bayesian Belief Network. The paper highlights the use of Bayesian Belief Network to improve the detection rate can be improved and showed that the training of the network can be relatively faster than a neural network with back propagation algorithm. It showed that the neural network has a high tolerance to noisy data, however, it requires longer training times and re-training the network is difficult as the fraud evolves.

The paper (72) produced by Azeem K et al. outlined the use of Simulated Annealing algorithm applied on the Neural Networks for real time fraud detection on Credit Card transactions. This paper showed that training ANN with simulated annealing performs better over other training algorithms such as Back Propagation. However, the result showed many cases that are

misclassified resulting in a high false positive rate (where legitimate customers are classified as fraud).

2.4.2. Support Vector Machines (SVMs)

SVM is another popular machine learning technique used in the fraud detection domain. Several research papers (73) (74), (75) found SVM ensembled with other machine learning techniques to improve the model performance of the dataset. SVM methodology of the binary classification is well suited for fraud detection as the output of fraud detection can be described by binary class as legal or illegal (fraudulent).

SVMs are useful alternatives to neural networks when the learning objective is a non-convex problem represented as multiple local minima. In such circumstances, neural network tends to get stuck in an area known as saddle point (76), (77). Also, another advantage of SVM over Neural Network is a large number of hyperparameters that are required to be tuned in neural networks (77).

The above two disadvantages of the neural network can be overcome by SVM. The objective of an SVM is to create an optimal hyperplane that can separate the two classes as seen in Figure 2.12 below:

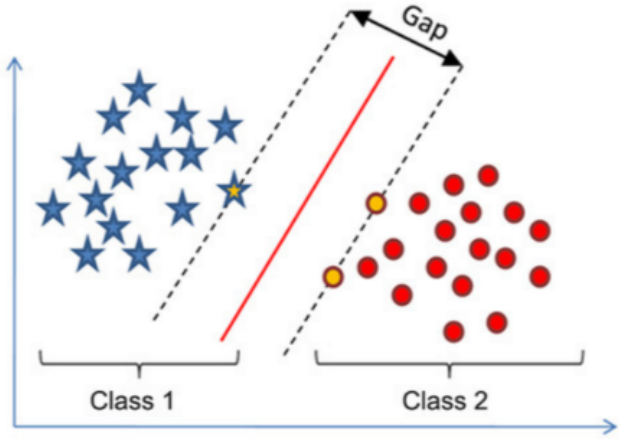


Figure 2.12. Support Vector Machine (77)

In the above diagram, the dotted lines are identified as a hyperplane which outlines a classification boundary. The optimisation can be achieved by maximising the margin between the two classes as further apart as they can be, this is highlighted as the gap in the diagram. This can be achieved by calculating the Euclidian norm and points that are closest to the hyperplane are known as support vector (77). For non-linear separable classes, a hyperparameter is used in the objective

function. For such non-linear cases, the input space is mapped to a higher dimensional feature space so that linear separability can be achieved. One of the key functions to achieve a higher dimensional feature space is called the kernel function (77). The above diagram is a convex optimisation problem, that can be achieved using quadratic cost function known as Lagrangian optimisation as there will be only one local/global minima (7). The most popular kernel functions are (78):

- Linear Kernel – Maps inputs directly to the binary classifier.
- Polynomial Kernel – used to map non-linear input space to a higher dimension feature space.
- Radial Basis Function Kernel: similar to the polynomial kernel, however, empirical evidence suggested the RBF kernel outputs the other two kernels but requires additional parameters that need to be tuned.

In a paper produced by Chen et al. (79) they make use of a support vector machine (SVM) to train the data that is collected as questionnaire-responded transaction (QRT) data of users. It then uses SVM to predict new transactions. In a recent paper, Chen et al. (79) presented a personalised approach by combining SVM and ANN to improve the credit card fraud detection with a limited amount of transaction data. While their research highlighted an improvement when compared to their previous experiments, they also highlighted the need for a fraud domain expert for prior knowledge that needed to be embedded in the model. The research also highlighted a need for the fully automated system as their system required heavy manual intervention.

Sahin and Duma (75) explored the use of SVM for credit card fraud detection and compared the results with Decision Trees. First, they determine the appropriate features that would successfully discriminate fraudulent with legal transactions. Profiling technique is used to create profiles of the stratified samples. Their results highlight SVM can generalise better than the decision trees with a limited set of data. The growth in dataset resulted in SVM model to degrade its performance while the decision tree methods improve its performance.

2.4.3. Decision Trees

Decision trees are considered as a supervised machine learning model that uses the underlying dataset to create a recursive hierarchical decision and their retrospective consequences (80). An example diagram of a decision tree is shown in figure 2.13 below:

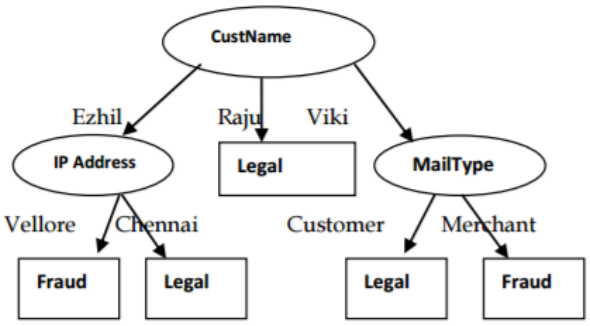


Figure 2.13. Example of a decision tree (81)

In the above diagram, the root node is used for testing the condition for which the classification is required. The last node is used for classification (i.e. fraud or not fraud). The internal nodes are used to recursively reach to the last node. Many popular algorithms have been proposed as (80):

- C.5: is a simple algorithm when compared to the other. It achieves its results by splitting and cropping of the data. It calculates an information gain by splitting the data in a certain way. The threshold is used for stopping the splitting followed by error-based cropping after the growing phase to come to an assignment decision. The information gain is calculated using the following Entropy equation:

$$E(S) = -p_G log_2(p_B) \quad \text{(eq. 2.11)}$$

- CART: classification and regression tree used to construct binary trees and has only two outgoing edges. Towing Criteria is used as splitting algorithm and cropping is done using Cost Complexity Criteria (82). Provides prior probability distribution using Gini Index as given by:

$$Gini(S) = 2p_G p_B \quad \text{(eq. 2.12)}$$

- CHAID: Chi-squared Automatic Interaction Detection, which aims to find output based on the pair of values that is least significantly different.

Decision trees are quite widely used in the fraud detection domain. Sahin and Duman (75) makes use of decision trees to select a list of variables that has the stastical strength of been good predictors. A hybrid approach proposed by S Chen, et al. (83) that uses the decision to create a segmentation of data and then applies logistic regression to further refine their results. They showed such hybrid approaches when combined provide a good predictive model compare to an individual algorithm. As highlighted by S Patil, et al. (84) improvement can be made to the knowledge-based authentication question for credit card application fraud detection process when decision trees are used in to define a set of questions that are much more intrinsic than simple security questions. The tree search can then be further used to eliminate the potential default candidates.

2.5. Deep Learning

Deep learning provides several techniques about learning multiple levels of representation that can quickly determine hidden characteristics and relationships in the data that will provide a good generalisation. Hinton et al. outlined the concept of deep learning in 2006 research paper (85). The paper highlighted, using deep learning one can process information using non-linear functions and hierarchical architectures. The objective is (86) to make these higher-level representations more abstract and based on the lower level features.

Since then the popularity of deep learning today is drastically increased, as Bengio Y, et al. suggested (87), using deep learning techniques can achieve a good level of generalisation, due to the four main characteristics:

- Lots of data: With the advancement in internet and next revolution been the Internet of Things (IoT) there is a vast amount of raw and labelled data that is now available, compare to 1990s.
- Very flexible models: advancements in models, meaning that the next generation of models understands the context of the world.
- Increases in computer power: In order to process lots of data, computer power is also growing. The advancement in GPUs means they are 10milion faster than they were in 1990.
- Powerful priors: according to Bengio Y (87), (88) the use of non-parametric models and classical kernel methods had all of the above characteristics. The single most

characteristic that separates the deep learning models is their ability to compute priors that can defeat the curse of dimensionality and generalise better. In (87) various such priors are outlined, but it is worth noting that apart from smoothness prior the two other priors that play an important role in helping reduce curse of dimensionality are distributed representation, which can learn features from low-level to high-level abstractions and notion of deep frameworks that can provide multiple levels of feature learning.

The fundamentals of deep learning date back to 1980 when they were first introduced by D. Rumelhart, et al. in their book called Parallel Distributed Processing (89). They coined the term "Connectionism" and outlined that the concepts are not represented symbols in our brain but by patterns of activation. The deep learning principles are built on similar beliefs. The idea is that symbols can only be manipulated by rules and when symbols are considered as objects, in their true nature they do not have any similarity built between them. For example, the concept of dogs and cats are closer together in some space and further away from the person.

2.5.1. Evolution journey of Artificial Intelligence

Figure 2.14 below outlines an evolutionary journey of Artificial intelligence (AI) over time (87).

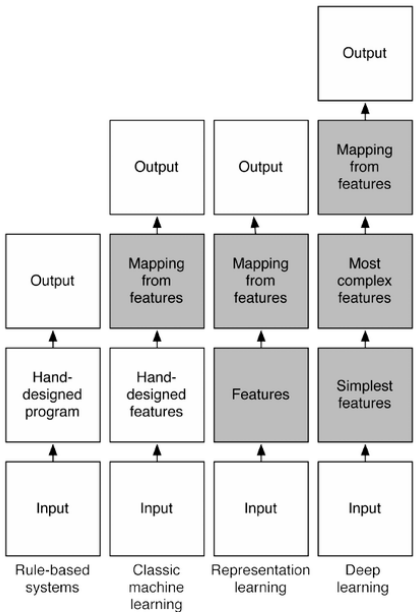


Figure 2.14. History of AI (87)

In the classical rule-based system, the domain knowledge and feature where handcrafted so that the system knows how to linearly compose these features to give some output. With the

advancement of machine learning the notion of composing features on top of features were introduced but still require heavy manual intervention. However, it was believed that only one level depth (1-h(x)) is enough to represent anything efficiently as highlighted by the seminal paper (90). It uses K-Nearest Neighbour which takes the input spaces and breaks it down into pieces (known as regions) and for each of those regions, there are examples that will tell you the answer. It showed that the complexity of the functions that is represented is growing linearly as the number of examples increases in each of those regions.

In representation learning such as Neural Networks, the above problem was solved as a number of regions can be represented with a linear number of parameters using hidden layer. Simple Neural Networks requires manually breaking the input space into multiple regions and using hidden units H(x) that discovers semantically meaningful concepts.

In deep learning, this manual intervention is reduced, and more and more features can be added on top of other features which are automatically discovered and composed together in the various levels to produce the output.

2.5.2. Deep Learning Architectures

Deep learning provides enhancement to the neural networks. The traditional neural network consists of a single hidden layer considered as a shallow network. Deep learning consists of many hidden layers each providing different representation and hence different abstraction that can generalise well. The terminology in deep learning is different to the terminology known for neural networks (87). The terms such as nodes, connections, etc. are not used anymore instead new terms have been coined that are closer to technology architecture terminology. These terms are referred to as modules and library of modules so a notion of "wiring of the system" can be applied to automatically figure out input from the output and to compute gradient with respect to all the parameters internal to the system.

Some of the common architecture for deep learning are:

- Deep belief networks (DBN): consist of many hidden layers, however, the top two layers have un-directed connections between them. This result in a network that is probabilistic and stochastic (91) as shown in Figure 2.15 below.

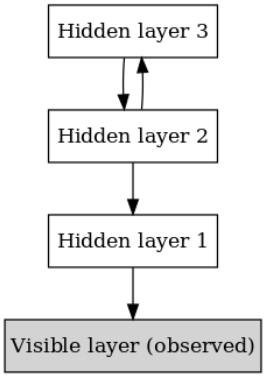


Figure 2.15. Deep Belief Network Architectures (91)

- Boltzmann machine (BM): considered as a stochastic binary machine. The network is represented as symmetrically connected nodes (91) as shown in Figure 2.16 below.

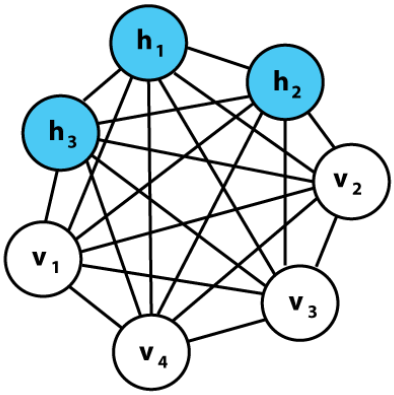


Figure 2.16. Boltzmann Machine (91)

- Restricted Boltzmann machine (RBM): there is no connection between the hidden neurons and the connections between the visible layer and the hidden layer are undirected (91). RBM have proven to be capable of providing a good level of posterior distribution which is a big advantage over directed belief network as shown in Figure 2.17 below.

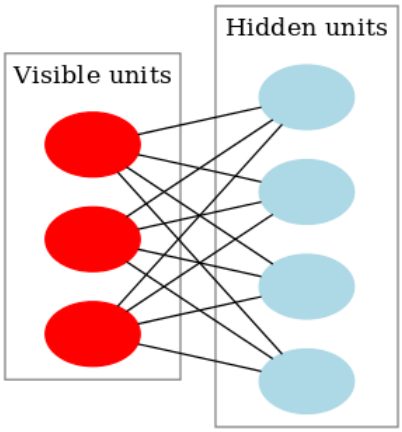


Figure 2.17. Restricted Boltzmann machine with three visible units and four hidden units (no bias units) (91)

Ugo F, et al. (92) explores the effectiveness of a detection approach using the Discriminative Restricted Boltzmann Machine and applied it to the network intrusion dataset. Their results show a moderate level of success to the real-life data. It combines the expressive power of generative models with good classification accuracy to infer part of its knowledge from incomplete training data. In this paper, the Discriminative Restricted Boltzmann Machine, a recently proposed classifier based on the family of energy-based models, has been applied to network anomaly detection in a semi-supervised fashion. A similar topology can be used in the fraud detection system where the data from many different channels can be combined to improve the detection rate of the model.

- Deep neural networks (DNN): DNN is generally referred to as a feed forward multi-layered neural network. It contains many hidden layers and provides probabilistic interpretation by providing linear and non-linear activation functions. It also generally referred to as fully connected multilayer perceptron and often initialized by using stacked RBMs or DBNs (91) as shown in Figure 2.18 below.

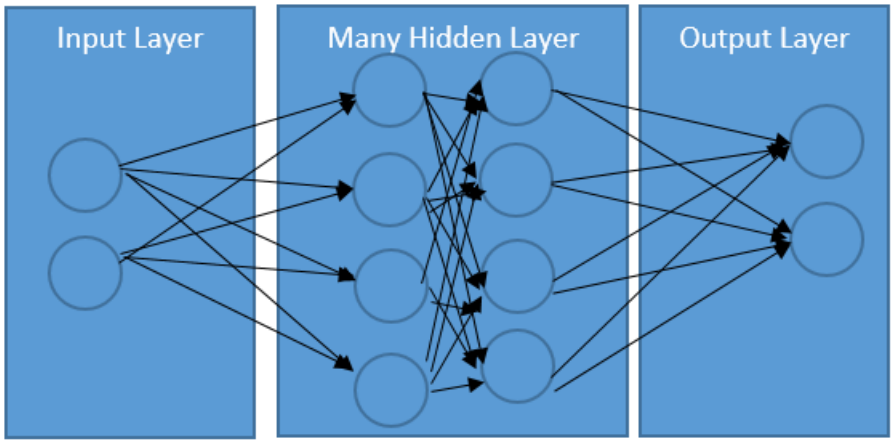


Figure 2.18. Deep Neural Networks

- Convolutional deep neural networks (CDNN): A Convolutional Deep Neural Network is comprised of one or more convolutional layers (often with a subsampling step) and then followed by one or more fully connected layers as in a standard multilayer neural network. Convolutional neural networks use three basic ideas: local receptive fields, shared weights, and pooling (93) as shown in Figure 2.19 below.

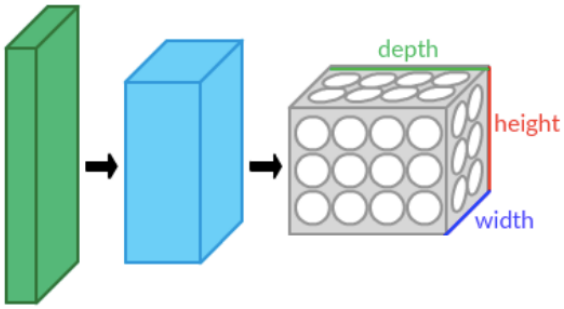


Figure 2.19. CNN layers arranged in 3 dimensions (91)

Kang Fu, et al. (94) outlined a methodology where credit card transaction data is represented as feature matrix and applied convolutional network for fraud detection to capture the intrinsic patterns in the data. They overcome the class imbalanced problem by applying cost-based sampling to alleviate the fraudulent dataset. In their proposal, they calculated trading entropy based to model customer profiles by measuring customer's preferences over the different timeframe. Feature engineering was performed, and latent variables were calculated and arranged in a way that can be easily passed through the CNN's filter bank. Their model was run on the real

banking data and empirically proved that their model outperforms the existing bank's model.

- Recurrent neural networks: Recurrent neural networks (RNNs) can be considered as another class of deep networks for unsupervised (as well as supervised) learning, where the depth can be as large as the length of the input data sequence (91), (95) as shown in Figure 2.20 below.

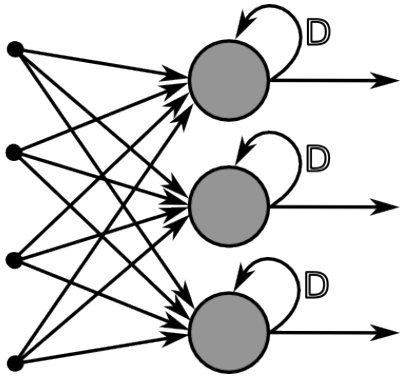


Figure 2.20. Recurrent Neural Networks (91)

A variant of RNN called Long Short-Term Memory (LSTM) is used to profile customer behaviour over several time steps. The recurrent neural network is used for processing sequence of values and it can share parameters across different part of the model (91). It this sharing of statistical strength over a long time across difference input sequence length makes it generalise well for time-series problem. The state of the system at given point in time can be given by:

$$S_t = f_\theta(S_{(t-1)}, X_t) \qquad \text{(eq. 2.13)}$$

Where S_t is the state of the system at time t, f_θ deterministic function, X_t is the input vector. The above equation can be diagrammatically represented as shown in Figure 2.21 below:

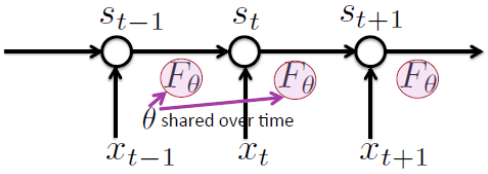


Figure 2.21. RNN states over different input time space (91)

The above diagram outlines that the current state of the system x_t is dependent on the previous state of the system at $x_{t-1.}$

Ando Y, et al. (8) applies recurrent neural networks (RNN) to profile fraudulent behaviour rather than profiling legal behaviour. The paper suggests using the similar RNN technique to effectively model sequential data, for example, speech recognition, natural language processing and handwritten characters recognition to apply at a web log data that contains fraudulent transactions such as stolen credit cards to build pre-paid cards or to use the stolen card for online auction purchase. They compare the F-measure of the RNN (LSTM and non-LSTM) and SVM (linear kernel and RBF kernel) and show RNN with LSTM outperforming the other three types. In a thesis published by Benard Wise (96), he provides a good overview on the use of LSTM to model customer behaviour in order to better detect credit card fraud. Benard analysed this problem by comparing LSTM with Feed Forward Neural Networks and SVM and the experiments showed LSTM outperformed FFNN and SVM.

In a paper produced by Xurui L et al. (97), a sandwich structured sequence learning model called “within-between-within” (WBW) is used for credit card fraud detection by stacking an ensemble model, a deep sequential learning model and another top-layer ensemble classifier. Firstly, artificial feature engineering work like RFM and weight of evidence (WOE) is carried out. Next, gradient boost decision tree (GBDT) model is involved to optimize features within a single transaction. Then gated recurrent unit (GRU) model is applied on transformed sequential samples to learn relationships between transactions better. Finally, a top-layer Random Forest classifier is trained using optimized transaction eigenvectors. The “WBW” sequence learning architecture was compared with simpler structures like “WB” or “BW” and other common classification models like SVM and Logistic Regression and has been proved to be more efficient. For further enchasing the model’s performance the paper suggested using various timestamp ranges and attention mechanism.

The findings of the research by Benjamin R et al. (98) they showed that LSTM recurrent neural network model is able to represent sequences of communications between computers on a network and identify outlier network traffic. The NetFlow was

tokenized and compressed into sequences of “words” that form “sentences” representative of a conversation between computers. Then these sentences are used to generate a model that learns the semantic and syntactic grammar of the newly generated language. The language model is then used to predict the communication between two IPs and the prediction error is used as a measurement of how typical or atypical the observed communication were. The model was tested in three scenarios; dataset with no attacks, with both attack and non-attack traffic present and dataset with denial-of-service attacks. The proposed model was able to detect malicious network traffic without the assistance of labelled training data and without visibility into each machine’s internal state or processes. The network model with both attacks and non-attacks present outperformed the other models.

2.6. Summary & Conclusion

The research carried out as part of the literature review outlined a vast number of papers both part of the academic journals as well as white papers from the commercial organisations focusing in the field of fraud detection. The range of paper covers various approaches such as supervised, unsupervised and hybrid techniques to a fraud detection problem. The research showed the use of the classical approach to fraud detection such as an expert rule-based approach that is based on the experience, the intuition of a domain knowledge expert wasn’t able to keep pace with evolving fraud and other techniques were required (99). Most of the research has been done in the field of credit card fraud detection using most popular machine learning techniques such as Neural Networks, Hidden Marko Models, logistic regression and other (52), (100).

A limited set has looked at the problem in a holistic customer-centric manner (101), where patterns and customer behaviour “learnt” can be utilised in cross-transactions. One such system described in (102) attempts to combine insurance data from multiple data sources and models are executed on big data.

Currently, machine learning community is driving the majority of the research in deep learning. The review showed the importance of building and learning deep hierarchies of features to resolve problems such as breaking down (91) the unknown factors of variation in the input space resulting in poor generalisation. The current challenges in the deep learning

are around optimisation (103) that needs to be further researched. Li Deng et al. showed with speech recognition examples; a deep learning algorithm is very effective on large datasets than smaller ones. The concerns around learning rate, the effectiveness of regularisation, the number of hidden layers and the number of neurons per layer, etc. does also require an understanding of the architecture and still very sensitive just like neural networks.

Not a lot of research has been done in using these deep learning concepts to detect fraudulent events in financial intuitions. The research showed one such empirical evidence where deep learning is used for fraud detection by PayPal (104), however, does not provide any detail on the actual architecture or feature learning aspects. Further study is required to understand if deep learning techniques such as unsupervised learning for automatically discovering data representation or even can combine with supervised learning to detect fraud and improve the detection rate.

Based on all the findings, literature review and author's experience, the research elements described in the proposal of applying advanced machine learning techniques on data set for cross-channel fraud detection is still valid. One may extend some of the technique described in the literature review such as SMOTE (105) and Discriminative Restricted Boltzmann Machine (92) to be applied to fraud dataset. However, the advancement in the machine learning techniques such as deep learning means there could be a possibility to build a complex hybrid model that can combine the generative power off feature learning algorithms via unsupervised learning with a supervised learning algorithm. Supervised models can be trained on existing labelled fraud data to improve the overall fraud detection rate especially detection that miss classified cases.

The research provided insight into RNN as a technique used for supervised credit card fraud detection. There are very limited set of published articles that uses RNN for fraud detection. However, research article published do provide insight into the detection mechanism based on RNN aimed at finding behaviour that deviates from normal behaviour achieved relatively good success in particular where labelled data is few and scares. This technique of detecting fraud that is different in nature from historical fraud (like supervised learning) or in other words technique that makes use of new, unknown mechanisms resulting in a novel fraud pattern as the time evolves. The researched articles showed that RNN a powerful technique

for modelling time series events. This provides a proof of our hypothesis that cross channel transactions can indeed be modeled as sequences of time series events. Therefore, the main goals of this research will remain un-changed as the research showed a significant amount of work is still required to look at the financial data set for efficient fraud detection.

2.7. Literature Synthesis

This section describes the pertinent literature that has been amalgamated in creating the cross-channel fraud detection system as shown in Table 2.4.

Literature	**How it is used**	**Section**
Rule based expert system	Used as part of the User Agent Model as part of the change analysis, when a user agent string is changed.	Appendix – A: UA Model
Clustering	Clustering is used as part of the data analysis and as part of the normalisation technique (i.e. PCA, SMOTE).	Data Analysis Normalisation Modelling Results – Chapter 5
Benford Law	Benford Law is used to calculate the probability of a specific digit to be found in the transaction amount.	Append – A: Benford Law
Markov Model	Markov Model is created as part of the cross-channel fraud detection system to evaluate with other models such as SVM and RNN	Markov Model
Synthetic Data Generation	Synthetic data generation is taken to determine the effectiveness of new algorithms. Thesis provides a novel approach of generating synthetic, with an aim to generate data quickly and efficiently that is a close representation of the real data.	Data Generation
Artificial Neural Network	This building block will create a fraud detection model using neural network topology.	Neural Network Model
Support Vector Machine	The SVM model is trained by starting with one variable and continue by adding one by one based on features, Amount,	SVM

	User agent and IP address. SVM is used to evaluate with other models such as Markov Model and RNN	
Recurrent Neural Network	RNN can be conceptually modelled to arbitrary map input sequences and to perform fraud detection. In this thesis a variant of RNN known as LSTM is used to evaluate with other models such as Markov Model and SVM	RNN

Table 2.4 Types of Fraud in the banking sector

3.

Cross Channel Fraud Detection System- CCFDS (Conceptualisation of fraud detection framework)

3.1. Overview

As discussed in Chapter 1, financial services such as banks face an increase in the payment fraud across their multiple channels. Existing approaches adopted by the banks are primarily rule based expert systems. These systems are channel centric and inadequate to detect cross channel fraud or zero-day attacks. These inefficiencies are further exacerbated by requiring fraud domain experts to maintain a large repository of complex rules and manual investigation that takes a long time. Research in Chapter 2 outlined a various machine learning approaches to fraud detection that are based on predictive analytics and in specific few successful approaches are based on supervised neural network techniques in the credit card space (72). However, these supervised techniques require a large volume of labelled data. Unsupervised based machine learning techniques that are based on behavioural profiling can detect various anomalies such as zero-day attacks and previously unseen fraud MO. However, such existing systems are plagued with a high false positive rate as they fail to differenciate between anomalous events and legal events. To resolve this deficiency requires manual intervention and hence harder to achieve overall efficiency.

This research is for a novel payment fraud detection system (PFDS) that can detect anomalous events quickly, accurately across multiple channels and can dynamically evolve to maintain the efficiency with minimum input from subject matter expert. The intent is to advance the state of the art of PFDS by proposing a framework that is cross-channel and self-tuning and achieve the following contributions:

- Provide flexible ingestion layer to capture customer interaction over multiple channels to create single customer view.
- Automate the process of flagging complex suspicious transactions over multiple channels in real time.

- Automate tuning of control parameters eliminating the need for domain expertise and the underlying detection algorithm.
- Propose a new machine learning model for optimal performance of a PFDS.
- Automate the optimization process on the line to achieve the goals established in the proposed performance model.

Figure 3.1 below displays a framework that illustrates the relationship between the theoretical and technical concepts proposed.

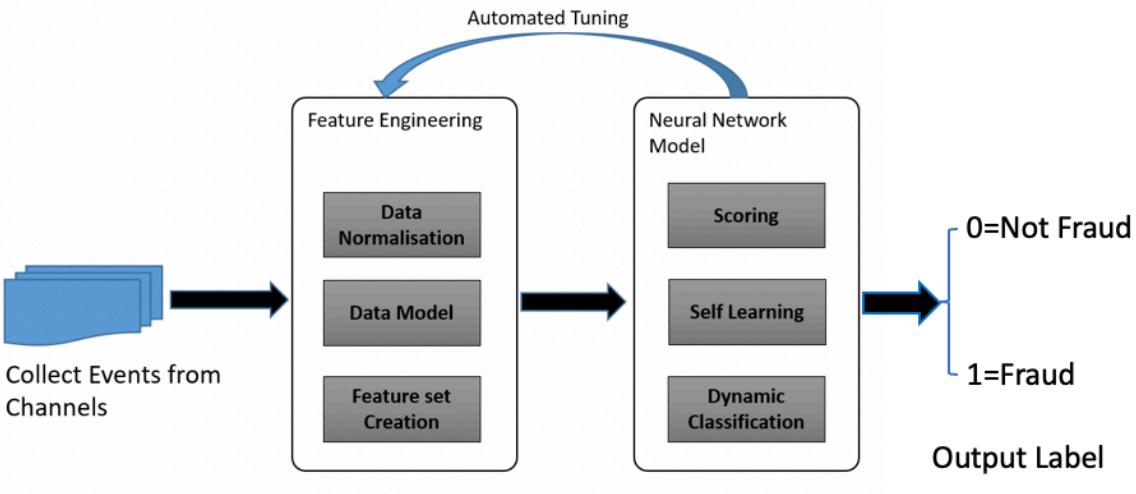


Figure 3.1. Conceptual Fraud Detection System Framework

The above diagram highlights an approach for achieving the desired contribution. The framework will consist of the following building blocks:

- Feature engineering: this building block will examine and analyse the events collected from bank's channel and create latent variables that will form a feature matrix that can be processed easily by the neural network model. Techniques such as data modelling through entity relationship diagram, data normalisation, principle component analysis, feature set creation will be examined in this section. Further elaboration can be found in Section 3.2.
- Neural Network model: this building block will create a fraud detection model using neural network topology. This section will examine the state-of-the-art neural network model that can perform automated improvement to its hyper parameters when a degradation of the model is detected as a process of automated tuning. This

section will also cover a section for evaluating model performance. It will also perform the following function:

- Scoring: this building block will be responsible for outputting a score from 0 to 100 that outlines a probability of current event been fraud. The higher the score the higher likelihood of an event been fraudulent.
- Dynamic Classification: this building block will be responsible for converting the probability score to a fraud / not-fraud label.

- The output of this framework is a label that classifies an event as either a fraud or not a fraud.

  Further elaboration can be found in Section 3.3.

Two environments will be created as shown in Figure 3.2 below:

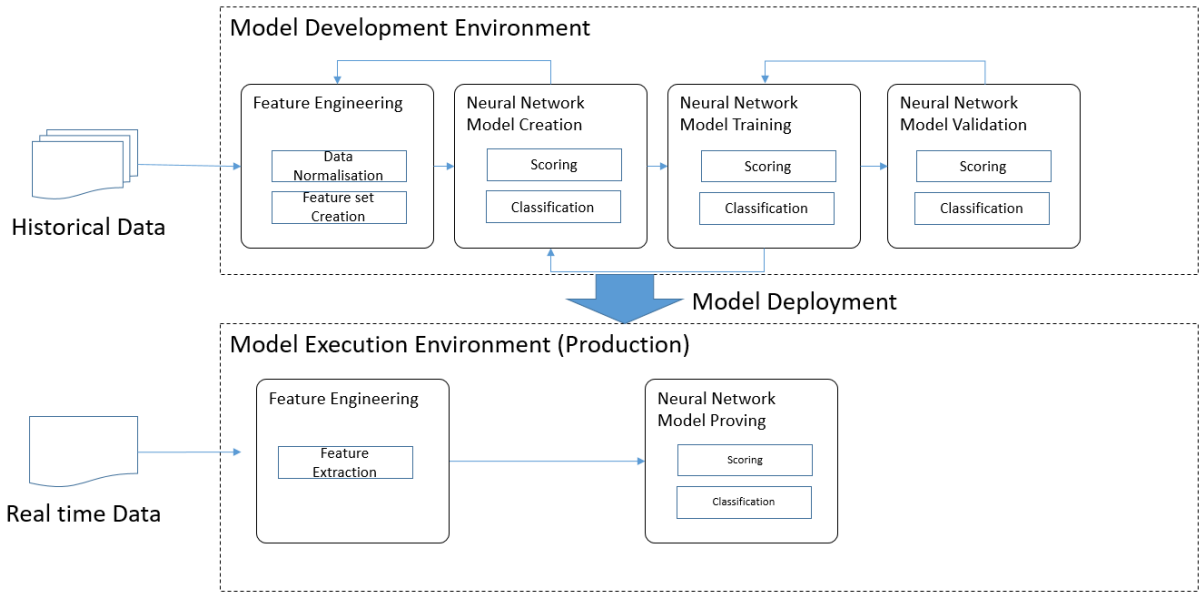


Figure 3.2. Two Environment: Model Development and Model Execution

The model development environment will be responsible for developing the model using historical data. Once the model is trained and validated, it will be deployed on to the model execution environment. In this environment, the real time events will be ingested and evaluated for fraud /not-fraud classification. The following sections describe the above building blocks in more details.

- Section 3.2 describes the process of feature engineering and feature extractions.

- Section 3.3 describes the architecture of the neural network model that will be used for event scoring and classification. This section concludes with the proposed evaluation approach.

3.2. Feature Engineering

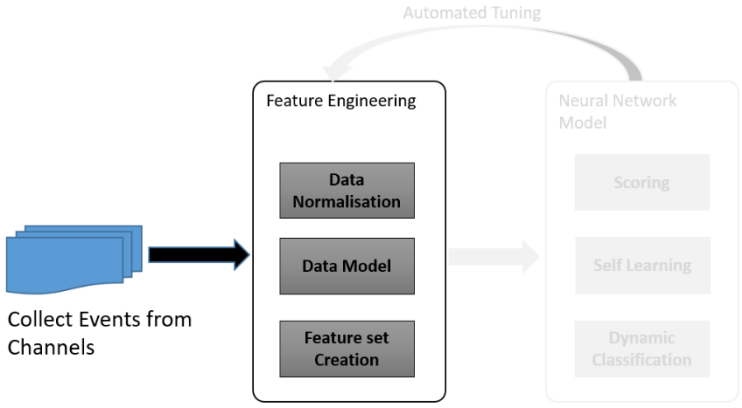


Figure 3.3. Feature Engineering

Feature engineering as shown in Figure 3.3, will be the fundamental part of this research. This process will make use of the domain knowledge to create features that can be easily processed by the proposed neural network model. The examination of the data will be performed here to learn about the impact of data correlation on different channels such as cards and internet banking. Work in (106) provides a definition of a feature as "*aggregate view across life-cycle artefacts"*. This definition will be used to break the feature engineering process as:

- Understand the property of the data in relation to the fraud detection problem to gain insight into the data required for the neural network model. For example, the impact of data variables such as amount, timestamp, device data, behaviour data and others will be examined here.
- Carrying our various experiments combined with domain expertise to analyse which data elements contribute to model efficiency. For example, few transactions with a small amount of money, followed by a large sum of money is an indication of fraud. Therefore, the amount plays a vital role as a data element.

The approach to feature engineering can be outlined in Figure 3.4 below:

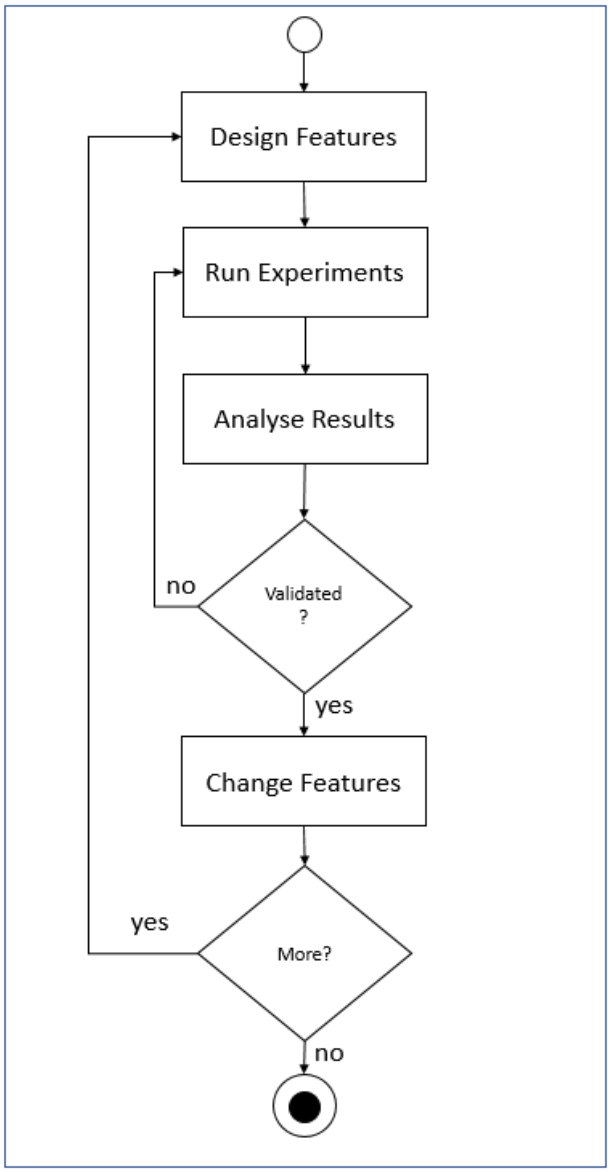


Figure 3.4. Feature Engineering Flow

This approach can be summarised as:

- Based on the raw data and techniques such as RFM (Recency – Frequency – Monetary) design a set of features (107).
- Validate the features by running a set of experiments.
- Create more features and repeat the process.

To gain a better understanding of the data space and how data from different customer interaction channels such as internet banking and POS can be related. Let *X* be the set of all

possible events $\{x_1, x_2, x_3, \ldots x_n\}$ of a bank's customer across two channels namely cards and internet banking.

For the internet banking channel this could represent customer journey on the internet banking site such as customer login, adding a payeeor making payment as shown Figure 3.5 below:

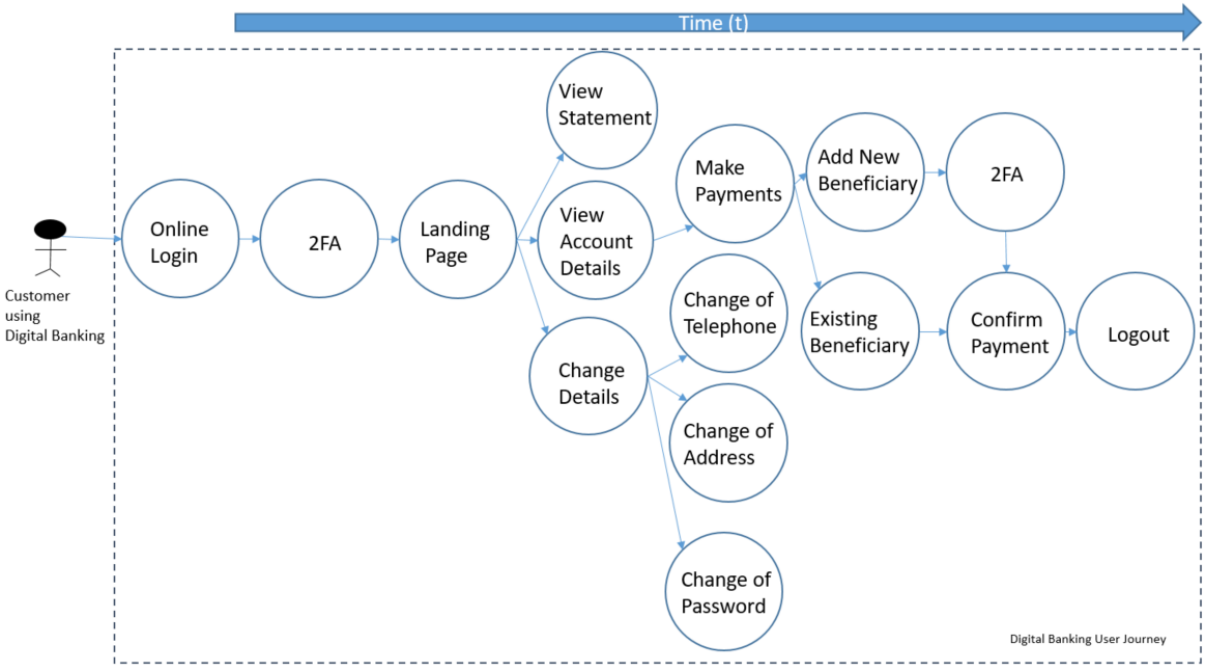


Figure 3.5. State event diagram for online banking

When a customer login to their internet banking account, their journey can be modelled using the above diagram where each page (defined as a circle) is represented as a "state" of a customer and "event" is represented as state transition (defined as an arrow). The above diagram depicts a customer journey and key states as shown in Table 3.1:

State	Description
Online Login	Online login page where customer can enter username and password.
2nd Factor Authentication	A page where customer can carry out 2nd factor authentication to further prove their identity.
Landing Page	A landing page where customer is shown a list of their active accounts and products.
View Statement	A page where the customer can view all their statement.

View Account Details	A page showing details about the selected account is displayed with top 10 most recent transactions.
Change Details	A page where customer have an option to select their specific details.
Change of Telephone	A page where customer can change their telephone number.
Change of address	A page where customer can change their address.
Change of password	A page where the customer can change reset their online banking password.
Make Payments	A page where customer can make payment to a payeeor setup a standing order or direct debit.
Add new Beneficiary	A page where the customer can add a new payeeby inputting beneficiary's account number and sort code.
Existing Beneficiary	A page where customer can make payment to existing beneficiary.
Confirm payment	A page for customer to confirm the payment.
Logout	An event via a button to log customer out.

Table 3.1 Online Banking User Journey Description

For the card payment journey, the term "event" could represent the customer purchasing items from the e-Commerce site by making *"card not present"* transaction or customer purchasing items by making *"card present"* transaction using Point of Sale Terminal. The type of events across different cards channels is shown in Figure 3.6 below. In this context, the states are the transaction events (depicted as Txn N):

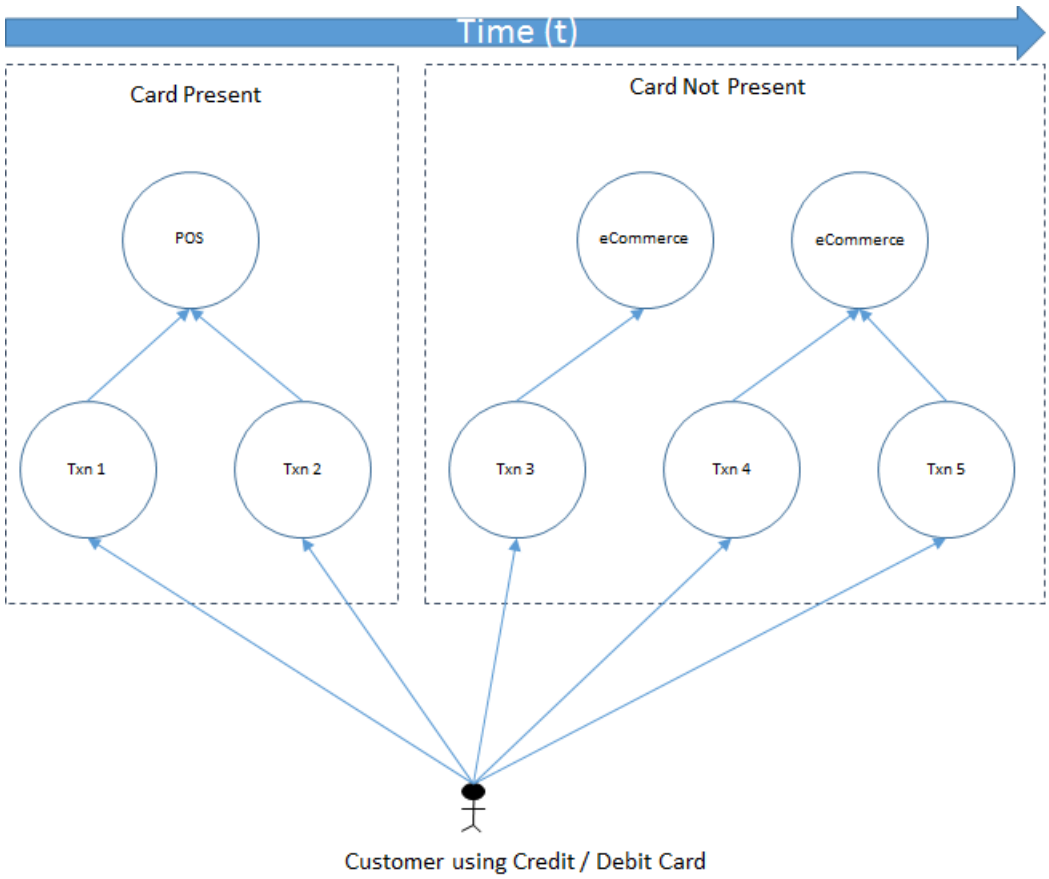


Figure 3.6. State event diagram for Cards Usage

The above diagram depicts the possible states for the credit/debit card transactions given in Table 3.2 below:

State	Description
Transaction (Txn_n)	Customer paying for their purchase

Table 3.2 User Journey for Cards Transaction

From the machine learning standpoint, this can be formulated as the C^n as the set of all cards events and I^n be the set of all Internet Banking events. Given this the input events X can then be defined as $(C^n \cup I^n)$ when card event is presented with internet banking event. Similarly, $(C^n \cap I^n)$ when card event is presented which is independent to internet banking event and vice versa. Let V be the vector representation of the information available for each event $\{v_1, v_2, \ldots v_n\}$. Typically, this would include:

- Transactional data – captures key attributes of a customer transaction that defines their characteristics and generally stored in OLTP relational databases. This includes

monetary events (i.e. Account Bill Pay) as well as service events (i.e. change of address). When applied to the fraud domain this type of data can be used in the setting of RFM (Recency, Frequency, and Monetary) (107), where data can be aggregated in accordance with a various time dimension. Unsupervised learning algorithm such as clustering can then be applied (20) for fraud detection.

- Personal and Account data – Transaction data can be augmented by personal and account data for better fraud detection. For instance, it shows that several factors such as age, gender and income contribute to people committing fraud (7).
- Behavioural data – Behavioural data is defined as information that provides context into individual's behaviour. When reference in the context of fraud detection on bank's channel this will include data collected on the customer side as well as on the bank's side. For example, for online banking data collected on the customer side will include mouse movements and keystrokes (Reference own paper). The bank side data will include page navigation, time spend on page, etc. (19).

Let Z be a classifier that has the following values $\{fraud, not\ fraud\}$, then model can be defined in Figure 3.7 below:

Figure 3.7. Payment Fraud Detection System

Where, x_i is the event, v_i is the information about the event and $z_i$is the classification output. Probabilistic view can be defined as:

$0 < P(x_i \in C^n|v_i) < 1$ for card events and (eq. 3.1)

$0 < P(x_i \in I^n|v_i) < 1$ for internet banking events (eq. 3.2)

The above two equations can be used to derive probability when events from different channels are analysed independently in Figure 3.8 below:

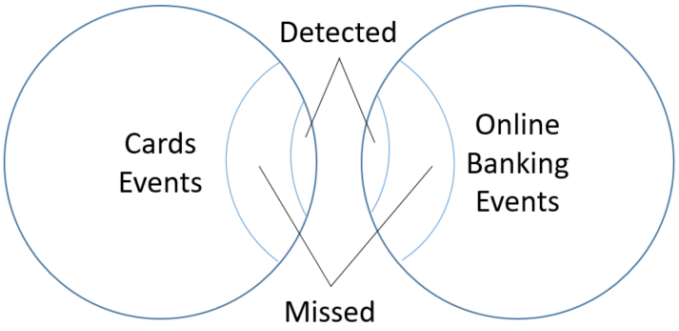


Figure 3.8. Fraud Probabilities for Cards vs Online Banking

Or a joint probability is given as:

$$0 < P(x_i \in C^n|v_i, I^n|v_i) < 1 \quad \textbf{(eq. 3.3)}$$

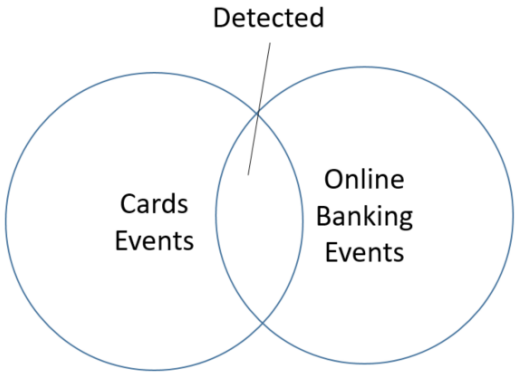


Figure 3.9. Joint Fraud Probabilities for Cards and Online Banking

The joint probabilities as depicted in Figure 3.9 above can be further explored once the data associated with each of the events are well understood. The next section on the data model provides an insight on the data.

3.2.1. Data Model

Figure 3.10. Data Model

As described in Figure 3.10, this section provides a Logical Data Model (LDM) that highlights the cross-pollination of data for cross channel events. It provides a unified view of the data used for Payment Fraud Detection System regarding of the channels.

There are four main subject areas within the logical model:

- Static Data (customer, account, address, customer-account linkage, product holdings etc.);
- Contextual Data (covering non-monetary data such as session data and logon requests);
- Monetary Transaction Data (covering transfer transaction, credit card transaction, POS transaction); and
- Reference Data (transaction type, fraud type, payment type etc.)

Figure 3.11 below provides a database schema representation of the Logical Data Model.

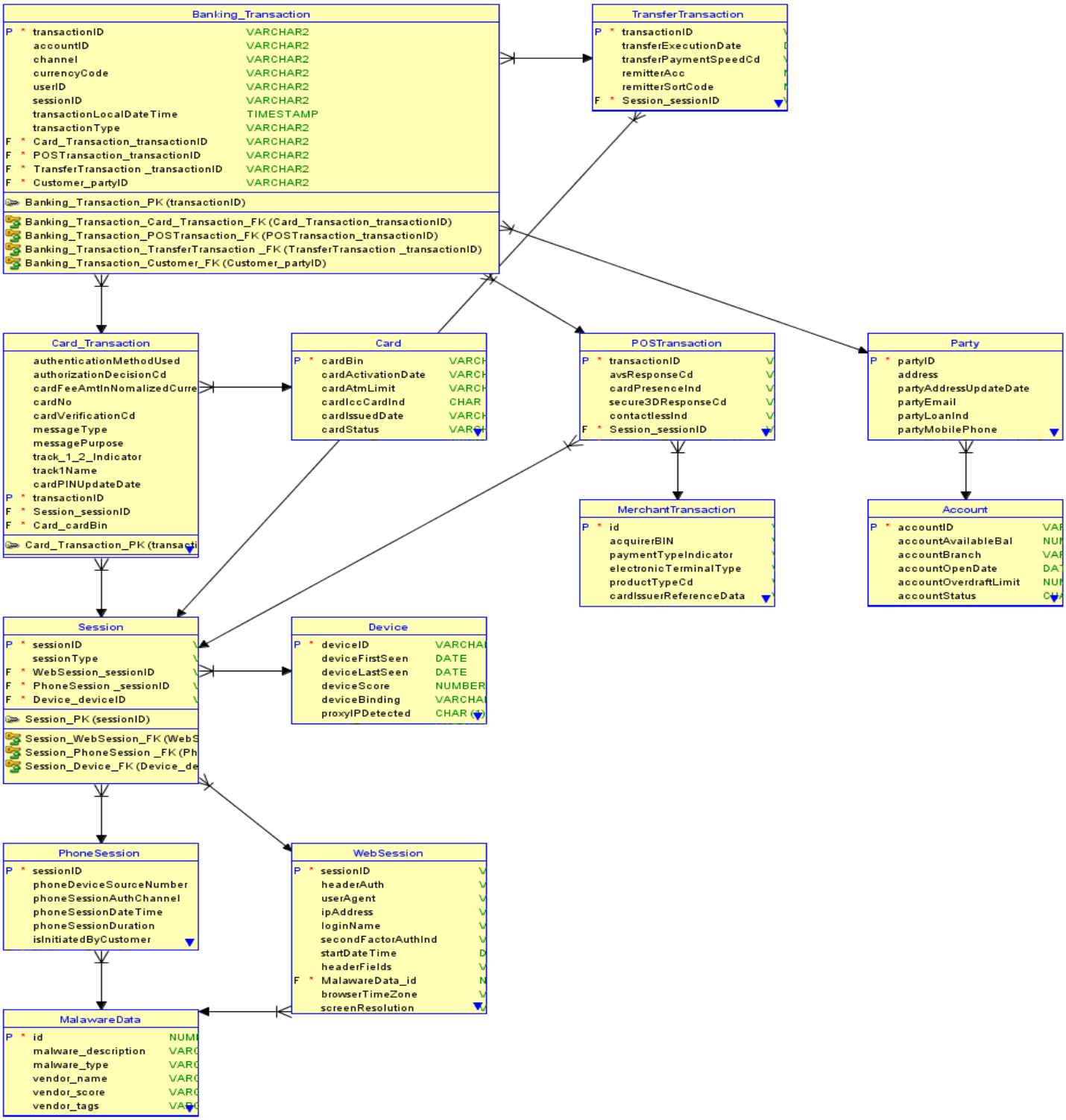


Figure 3.11. Logical Data Model for Payment Fraud Detection System

Table 3.3 below describes the entities:

Entity	Description
BankingTransaction	This entity presents, events that are received from the channels. They describe account activity involving money and customer's behaviour on a specific channel.
TransferTransaction	A transaction that is done by the party on the online channel, which involves the transfer of money from the party's account to a remitter.
CardTransaction	This will include relevant data about the card transaction and define a type of card transaction as CardPresent or Card Not Present Transaction.
POSTransaction	This entity will include data about the card present transaction. When a card and party is present at the same time. For non-chip and pin cards, data from the magnetic strip on the back of the card will be used.
MerchantTransaction	This entity will include data about the card not present transaction when the party or card are not present.
Session	Provides information about the user activities on a browser or a mobile to a specific unique id for a specific amount of time. This unique id is passed back and forward, between the bowser/mobile app and the web server on the bank's side until the time expires or the user logs out.
PhoneSession	Provides information related to the phone and user behaviour as part of a session. Phone's context data will be passed back to the server.
WebSession	Provides information related to the web browser and the user behaviour as part of a session. Browser's context data will be passed back to the server.
MalwareData	Malware data found on the device, using external service.
Device	Details about the device used by the party to execute a banking transaction.
Party	The Party defined as a customer to the financial institution that will be monitored by the PDFS. In specific, the party represents a person who holds account/s within a financial institution.
Card	Details about the actual card involved in a particular transaction. This includes credit cards, debit cards, cash cards, etc.
Account	The account entity is the actual financial account on which the monetary transaction is performed, and which is related to the Party.

Table 3.3 Entities for Data Processing Service

Table 3.4 below provides a description on the relevant attributes associated with each entity:

Entity Name:	**BankingTransaction**	
Attribute Name	**Data Type**	**Description**
transactionID	Varchar	Unique identity for each transaction represented as the primary key.
accountID	Varchar	A unique identifier for the account that the transaction is performed on.
Channel	Varchar	Constant values of all supported channels (Online, Phone, Mobile, POS, ATM).
currencyCode	Varchar	The local currency code.
userID	Varchar	The user that is associated with the transaction (or transaction version).
sessionID	Varchar	A unique identifier for a session (either web or phone).
transactionLocalDateTime	Timestamp	The local date and time of the transaction. When receiving transaction version, this field holds the local time and date of the current version.
transactionType	Varchar	The client transaction type. This also includes the client specific services (information change), such as address change, password change etc.

Table 3.4 BankingTransaction Entity Definition

Entity Name:	**TransferTransaction**	
Attribute Name	**Data Type**	**Description**
transactionID	Varchar	Unique identity for each transaction represented as the primary key.
transferExecutionDate	Varchar	The date the transfer will actually take place (not the setup date i.e. not the transactionLocalDateTime).
transferPaymentSpeedCd	Varchar	The speed at which the payment will be executed. (i.e. faster payments or BACS).
remitterAccountNo	Varchar	Account number of the remitter where the money will be deposited.
remitterSortCode	Varchar	Sort code of the remitter where the money will be deposited.

Table 3.5 TransferTransaction Entity Definition

Entity Name:	CardTransaction	
Attribute Name	**Data Type**	**Description**
transactionID	Varchar	Unique identity for each transaction represented as the primary key.
authenticationMethodUsed	Varchar	The code describing the actual authentication method for the transaction.
authorisationDecisionCode	Varchar	The result of the authorization process.
cardFeeAmount	Varchar	The surcharge amount converted to the bank's base currency.
cardNo	Varchar	Also known as PAN. The card number, which is an identification of the card.
cardVerificationCd	Varchar	Card Verification standard response code.
meassageType	Varchar	A 4-digit numeric field which classifies the high-level function of the message ISO (MTI).
messagePurpose	Varchar	The transaction type as deduced from the Message Type Indicator (MTI) and other fields, for example - Refund authorisation requests.
track_1_2_indicator	Varchar	Indicates whether track 1 or 2 was read from the card.
track1Name	Varchar	The name of the cardholder that is encoded on the magnetic stripe.
cardPinUpdateDate	Timestamp	The update date of the card PIN.

Table 3.6 CardTransaction Entity Definition

Entity Name:	POSTransaction	
Attribute Name	**Data Type**	**Description**
transactionID	Varchar	Unique identity for each transaction represented as the primary key.
avsResponseCode	Varchar	The code that identifies the result of comparing address verification information received in the transaction with address verification information contained in the cardholder address fields. Standard ISO values are acceptable.
cardPresentInd	Varchar	Indicates whether or not the card was present during the activity.
3DsecureResponseCode	Varchar	Online security protocol response code (VbV, SecureCode).
contactlessInd	Varchar	Indicates if the transaction authentication method was using RFID (as in DE22 - POS Entry mode).

Table 3.7 POSTransaction Entity Definition

Entity Name:	MerchantTransaction	
Attribute Name	**Data Type**	**Description**
transactionID	Varchar	Unique identity for each transaction represented as the primary key.
acquirerBIN	Varchar	Bank Identification Number (ICSA) for acquirer as an acquirer.
paymentTypeIndicator	Varchar	For clearings of payment transactions only, shows the type of payment.
electronicTerminalType	Varchar	For electronic terminals, distinguishes between e-Commerce and card-activity terminal transactions.
productTypeCd	Varchar	Indicates product type. (Optional from Private Data, clearings only.).
referenceData	Varchar	Acquirer reference data, needed for clearings only. Use to track to presentments.

Table 3.8 POSTransaction Entity Definition

Entity Name:	Session	
Attribute Name	**Data Type**	**Description**
sessionID	Varchar	Unique identity for each session represented as the primary key.
sessionType	Varchar	Type of session associated with the current transaction (i.e phone session, web session).

Table 3.1 Session Entity Definition

Entity Name:	PhoneSession	
Attribute Name	**Data Type**	**Description**
sessionID	Varchar	Unique identity for each session represented as the primary key.
phoneDeviceSournceNumber	Varchar	Source phone number of the call session.
phoneSessionAuthChannel	Varchar	The channel through which the customer authentication took place.

phoneSessionDateTime	Timestamp	The date and time that the session was started.
phoneSessionDuration	Double	The duration of the call session.
isInitiatedByCustomer	Boolean	Indicates whether the call is initiated by the customer and not an automated procedure, or an active call from the bank.
phoneNumber	Numeric	The standardized phone number associated with the call.

Table 3.2 PhoneSession Entity Definition

Entity Name:	**WebSession**	
Attribute Name	**Data Type**	**Description**
sessionID	Varchar	Unique identity for each session represented as the primary key.
headerAuth	Varchar	The relevant client session information from the http header data.
userAgent	Varchar	The relevant client session such as web browser type, plug-ins, etc.
ipAddress	Varchar	The client browser IP address that initiated the session.
loginName	Varchar	The online user of the Internet session. Usually, there is a one to many relationships between party and user.
secondFactorAuthInd	Varchar	Indicates whether the client passed a second factor authorization during the session (If applicable).
startDateTime	Varchar	The date and time when the Internet session began in the time zone of the local web server.
headerFields	Varchar	The components of the message header of the HTTP request that triggered the event. This should include both standard and non-standard Header Fields.
browserTimeZone	Varchar	The difference in minutes between the browser time and GMT.
screenResolution	Varchar	The screen resolution of the client browser used during the Internet session.

Table 3.11 WebSession Entity Definition

Entity Name:	**MalwareData**	
Attribute Name	**Data Type**	**Description**
ID	Numeric	Unique identity for each malware represented as the primary key.
malware_description	Varchar	Description of the malware found.

malware_type	Varchar	Type of malware found.
vendor_score	Double	Score provided by the vendor on the malware.
vendor_tag	Varchar	Additional tags provided by the vendor.
vendori_ID	Varchar	Unique id for the vendor.

Table 3.12 Malware Entity Definition

Entity Name:	**Device**	
Attribute Name	**Data Type**	**Description**
deviceID	Varchar	The unique device identifier generated using cookies.
deviceFirstSeen	Date	The date a transaction was first performed with this device.
deviceLastSeen	Date	The last date a transaction was performed with this device.
deviceBinding	Varchar	The client IP Address as detected by the system that the device is using.
proxyIPDetected	Boolean	Indication on whether a device is behind a proxy.
deviceScore		The probability that this device is really the device identified by the system. The higher the number, the greater the chance that the ID was correctly detected by the system.
Motion detection	Varchar	Raw data that provides information on the motion (only for mobile).
Rotation detection	Varchar	Raw data that provides information on the rotation of the device (only for mobile).
Device position	Varchar	Geo information about the device.
Pressure point	Varchar	Raw data that provides information on the pressure point of the device (only for mobile).
authUsed	Varchar	Authentication used by user on the phone (i.e. finger scan, facial, PIN, etc.).

Table 3.13 Device Entity Definition

Entity Name:	**Party**	
Attribute Name	**Data Type**	**Description**
partyID	Varchar	Identifier for a customer.
partyName	Varchar	The party's name.
onlineServiceJoinDate	Timestamp	The date the party signed up for the institution's online service. Any standardized date format is acceptable, as long as all date formats are the same.

passwordUpdateDate	Timestamp	The date the party's password was last changed (from ANY channel).
partyDOB	Varchar	The party's date of birth
Address	Varchar	The party's address
partyAddressUpdateDate	Timestamp	The date that the party or one of his related accounts had the last address change (from ANY channel).
partyEmail	Varchar	The primary e-mail address associated with the party.
partyEmailUpdateDate	Timestamp	The date that the party's e-mail address was last changed (from ANY channel).
partyInfoUpdateDate	Timestamp	The date that any party's information was last changed (from ANY channel).
partyLoanInd	Boolean	Indicates whether the party owns a loan product.
partyMobilePhone	Varchar	The first mobile phone number. Any standardized format is acceptable.
partyMobileUpdateDate	Timestamp	The date the party's mobile phone number was last changed (from ANY channel).

Table 3.14 Party Entity Definition

Entity Name:	**Card**	
Attribute Name	**Data Type**	**Description**
cardBin	Numeric	The bank's identification for the PAN.
activationDate	Timestamp	The date the card was activated. Any standardized date format is acceptable, as long as all date formats are the same.
AmtLimit	Double	The maximum daily withdrawal amount for the card type.
iccCardInd	Boolean	Indicates whether this card is an integrated circuit card (chip card).
issuedDate	Date	The date the card was last issued.
status	Varchar	A code that indicates the current status of the card (i.e., lost, stolen, etc.).
expirationDate	Date	The date embossed on the card beyond which the card must not be honoured. Any standardized date format is acceptable, as long as all date formats are the same.

Table 3.15 Card Entity Definition

Entity Name:	Account	
Attribute Name	**Data Type**	**Description**
accountID	Varchar	The unique identifier for the account.
availiableBal	Double	The available balance consisting of the payee account balance affected by this transaction plus the payee's overdraft limit / Available Credit / etc. (i.e. after the current transactions).
Branch	Varchar	The branch of the payee account.
accountNumber	Numeric	The payee Account Number or IBAN.
openDate	Date	The date that the payee account was opened.
overdraftLimit	Double	The overdraft limit of the account affected by this transaction. The value should be either 0 or a negative value.
Status		

Table 3.16 Account Entity Definition

Statistical relational learning (108) will be applied to understand the statistical strengths of the cross-channel data, this technique will be further explored during the implementation phase.

3.2.2. Data Normalisation

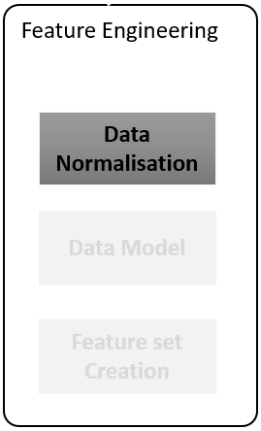


Figure 3.12. Feature Engineering - Data Normalisation

As shown in Figure 3.12, this section focuses on Data Normalisation technique. From the data model above, the input data will be from the different range of values and types. These range of values could vary widely, that could impact the model performance. Data normalisation

will be carried out by fine-tuning the inputs variables to bring the entire probability distribution of fine-tuned values into alignment. Different data normalisation techniques such as shifting, and scaling will be applied to eliminate the effects of certain gross influences. For example, for some of the numeric values such as amount averages can be applied, rescaling and decorrelation techniques can be applied so that input sequence data can be easily modelled by function approximators such as Gaussian Mixtures models (109) to provide smoother distribution where mean is centred and the standard deviation is around 0.

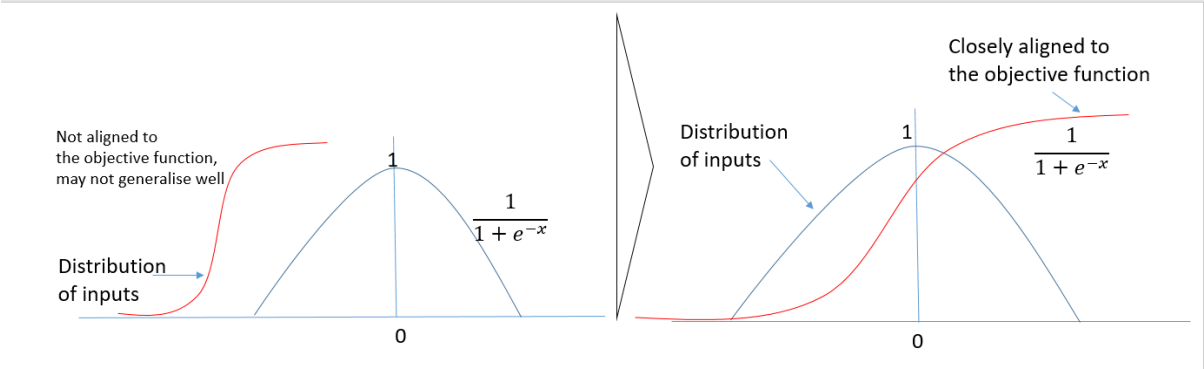


Figure 3.13. Data Normalisation requirement

As seen in Figure 3.13 above, the input distribution needs to be fine-tuned so that it is aligned to the objective function for a model. The data normalisation technique will help the model to generalise well.

Another approach is to create profiles and augment that to the input sequence to reduce long range contextual dependencies to short range (19) (110). For example, a profile of a customer can factor different amount ranges (min, max and average) for different timeframes and append this information to the input feature.

Rescaling and decorrelation will need to be applied to the raw input prior to feeding data to the model. One well known technique is known as PCA (Principle Component Analysis) (111). Fundamental components for PCA are adders, delayers, multipliers. These components are similar to neural network component called neuron where inputs are multiplied by weights and linear combiners are in the form of activation potential.

For example, one of the input parameters (i.e. age) can have a range from 1 to 100. While another input parameter such as transaction amount may range from £1 to £100,000. When

these two inputs are provided to a machine learning model (e.g. RNN) would result in a large activation function output. Since weights are fixed, multiplying these two inputs with fixed weight will produce large activation function. I.e. amount coefficient to be very large compared to age coefficient, resulting in poor model performance.

Data can be rescaled and reshaped so all inputs are centred around 0, however, this will produce input vector where data is correlated. To decorrelate the data PCA can be applied (or data whiting) (112) by taking the average and the difference between the above two inputs.

PCA definition can be given as:

$$PCA = matrix\ mutiplication(scale\ and\ rotate) * matrix\ addtion\ (shift) \quad \text{(eq. 3.4)}$$

For example

$$[newAge \quad newAmount] = [oldAge \quad oldAmount] * \begin{bmatrix} 0.05 & 0.13 \\ 0.1 & -1.2 \end{bmatrix} + [-1.2 \quad -1.6]$$

During the pre-processing phase, min/max standardising technique can be used as given by the formula below (7):

$$X_{new} = \frac{X^{old} - \min(X_{old})}{\max(X_{old}) - \min(X_{old})}(newmax - newmin) + newmin \quad \text{(eq. 3.5)}$$

Where, $newmax\ and\ newmin$ are the new values.

A recent work has shown that (113) by adding more hidden layers can achieve the similar results to PCA whiting by letting the network determine the right level of whiting is required. A technique is known as batch normalisation (113) which is applied between the layer to renormalize the activation function before feeding the input to the next layer. The input to the next layer is automatically centred and scaled. The equation below forms the basis for batch normalisation:

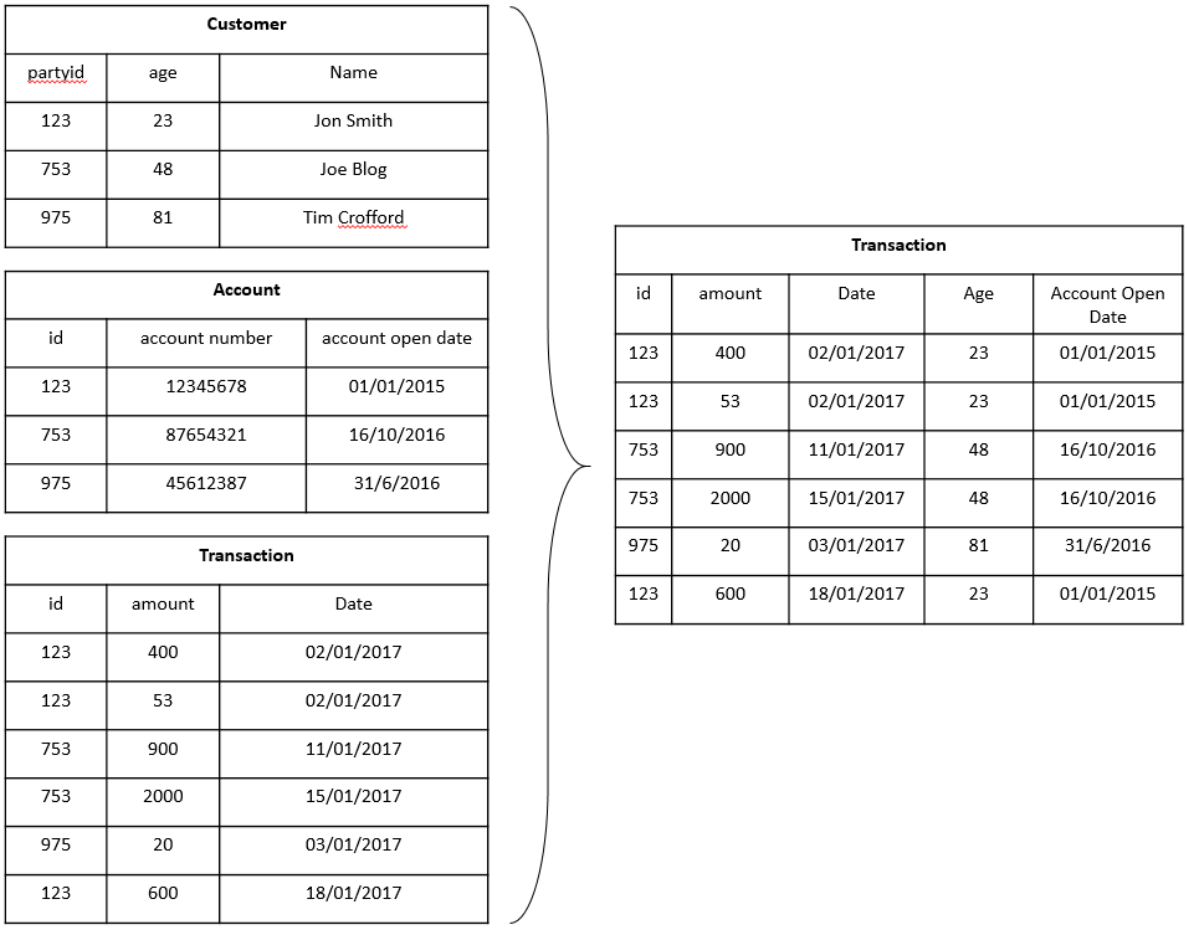
$$\hat{x} = \frac{x - avg_{batch}(x)}{stdev_{batch}(x) + \in} \quad \text{(eq. 3.6)}$$

Where statistics on logits such as weighted sum and biases are computed in batches and rescaled by subtracting the average. Furthermore, it is then divided by standard deviation before the activation function.

3.2.3. Feature set creation

Figure 3.14. Feature Engineering - Feature Set Creation

As shown in Figure 3.14, this section focuses on feature set creation. Feature set creation is a process of selecting a set select of features that are relevant for the model to perform efficiency. The technique of RFM will be used to view, create non-normalised data that can be used to facilitate further analysis. For example, transaction, personal and account tables will be merged into a single non-normalised data table and processed through for feature extraction as shown in Figure 3.15 below:



Customer		
partyid	age	Name
123	23	Jon Smith
753	48	Joe Blog
975	81	Tim Crofford

Account		
id	account number	account open date
123	12345678	01/01/2015
753	87654321	16/10/2016
975	45612387	31/6/2016

Transaction		
id	amount	Date
123	400	02/01/2017
123	53	02/01/2017
753	900	11/01/2017
753	2000	15/01/2017
975	20	03/01/2017
123	600	18/01/2017

Transaction				
id	amount	Date	Age	Account Open Date
123	400	02/01/2017	23	01/01/2015
123	53	02/01/2017	23	01/01/2015
753	900	11/01/2017	48	16/10/2016
753	2000	15/01/2017	48	16/10/2016
975	20	03/01/2017	81	31/6/2016
123	600	18/01/2017	23	01/01/2015

Figure 3.15. De-normalised data structure for faster analysis and processing

The data will be presented in an offline feed for training the network and as an event in real time as they appear in the channel. The tables 5 to 17 listed above highlights a list of attributes that are available for the analysis.

Once the data set is created, the next step will be to go through a variable selection process. Various techniques are used to reduce variables to a manageable subset before model construction is undertaken (7). Using real world data, there will be a vast number of data points that may result in a vast number of features. There is a need to reduce/ amalgamate the variables by carefully selecting interesting variables, both to ensure good system and model performance. One of the techniques is to use filters which examine each variable in turn by measuring and calculating the strength of the relationship between the predictor variable and the modelling objective (output). The only variables that are taken forward that exhibits strong correlation. Two well-known techniques used for filtering are Pearson

correlation and Fisher score. One more technique often use is called Cramer's V filter which is based on the Chi-squared analysis (7).

Fraud detection is an imbalance class problem. The majority of the bank's events are genuine with a proportion and are executed by fraudsters. There will be a need to come up with a strategy to rebalance the dataset and ensure a balance of classes (fraud / not fraud) is maintained during the learning process. Several strategies have been defined such as ensemble, cost-based, distance-based, sampling (114). The sampling approach will be used to take a subset and use that to build an analytical model while avoiding the sampling bias. In fraud detection, the typical problem with sampling is that customer behaviour will change from time to time and this may introduce bias in the model which could result in poor performance and skew the results for the actual test. One technique (105) is to use Synthetic Minority Over-sampling technique. This technique will oversample the data and then applying the classifier that achieves the best False Positive (FP) and True Negative (TN) rates. The experiment conducted using SMOTE technique achieved 96.7% TN and 4.5% FP. Another approach is to make use of the stratified sampling, where the sample will contain the same amount of data that represents both classes for fraud and not fraud.

Looking only at the individual transaction for classification may not be enough since, it does not capture the customer behaviour, especially across multiple channels. To deal with this there will be a need to include time as a critical element to the feature set. In addition to time new latent features such as previous amount, pervious ip-address and others across different time frames will be introduced. During the implementation phase, transaction aggregation strategy will be considered and implemented (115). This methodology will allow us to create a group of raw features based on time (i.e. a number of hours, weeks, months) and then create new sets that are based on distance algorithm such as time elapsed between customer first register for online banking and made the first payment. It will also allow the creation of further statistical counters such as a number of transactions during the last hour, total spend in the last hour and others.

As seen in the table below there are different data types associated with different attributes such as numeric, categorical, nominal. Since the input to the model is numeric values, there is a need to derive categorical and nominal values into numeric values. Thus, creating latent

variables that will form part of the feature vector (116). For example, one of the technique known as one hot encoding (117) can be applied for categorical values such as transaction type below by listing down all the different transaction type and updating the transaction table as shown in Figure 3.16 below:

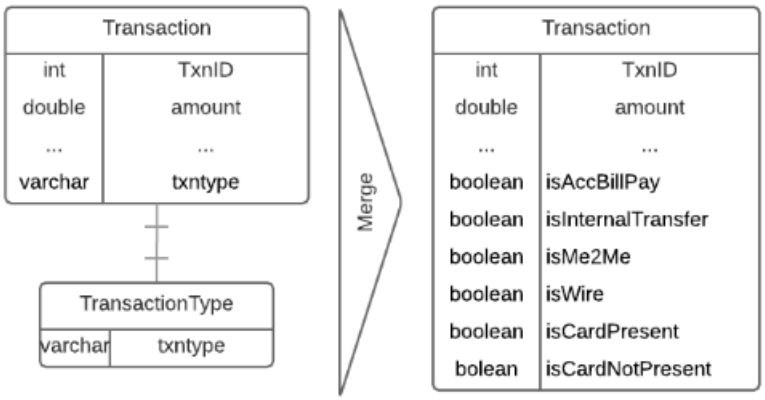


Figure 3.16. Example of One Hot Encoding

It is noted that some of the categorical value such as account number cannot be represented using one hot encoding (i.e. there could be millions of account numbers). These values will be treated as continuous values. The representation such continuous values will be tackled during the later phases.

Finally, in order to reduce the redundant features regularisation techniques will be applied. A lot of features with a limited set of data causes the training data to overfit (24). Regularisation is applied by adding a penalty to the error function so that it limits the large variations to the features. From a Bayesian, probabilistic standpoint this is equal to applying prior knowledge to the features so that weight mass doesn't scatter too much. As defined by Bishop (24) the simplest form is known as the sum of squares of all the features so that the sum of the equation becomes:

$$E = \frac{1}{2}\sum_{n=1}^{N}\{y - \hat{y}\}^2 + \frac{\lambda}{2} \parallel w \parallel^2 \qquad \text{(eq. 3.7)}$$

Where, $\parallel w \parallel^2$ are the coefficients and λ is the regularisation term. The two standard regularisation methods (118) are:

- L1 Regularisation – minimises the sum of the absolute values.

- L2 Regularisation - minimises the sum of squares of the absolute values.

According to the research paper proposed by Andrew (118), applying L1 regularisation causes many parameters to result to zero and hence a considered a preferable candidate for feature selection as it speeds up the learning process.

The topics covered in this section can be summarised as:

- A need to ensure data is coherent and correlated. This will be achieved by normalising the data according to the data model and followed by flattening of data for easy of processing.
- Sampling and rescaling techniques will then be applied to ensure the ratio between fraudulent and non-fraudulent is consistent. This will ensure the feature selection consists of a set of features from both fraud and not-fraud population.
- Applying data whiting techniques such as PCA to derive a set of relevant features (both latent and non-latent features) that can be easily processed by the model.
- Finally, dimensionality reduction technique such as L1 regularisation so that only the relevant features are selected.

3.3. Neural network model

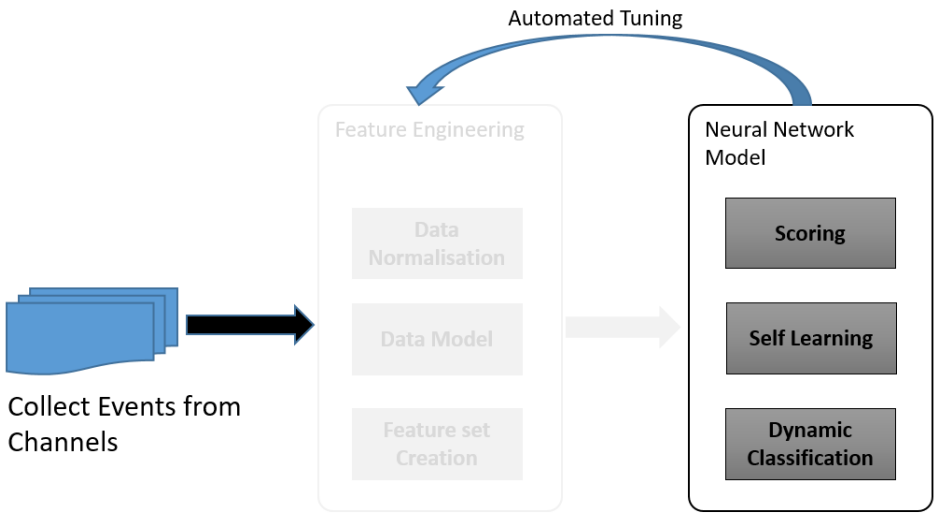


Figure 3.17. Neural Network Model

The findings as per chapter two, neural networks can be used to build models that represent non-linearity efficiently. This efficiency can be exploited in terms of its ability to detect complex nonlinear relationships between dependent and independent variables. The research also showed neural networks is the most commonly used method for developing fraud models for the banks. As shown in Figure 3.17, Neural network model will be used to for scoring and classification of the incoming event. The model will be based on the supervised learning. That means in the model development environment, the model will be trained on the labelled data by comparing its output with expected output and making the relevant adjustment to minimise the overall error and learning the inherent properties of the input data. Subsequently classifying the inputs when deployed in a production environment as depicted in Figure 3.2.

The fundamental assumption for the input to be statistically nonstationary so that a dynamical system can be implemented that can have the following properties:

- A set of adjustable parameters (often referred to as weights) as the inputs to a non-linear function so that it can capture dynamics of time varying inputs.
- A block for error calculation, calculated as the difference between the actual output and the predicted output.
- A control (learning) algorithm for the adaptation of the weights.
- Ability to fuse data together so that it can be represented in both qualitative (symbolic – represented by a data model) or quantitative (engineering – represented by a feature vector) form.
- Categorisation of the events into two class:
    - Class A – Fraud: This event will be marked by an expert and will be used as a feedback for auto-tuning as well as during the learning process.
    - Class B – Not Fraud: events not marked as fraud.

This assumption is valid because the events are continuously produced by the bank's customers and feed to the dynamical system via channels for fraud detection. The nature in which the events are generated means one cannot assume the events to be identically and independently distributed (i.i.d). For example, the customer spend patterns on the digital channel will have a different distribution to the spend patterns on the cards channel.

However, there is a requirement that the data from different channels needs to be cross pollinate to form a strong correlation and improve the overall detection. For example, when making payments from a digital channel, one would expect the payment process to be completed in the digital channel. Furthermore, one do not expect a credit card payments to be initiated in parallel.

Based upon the above, the model will need to have the following capabiliites:

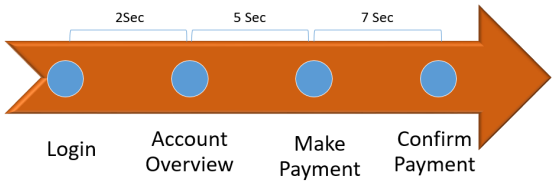
- Unconstrained input space to cater for variable input length for corss-pollinated data.
- Ability to easily include contextual information based on past events.
- Robust to noise, as the events unfold over time (119).

The research (119), (114), (120), (121) have highlighted a Recurrent Neural Network as the model that has above capabilities that will make the network desirable for fraud detection domain. Furthermore, a lot of research is done on the image and video processing using multi-dimensional RNN as they form a directed acyclic graph that generalises well for multidimensional data (119) which could be useful.

3.3.1. Scoring, SelfLearning and Classsification

At the heart of the neural network model that will perform scoring and classification by sequence learning the temporal patterns of the entire customer history. Recurrent Neural Networks algorithm will be used as it can classify individual transactions as fraud or not sequences of events. The Universal Approximation Theorem for recurrent neural network highlights that given many hidden units can arbitrary map input sequences to output sequences with reasonable accuracy (120). The usefulness of this theorem can be shown in the following examples that highlight how RNN can be conceptually modelled to arbitrary map input sequences and to perform fraud detection:

Example 1: Normal Customer journey for online banking

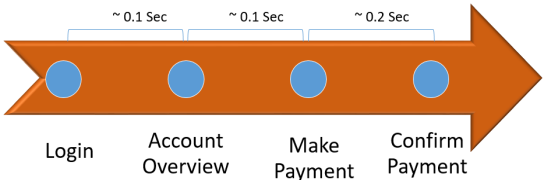


Normal customer behaviour where the following can be measured:

- Information about the event.
- Time spent at the event.
- Latency introduced while moving from one event to another.

The above diagram depicts a user navigating from one page to another on the bank's online portal. For example, time taken for a user to log into a bank and navigate to Account Overview is approximately 2 seconds. Subsequently, the user navigates from Account Overview page to Make Payment page in approximately 5 seconds and so on.

Example 2: Malware journey for online banking

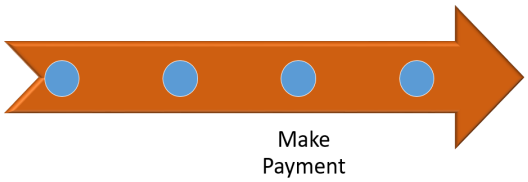


The following can be measured:

- Information about the event.
- Time spent at the event.
- Latency introduced while moving from one event to another.

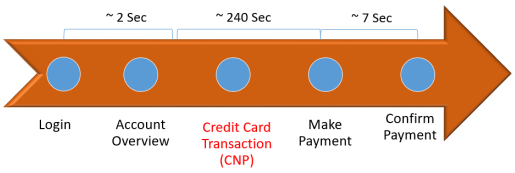
The above diagram depicts a malware journey for online banking, where malware logins and subsequently goes through the same events as a normal user in quick succession. Time spend on each event and in-between event is minimum.

Example 3: Non-Repudiation attack on online banking



A non-repudiation attack is when only the make payment event is observed. This is also known as the "man in the middle attack" where an attacker captures the event and replays the over after a while.

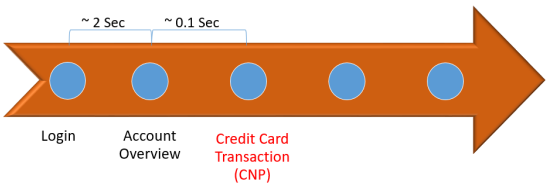
Example 4: Normal customer behaviour – cross-pollination



Normal customer behaviour where card not present transaction is introduced when logged to the online banking.

In the above example, customer login to the online banking in order to make a payment. However, before the make payment event is completed, the customer also initiates a card not present transaction. By measuring the elapsed time between Account Overview and Make Payment, certain inferences can be made. For example, the time elapsed is large for the customer to initiate a card not present transaction while in the process of making a payment. Contextual attributes between the different events can be considered by making an inference.

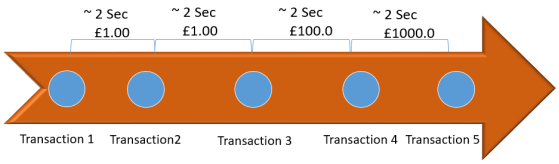
Example 5: Card cloning attack – cross-pollination



An example is where normal behaviour is followed immediately by unusual behaviour.

In the above example, a customer does login event and moves to account overview event as normal, which is most immediately followed by a card not present transaction. This might be an example of card cloning and zero-day attack which can be quickly determined by the scoring model.

Example 6: Card Fraud – velocity fraud



Velocity fraud is when fraudster carries out few transactions for the low amount to test the card details are valid, followed by high value transactions until the card is blocked or no funds are available.

As seen in the above examples, by providing a temporal dependency for sequences of events, the model can store customer behaviour based on its past input and outputs as "memory" to influence the current network output for a given event.

This can be further elaborated mathematically as a set of all possible events defined by *I,* that can be possible by a customer $\{i_1, i_2, i_3, i_n\}$ from distribution $D_{\varkappa xl}$. Every I_n can belong to a class with a label of fraud or class with not fraud. The input space $I = (\mathbb{R}^m)$ is the sequences of size M real value vector. Let the output space $C = L^n$ be the sequences of L labels that can take the class values {fraud, not fraud}. The derived variables called features will be calculated. Let X be set of all possible features $\{x_1, x_2, x_3, x_n\}$ drawn from the discrete time event $y(t)$. The objective is then to use I to train fraud detection algorithm $h: X \rightarrow C$ to label outputs in a test set $I' \subseteq D_{\varkappa xl}$ in a way that minimises the error measure. The representation of the typical RNN is shown in Figure 3.18 below:

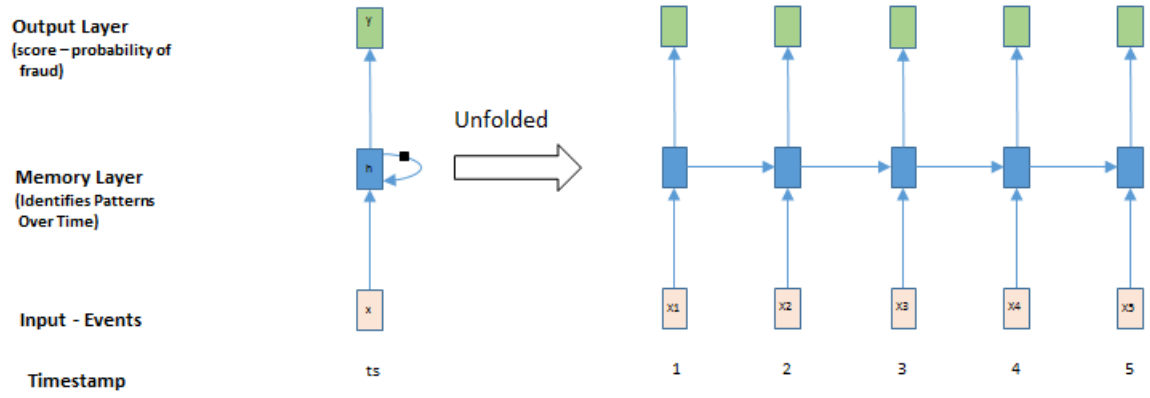


Figure 3.18. RNN Architecture

The RNN architecture will be like an ANN architecture but weights will be shared across evolving time stamp. RNN consists of an input layer that contains event at discrete time stamp (ts). The number of nodes will be created based on the number of features in a feature set (x), the number of hidden nodes (h) will be based on trial and error and the output will consist of a single node (y).

For online banking the input feature vector will be:

$$X_t = (F_t)^T \qquad \text{(eq. 3.8)}$$

Where F_t is the feature matrix as defined below:

$$X = \begin{bmatrix} id & latency & CFV & & & \\ id & latency & CFV & AFV & & \\ id & latency & CFV & AFV & TFV & \\ id & latency & CFV & AFV & TFV & \dots \end{bmatrix}$$

Where:

- id - state transition id. For example, login page, authentication page, landing page, view statements page, view account page, make payment to new beneficiary, make payment to existing beneficiary, change of details, confirm payment, logout.
- latency – is the time difference between the page navigation.
- CFV – context feature vector. For example, ip address, malware flag.
- AFV – authentication feature vector. For example, authentication type.
- TFV – transaction feature vector. For example amount, transaction type, payeeaccount number, etc.

For credit/debit card the input feature vector will be:

$$x1 = [state\ id \quad ip\ address \quad amount \quad time\ stamp \quad merchant\ id \quad account\ no]$$

The state of the currently hidden nodes will depend on some objective function that takes in the state of the pervious hidden nodes and input vector at time t.

$$h_t = f_w(h_{t-1}, x_t) \qquad \text{(eq. 3.9)}$$

Where h_t is the new state, h_{t-1} is the old state, x_t is the input at time t. This provides a feedback into the network and can be considered as introducing memory to the network. Since, the network can hold memory it can be better utilised for classification and prediction tasks, when compared to other neural networks. Such a feedback at the hidden layer is referred to as local feedback, global feedback is out of scope for this thesis (121). If the tanh is used as an objective function f_w for the hidden layer, then the above equation becomes:

$$h_t = (\tanh(w_{hh}h_{t-1} + w_{hh}x_t)) \qquad \text{(eq. 3.10)}$$

Tanh is a logistic function that is non-linear and continuously differentiable. This enables the study of the properties of neural networks for developing learning algorithms.

The values of the output layer y(t) are predicted based on the current input, weighted by the coefficients w_i of all the past input. The aim is to minimise the error by incrementally adjusting the parameters of the algorithm to optimise the objective function h_t during training. The prediction error, e(t) thus becomes:

$$e(t) = y(t) - \hat{y}(t) \quad \text{(eq. 3.11)}$$

While for regression models, typically the sum of squared error or squared Euclidean distances is used. For parametric models, it is generally mean squared error that is used, E$[e^2(t)]$, where E[.] represents the statistical expectation operator and $\{y(t)\}$ is assumed to be statistically independent (122).

The classifier output node will produce 1 if fraud is detected or 0 otherwise. A probabilistic classification approach is preferred to calculating conditional probabilities $p(classFraud|x)$. This will allow probabilities to be retained in a consistent manner. This classifier will accept x_i describing input I_i and will produce a probability score that belongs to a class C_i. The output will be based on the SoftMax function that output a cross-entropy loss and given by the equation:

$$P(Y = fraud\ |x_t) = \frac{e^{w_{hh}x_t}}{\sum e^{w_{hy}x_t}} \quad \text{(eq. 3.12)}$$

Based on the above equation the output can be given by:

$$0 \leq P(i_i \in C_n|x_i) \leq 1 \quad \text{(eq. 3.13)}$$

For example, the output of the above equation will be between 0 to 1 outlining the probability of the event. If the probability is above a certain threshold the system will output 1 as fraud else 0.

In order to learn the sequences of events and their classification accurately, an error needs to be calculated and propagated back through the network using the gradient decent algorithm known as Back Propagation Through Time (BPTT) (123). The advantage of using BPTT is that

linear and non-linear constraints between the shared weights can be easily incorporated. It will allow the model to generalise well to temporal dependencies of different states. For examples, let's say one started off satisfying the following constraints $h_1 = h_2$ and one need $\Delta h_1 = \Delta h_2$ then one can compute the following:

$$\frac{\delta E}{\delta h_1} \; and \; \frac{\delta E}{\delta h_2}$$

Then the following partial derivatives can be used:

$$\frac{\delta E}{\delta h_1} + \frac{\delta E}{\delta h_2} \; for \; h_1 \; and \; h_2 \qquad \text{(eq. 3.14)}$$

Using the above equations, one can start to build forward pass that determines user state at each timestamp and followed by a backward pass by computing the error derivatives at each time stamp. One can then add together the derivatives of all the time stamp for each weight and then change all the copies of the weight by the same amount which is proportional to the sum of those derivatives as depicted in Figure 3.19 below:

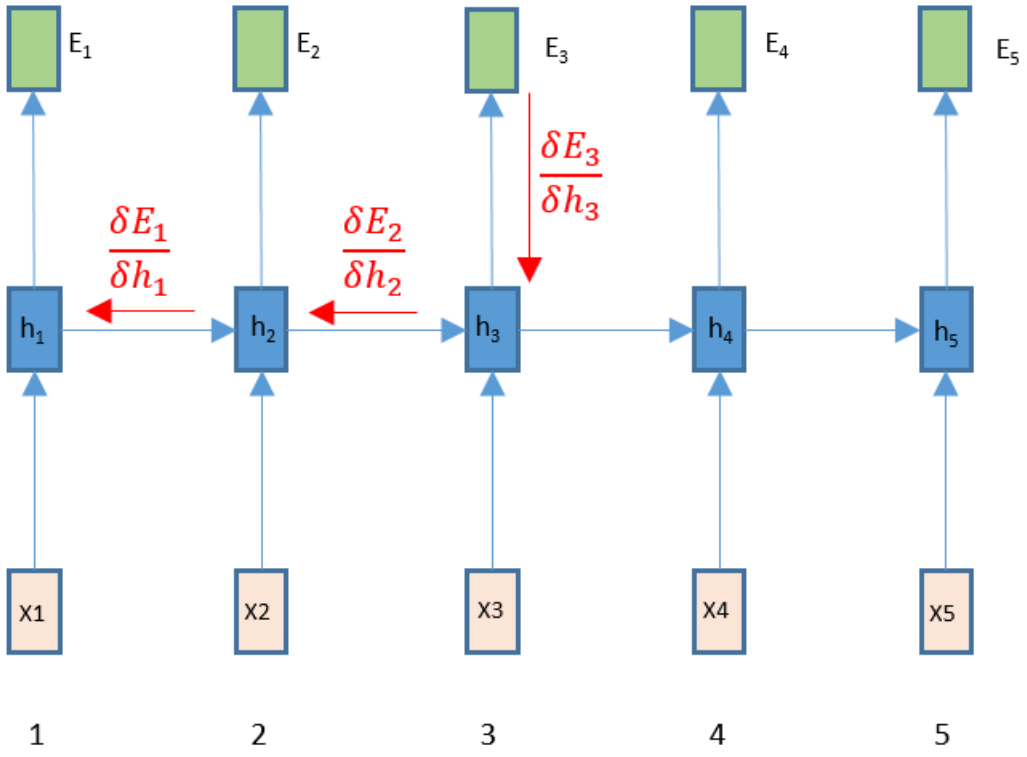


Figure 3.19. Back Propagation Through Time for RNN

The above diagram can be summarised as:

1. Events are passed into the unfolded time and error is calculated based on some constraint at each time stamp. For example, when an event is misclassified as legal while it is fraud.
2. Error is then fed back into the network and weight deltas are calculated and associated weights are updated by calculating the sum of all the deltas.
3. The process is repeated for each training epochs until the desired error measure is achieved.

Another approach to propagating error is known as real time recurrent learning (122). In this process, the gradient information is calculated in real time as the inputs are presented to the network. This will be useful when the fraud model will be required to be updated in real time as a self-learning approach.

During the implementation of the model, the model will be first trained using the batch learning approach and once the model is sufficiently trained, online learning method for self-learning can be used. The batch learning approach will be used in the model development environment and on the historical data in order for the model to quickly converge as learning can be done in batches. In order for the model to be adaptive to new fraud MOs in real time, online learning approach will be used in the model execution environment (production). This was as the new fraud emerges, the model's internal state will be updated along with detection, hence providing the self-tuning aspects of the framework.

The steps for the batch process can be summarised as:

- Randomly initialise all the weights.
- Iterate over training data based on a particular pattern (batch size).
    - Pass input data through, based on the batch size.
    - Calculate the mean sum of errors after each run based on the expected output vs the currently predicted output.
    - Update all the weights based on the chain rule by calculating the error difference.
    - Stop the process when some threshold in error performance is reached or no other further improvement is observed.

As discussed in the several research papers (119), (123), both above algorithms fail to learn where there are long-time lags between the input events and their associated errors flowing back through time or when the network grows from t to t+n. As discussed in the paper published by (123), the error derivatives of the hidden layer consist of Jacobin maps. These maps will consist of eigen values, which when goes through differential equations resulting in values converging to zero when values are less than one. This is called vanishing grading problem. However, if the values are greater than zero, they are pushing away from optimal minima resulting in exploding grading. These issues of exploding and vanishing gradients was addressed by Hochreiter and Schmidhuber by inventing a new RNN model called Long Short-Term Memory (LSTM) (124).

The LSTM network can be used alongside hidden layer or can be used as a hidden layer to overcome the vanishing and exploding gradient problem. The properties of the LSTM memory block are outlined in Figure 3.20 below:

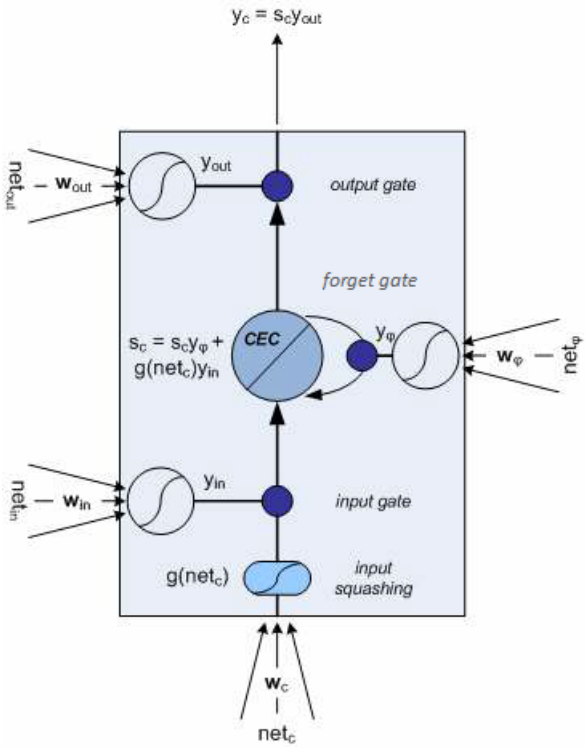


Figure 3.20. - Long Short-Term Memory

Network cell description is outlined in Table 3.17 below:

Component	Description
The Constant Error Carrousel (CEC)	The purpose of CEC is to store information over a long period. The core function of CEC is to act as a memory function.
Input Gate	The purpose of the input gate is to control the input to a memory cell. Its core function is to allow data into the memory cell and to protect the memory cell from noise from other network units.
Output Gate	Similar to the input gates, the purpose of this gate is to control access to the memory cell's content. Its core function is to protect the memory cell from noise.
Forget gate	The purpose of this gate is to forget the memory the cell has acquired during earlier time stamp as the content becomes irrelevant.
Peephole connections	Provides further waited connections that connects a memory cell CEC with the memory block gates to provide additional feedback.
Memory blocks	Multiple memory cells are grouped together to form memory blocks. In memory blocks, input and output gates are shared. Each memory block consists of CEC to ensure constant error flow even if there are no inputs or error flowing through. It is this property of the LSTM network that solves the vanishing gradient problem (123).

Table 3.17 LSTM Cell Definition

This section highlighted a potential use of RNN to improve the cross-channel fraud detection. RNN seems to be the natural fit as it can efficiently deal with temporal dependencies of the sequence classification problem which can be used for dealing with events from internet

banking that is part of an internet banking session as well as a chain of card events from POS and e-merchant terminals. The recent advancement in RNN such as LSTM has resulted in a significant improvement to the network overcoming the vanishing and exploding gradient limitation. This means that modelling of events over a long period of time to truly understand customer behaviour over multiple channels can now be implemented.

3.3.2. Framework Performance Evaluation

This section presents our initial ideas for evaluating the effectiveness of our proposed methodology. A common method for testing, the framework efficiency is to measure the expectancy ratio such as:

- True Positive Rate (TPR): legal events detected as legal. The model correctly classified not events as not fraud correctly.
- False Positive Rate (FPR): fraudulent events detected as legal. i.e. model misclassified fraud as a not fraud.
- True Negative Rate (TNR): legal events detected as fraudulent. i.e. model misclassified not fraud as fraud.
- False Negative Rate (FNR): fraudulent events detected as fraudulent. i.e. model classified fraud events as fraud.

The measure generally is given by confusion matrix as shown in Table 3.18 below (125):

		True Class (y_i)	
		Fraud $y_i = 1$	Not Fraud $y_i = 0$
Predicted Class ($\hat{y}_i$)	Fraud $\hat{y}_i = 1$	TP (True Positive)	FP (False Positive)
	Not Fraud $\hat{y}_i = 0$	FN (False Negative)	TN (True Negative)

Table 3.18 Fraud Confusion Matrix

Once the confusion matrix is produced, the following metrics can be calculated:

$$\text{Misclassification: } 1 - \frac{TP+TN}{TP+TN+FP+FN} \quad \text{(eq. 3.15)}$$

$$\text{Recall: } \frac{TP}{TP+FN} \quad \text{(eq. 3.16)}$$

$$\text{Precision: } \frac{TP}{TP+FP} \quad \text{(eq. 3.17)}$$

$$\text{F}_1 \text{ Score} = 2\frac{(Precision \times Recall)}{(Precision+Recall)} \quad \text{(eq. 3.18)}$$

Subsequently, the false positive rate and true positive rate can be calculated as follows:

$$FPR = \frac{FP}{FP + TN} \quad \text{(eq. 3.19)}$$

And

$$TPR = \frac{TP}{TP + FN} \quad \text{(eq. 3.20)}$$

The aim of the model is to lower the False Positive Rate and True Negative Rate and increase the True Positive Rate and False Negative Rate.

The above two metrics can be plotted against each other to calculate a curve called Receiver Operating Curve (ROC) as shown in Figure 3.21 below:

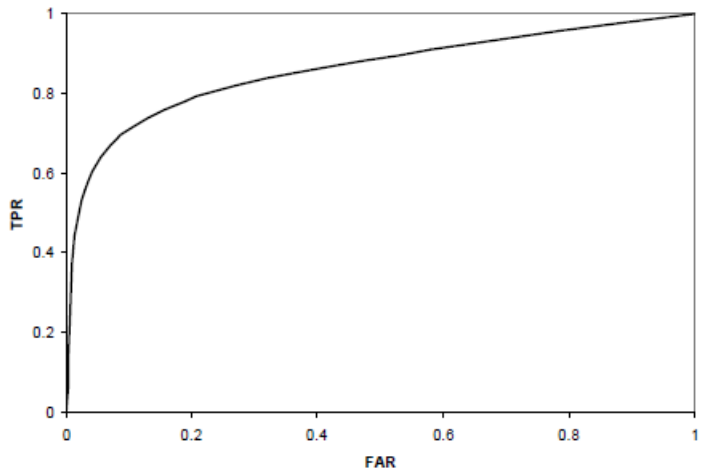


Figure 3.21. Example of a ROC

In general terms, an interest is in the section called Area Under the Curve (AUC) for the ROC above which gives the indication on the generalisation of the model. The higher the curve the better the model performance.

ROC will be used as a measurement for all the experiments that are conducted. For each experiment, FRP and TPR will all be presented.

The activity diagram below provides an overview of the learning process and real time scoring and decision-making process will work.

3.4. Synthetic data

3.4.1. Introduction

Fraud detection on the retail channel is not a trivial task, to classify transactions as either legal or fraud in real time is difficult and complex. While machine learning techniques can be applied to learn from past experience, Chapter 1 and Chapter 2 describes the problems in detail that affect the performance parameter for any machine learning classifier in fraud detection domain. This is summarised as:

1. To detect fraud, it requires complex statistical models and hence requires a large amount of data that is not readily available.
2. There is also a class imbalance problem and unlabelled data problem, resulting in reduce accuracy for any classifier.
3. Due to the proprietary nature of the data around fraud losses, there are legal issues resulting in no standard benchmarks on the accuracy.
4. Privacy concerns around the PII data leakage, resulting in companies not openly willing to share data.

Often, in order to compensate the above concerns, an approach to lean towards the synthetic data generation is taken to determine the effectiveness of new algorithms. This section provides a novel approach of generating synthetic, with an aim to generate data quickly and efficiently that is a close representation of the real data. By a close representation meaning, keeping the statistical relationship intact while, increasing the volume of data. The goal is to identify control measures, given a set of conditions to generate new data, where is it difficult for a machine learning algorithm to differentiate between generated data and real data. An approach described here, is to examine a set of common sequences of events that a user will carry out on a retail channel. Then using this data generate a large amount of synthetic data

that is used for training, testing and evaluating purposes. This approach aims to preserve the statistical inherence properties of the original data and generate additional data that forms a similar nature to original within certain bound of error confidence and hence reduce the overfitting problem in order for models to generalise better. It is necessary to define a list of good characteristics on what this data should like, which this section will aim to capture:

- Data is clean and accurately labelled
- Large dataset to avoid underfitting and overfitting problems
- Balanced labelled data to avoid any biased creep

3.4.2. Related Work

Very few papers discuss the use of synthetic data for fraud detection. Most of the papers used real data for training, testing and evaluating purposes. With the advent of technology and explosion of data, it is now possible to capture micro behaviour data on individuals and the demand of such publicly available dataset is growing in the field of fraud detection. However, privacy and reputational damage of an organisation since remains major concerns.

Surendra H and Mohan H. (126) provides a detail review of the synthetic data generation methods for privacy preserving data. Their approach is to classify synthetic data into fully synthetic, partial synthetic and a hybrid. Fully synthetic approach is where all the data and the attributes are completely generated based on some hypothesis (i.e. normal distribution) and generation is not dependent on real data. A partial synthetic approach makes use of the real time data, but all the sensitive attributes are replaced by synthetically generated values. Hybrid approach is where the combination of fully and synthetic data is used to generate additional data. According to their survey, synthetic data generation based on fully and partial techniques are highly active research fields. However, a lot more attention is required in the hybrid space, this chapter will be focused on using the hybrid approach, where sensitive attributes will be transformed into a set of non-sensitive features and these features will be used to generate additional data.

One of the most commonly used approach to synthetic data generation for fraud detection is called Synthetic Minority Over-Sampling Technique (SMOTE). It was discussed in detail by Nitesh V. Chawla, et al. (105) and provides perspective around the use of SMOTE to construct

a dataset where there is a high imbalance class problem. A high level of accuracy is achieved when they used over sampling techniques from the minor label and under sampling techniques for the majority labels. Class priors then used with classifier such as Naïve Bayes and the ROC curves demonstrates that this approach outperforms other methods. However, a limitation with this approach is that it works well when there are only two class labels and additional classes will introduce complexity and will different clustering strategies be required to ensure the minority and majority sampling on multi-class is done appropriately. The most traditional approach to fraud detection has a label fraud and not-fraud and hence SMOT technique is quite useful. For the purpose of our study there are different fraud labels depending on the fraud scenarios and hence the use of technique will require a significant amount of time to evaluate. This approach is extended in (127) by Haibo He, et al. where they present approaches such as Adaptive Synthetic Sampling Approach for Imbalanced Learning (ADASYN) and Kernel Adaptive Synthetic Sampling Approach for Imbalanced Learning (Kernel ADASYN) (128). Their papers showed that by applying a mean weighted distribution, one can approach to a multi-minority class. Different weights can be applied around a cluster of class label by determining the level of difficulty in learning. This approach of weighted distribution helps reduce the class biased and adaptively fine tune the decision boundaries.

Emilie L, et al. (129) provides a comprehensive methodology around generating fraud data. The paper describes deriving synthetic data from original data based on the characteristics of the relationship between different attributes. This methodology generates a set of user profiles from data, followed by modelling of legal and attack flows. Legal and attack simulators are used to generate additional synthetic data. Their methodology is further outlined in (130), where the authors applied the methodology to detect fraudulent IP based video on demand service. Their approach proved that user and system can be modelled using synthetic data that maintains the statistical relationship among the attributes and then subsequently used to train a fraud detection system.

3.4.3. Data generation methodology

The data generation methodology provided below allows the repeatability of the data generation process as the complexity of the raw data and latent features increases. The main components are:

- The specification of user behaviour represents as events in the historical database
- The specification around features that needs to be created for user profile creation,
- An algorithm that reads the profiles and output synthetic data.

The data generation methodology provided in this chapter is a variant of the methodology provided by Emilie Lundin (129) and summarised in Figure 3.22 below:

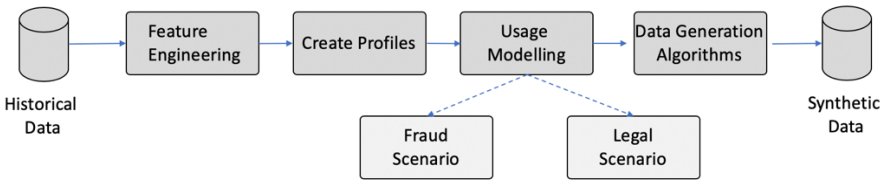


Figure 3.22. Data Generation Methodology

The figure above illustrates the methodology, the first step is to analyse the historical dataset that is the representative of the real dataset and contains a set of users and events (user journeys). The second step is to create a set of statistical latent features such as page navigation time. The third step is to create a set of profile for each customer and their sessions, in terms of page navigation time for each of the user journeys. The fourth step is to create different scenarios that need to be modelled (legal scenarios and fraud scenarios are modelled here as user behaviour). The fifth step is to implement an algorithm to generate synthetic data.

3.4.4. Analysis of the historical data

Data was extracted from a database of a big retail bank that contained retail transactional events. This is a representation of an Internet Banking dataset, where a lot of users would login to check their balance, make payment to existing and new beneficiaries and use services such as change password and consist of:

1. Fraud Label: There are 22,889 transactions from 305 unique users, with transactions labelled as fraud.
2. Non-Fraud dataset: There are 100,000 transactions from 68,425 unique users.

The number of parameters that are important to this research were analysed. The data was then rearranged in a way that allowed transition state to be created for each of the user's

session. This approach is then used to model user behaviour. An example of a number of sequences and for each user is Figure 3.23 below.

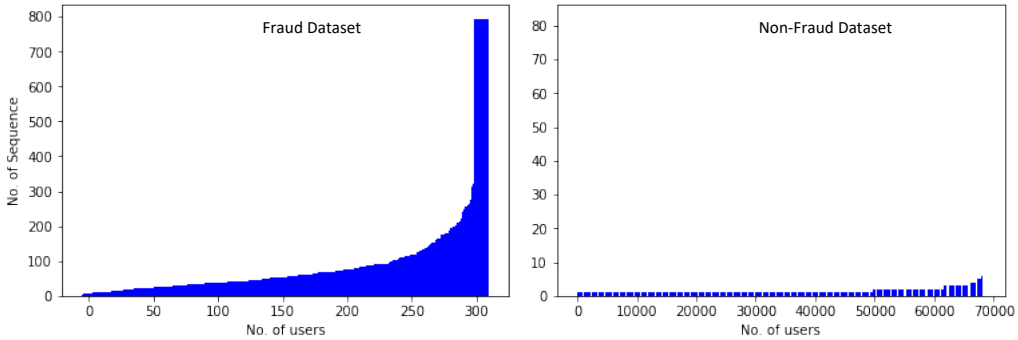


Figure 3.23. Number of Users & their Sequence of Events

As shown in the figure above the minimum sequence length is 4, maximum is 794 and an average length is 75. Compare that to the non-fraud dataset, the minimum sequence length is 1, maximum is 82 and average is 1.5. This is due to the data that was extracted from the database, this disproportionate sequence length makes it hard for sequence analysis to occur. The order to prove the hypothesis that data generation technique can be applied, a simpler approach is required.

Figure 3.24 below provides an overview of the most common sequences that are found across both datasets.

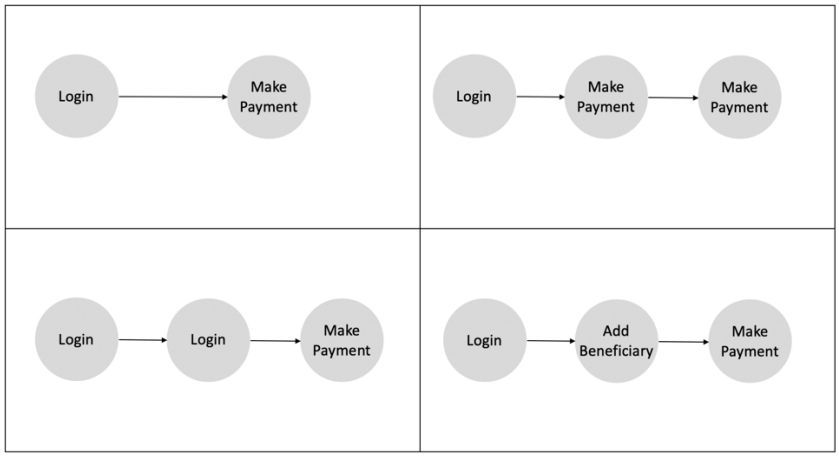


Figure 3.24. Common Sequences in Dataset

In order to test the hypothesis, the following, sequence is used as user state as shown in Table 3.19 below:

State	Description
Login	User is logged into the banking portal
Add Beneficiary	User added a new beneficiary
Make Payment	User made a payment to the newly added beneficiary

Table 3.19 Sequence To-Be Used for Different Experiments

The rationale behind this is that it covers more user journeys than the rest of the above sequences. Further data analysis around the above sequence resulted in a reduced dataset of 280 transactions for non-fraud dataset and 361 transactions for fraud dataset. This approach will substantially simplify the analyses of the data distribution and would easily allow us to impact assess the test instances. As a next step, the time taken between the login event and add payeeevent were analysed for both of the datasets. Results are shown in Figure 3.25 below:

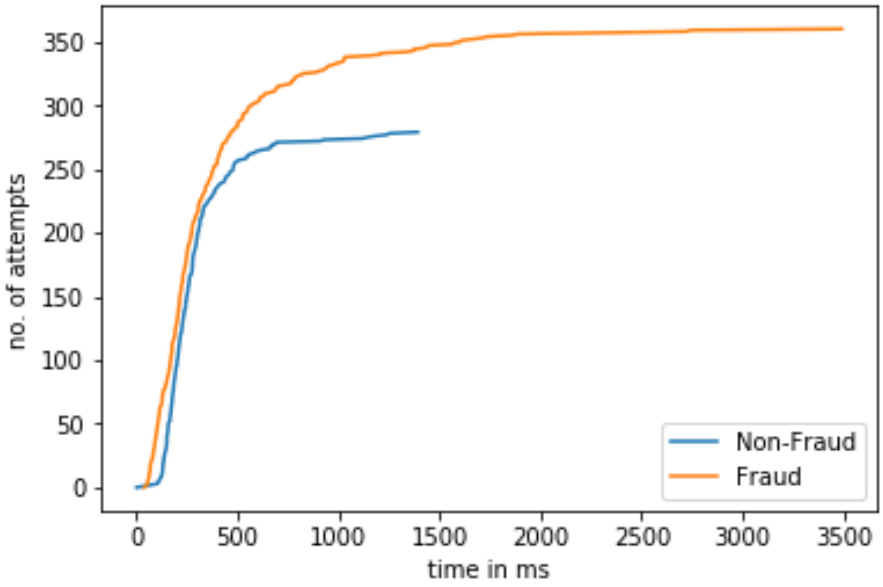


Figure 3.25. Average Transition Time Between Fraud and Non-Fraud Sequence

The above graph indicates that there is an intrinsic difference between fraud and non-fraud dataset. Examining the set of sequences for the both fraud and non-fraud, it is clear that the transition time between fraud is longer than non-fraud. This can be used as a core feature on which a set of profiles can be created. The statistical profile contains a list of sequences per each user session and a distribution function with fitted parameters for the transition time. Based on the above information, the following attributes are used to derive a profile as depicted in Table 3.20 below:

Attribute	Description
UserID	Unique user id that is used for profiling
SessionID	Unique session id for which
Event 1	Name of first event
Timestamp 1	Time it took from the first event to the second event
Event 2	Name of the second event
Timestamp 2	Time it took from the second event to the third event
Event 3	Name of the third event, an end state
Overall Time	Overall time it took
Fraud_flag	Fraud flag 0 = no fraud and 1 = fraud

Table 3.20 Attributes Used for Modelling

3.4.5. Data Generation Approach

The activity flow diagram depicted in Figure 3.26, provides an approach to generating data based on the data generation methodology and set of sequences described above.

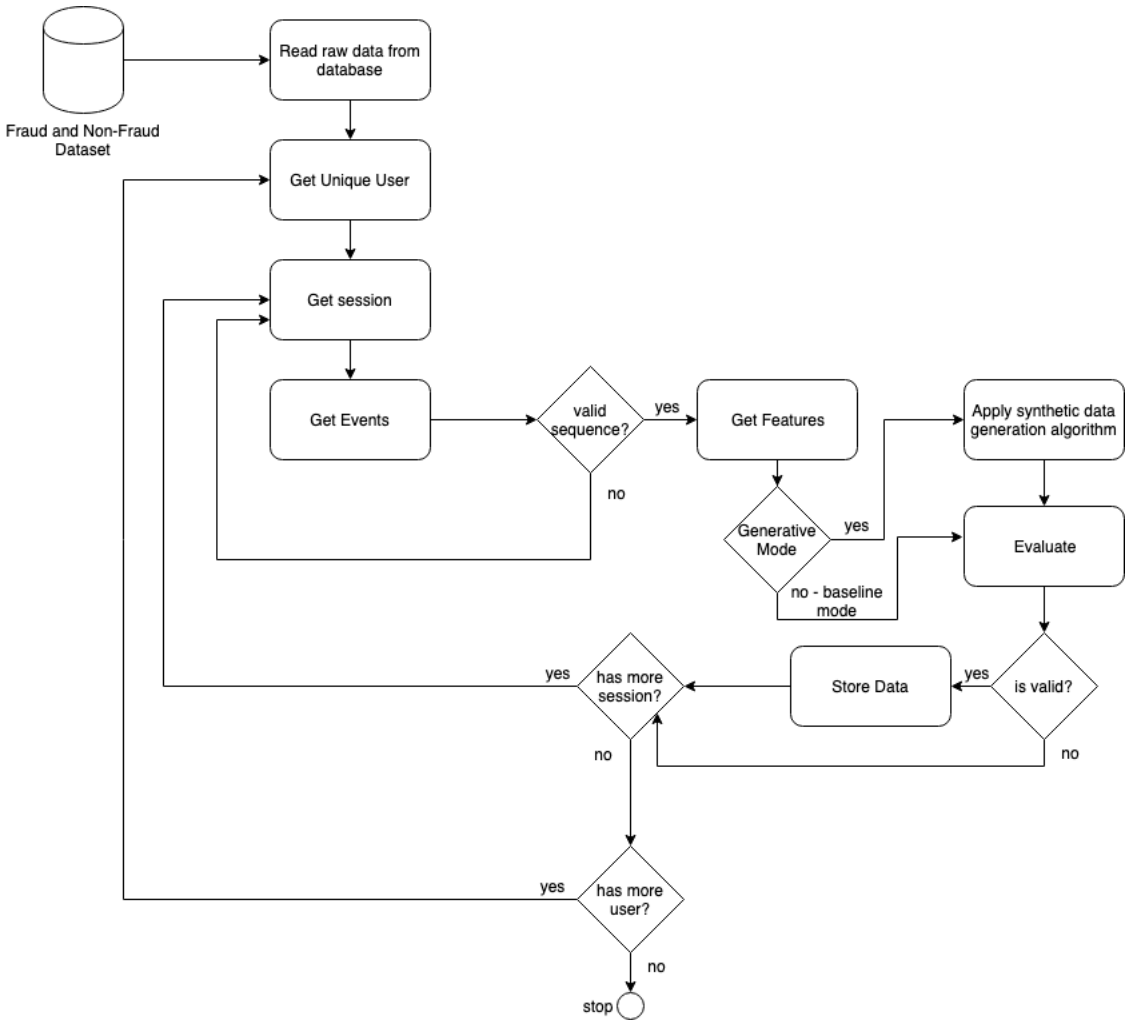


Figure 3.26. Activity Diagram for Data Generation

The activity steps for the data generation process is given as:

- Read all the raw data from database that contains both fraud and non-fraud dataset.
- Get a list of unique users and for each user
    - Get a list of sessions and for each session
        - Get a list of event sequences
        - Check to see if the event sequences match the pattern defined in Table 3.19 (i.e. Login, Add Payeeand Make Payment)
    - For list of sessions, that have matched the above pattern, calculate a set of features as defined in Table 3.20
    - Check if the generative mode is enabled. In this mode, the system will use a list of sessions and features calculated above to generate new data.
    - In baseline mode, it will invoke evaluate process

- Evaluate process will apply a classifier to discriminate between a fraud and non-fraud events

3.4.5.1. Baseline Mode

In order to evaluate the effectiveness of generated data, a baseline is required that allows a classifier to evaluate the original dataset. For a purpose of evaluating the original data is by the K-Means algorithm with cluster setting of 2 (fraud and non-fraud). A confusion matrix is created in Figure 3.27 below, False Positive Rate and True Positive Rate are the two parameters used to evaluate the original dataset.

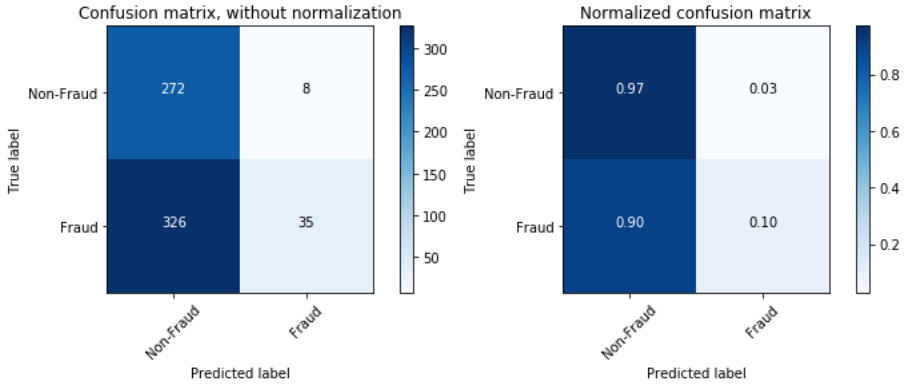


Figure 3.27. Confusion Matrix on Original Dataset

As seen above the k-means algorithm gets most of the non-fraud label right (97%) but gets most of the fraud label wrong (90%). This means the model operates on a 90% false positive rate. This is not surprising as per the graph above that a small number of sequences the fraud transition time is not differentiable. Please note the intent of this approach is not to analyse the effective of the k-means algorithm but base the results and compared against the generated dataset. The intension is to maintain this base while increasing the size of the dataset.

3.4.5.2. Generative Mode

Research highlighted that recently Generative Adversarial Network (GAN) (133) has become quite popular in expanding real dataset by making use of the real dataset to generate synthetic dataset. Lots of experiments have been carried in the field of computer vision where using images to generate additional images. This has achieved a good level of success in an image generation (131). Maayan F, et al. (132) applied GAN on medical image segmentation

of 182 liver lesions. Their approach is to create synthetic images that provides different permutation of liver lesion and using this data to augment the original dataset. This approach has significantly increased their model accuracy for Liver Lesion classification.

This section investigates the use of GAN to model the underlying statistical distribution of transactional data described above in section 4 to generate synthetic data that can be used to augment the real training data. The use of GAN has several advantages such as reducing the variance to noise ratio (133), while preserving the discriminative features as well as removing the need to hand-craft features. GAN has an ability to learn from a parallel set of features and infer this model to generate and augment directly from the available data.

Figure 3.28 below provides an architecture view of the system used to build the generative mode. The core component is the function $f(x)$ which makes use of the GANs.

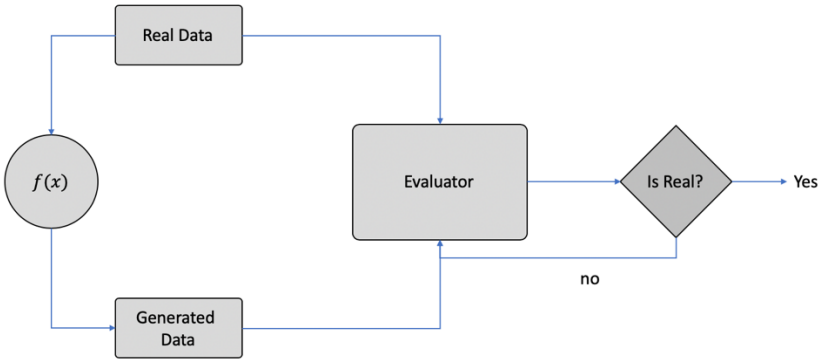


Figure 3.28. Generative Mode Architecture

Function $f(x)$ is a generative model that is used to create a sample from a real distribution by applying Gaussian fit to the data. That is, for each class $y\{fraud \mid not\ fraud\}$, $p(y|x)$ is modelled rather than modelling directly $p(x|y)$. Since the data is of continuous nature, continuous distribution is derived from the maximum entropy for a specified mean and variance. This approach is used to provide continuous probability distribution of the random variables (for example, time taken to move from one state to another) which is shown in Figure 3.29 below.

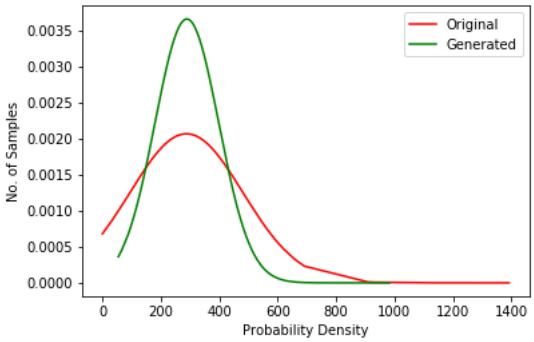


Figure 3.29. Probability Density Function on Original Data

For each user, the data modelling $p(x|y = user)$ is based on a single Gaussian fit. The pdf in the above diagram shows a data generation of a continuous random variable (time taken between login page and add payeepage) that has many local maxima and the peak of a data generation and the mean distribution has multiple peaks. This make sense in our case as not every user spends the same amount of time on each page, but it can be imagined that there are only finite amounts of time is spent on each page by different user. This is reflected in the data that has been generated with much higher peak.

The statistical relationship of original data is then learnt by the GAN network to generate synthetic data. An example of such synthetic data generation is shown as a graph in Figure 3.30 below. Based upon an approach defined in Figure 3.26, a random user is chosen from the list, a set of events (as described in Table 3.19) and the network then learns the relevant features:

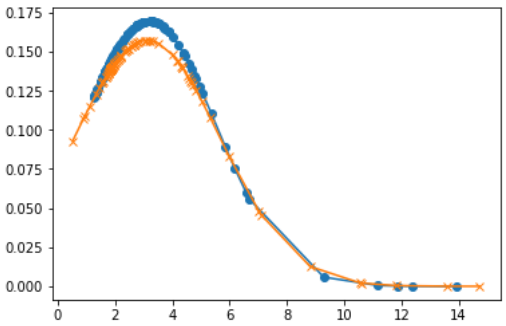


Figure 3.30. Applying GAN to Learn from Original Data

As seen from the above diagram the GAN as learnt the behaviour of an individual user and has able to replicate the user behaviour. The table below provides an insight on how close the GAN network came to generating data that matches the original data:

Event	Mean Original Time (seconds)	Mean Generated Time (seconds)
Login to Add Beneficiary	3.185	3.169
Add Payeeto Make Payment	2.352	2.436

Table 3.21. Attributes Used for Modelling

As seen in the table 3.21 above, the mean generated transition time for a given sequence of events, is closely similar to the mean original transition time. In order to compare with the baseline, K-Means is applied with the same settings used in the baseline mode. The results are shown in the diagram below:

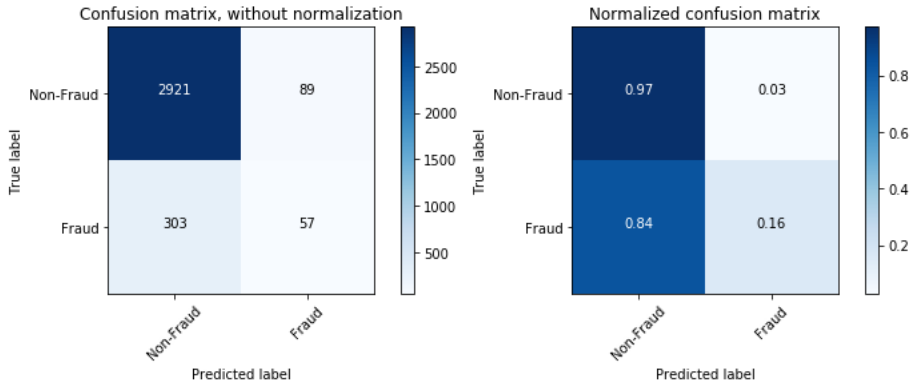


Figure 3.31. Confusion Matrix on Generated Dataset

As seen above in Figure 3.31, based on just two data points (transition time from Login to Add Payeeand transition time from Add Payeeto Make Payment), the K-means algorithm has reduced on the False Positive rate from baseline of 90% to 84% while the non-fraud detection rate remains the same. This proves our hypothesis that synthetic data can be used for testing and evaluating fraud detection system and GAN network can successfully generate a better dataset that can be used to augment original dataset for training and evaluating. Next step is to add additional features and repeat the process.

3.5. Conclusion

This chapter discusses the core elements that will provide the major contribution to this thesis, namely, feature engineering, neural network model, model evaluation. Furthermore, this chapter detailed fraud detection framework and different components can be used in different environments. Feature engineering process proposed in this chapter, is focused on taking transactional data, personal data, account data and behavioural data transforming the data into features and latent features that can be consumed by sequence learners and ensembling classifier. The importance of session length and time taken to complete an event is also discussed in detailed. In order to address the class imbalanced and lack of data problems, GAN is proposed to generate additional data without losing the statistical relationship. Neural network presented in this section provided additional context and insight into sequence learners such as LSTM network is suitable for scoring, self-learning and classification of the real banking events. The next chapter presents the implementation methodology that will allow us to run a series of experiments on the data sets using LSTM, SVM and Markov models, during which the resulting performances of each algorithm will be computed using the chosen performance measure. These results can then be analytically compared to see how the three algorithms compare to each other when applied to a non-trivial real-world problem. This will either prove or disprove our original hypothesis.

3.6. Key Summary

This section describes the high-level summary of the content of this chapter.

- Proposed a novel cross channel fraud detection framework for efficiently and quickly detect fraudulent events using event-based sequence learners.
- Details a Logical data model for cross-channel fraud detection.
- Technique to perform feature engineering and data normalisation is discussed. Focus is specifically on creating features that are based on sequences and technique to normalise the data so that it can easily fit to a sequence-based model.
- Provides insight into the use of recurrent neural network model as sequence learner and how it can be adapted to fit the customer journeys for online banking.
- Model development and model execution environments are presented for separation of concerns.

- Discusses related research into the data generation technique and proposed a novel data generation technique using GAN.
- Framework for model performance and evaluation is presented for testing such as TRP, FRP, TNR and FNR.

4.

CCFDS – In Practice (Sequence learning models for fraud detection using real life banking transactions)

4.1. Model development

In Chapter 3, a novel payment fraud detection framework consisting of sequential data inputs across multiple channels, Feature Engineering and Neural Network model components to detect anomalous events is conceptualized. In this chapter framework is detail to develop an integrated Payment Fraud Detection System (PFDS) model with a real-life scenario dataset from a large European bank. The model can dynamically evolve to maintain the efficiency with minimum input based on data sampling, feature engineering and multiple intelligent algorithms. The overview of the model is shown in Figure 4.1.

The model involves five main modules, i.e., Feature Engineering and Data Sampling, Fraud Detection Models, Deep Neural Network Ensembler, Model Evaluation Module and Rule System. The Feature Engineering and Data Sampling building block examines and analyses the events collected from remote banking channel and create latent variables to form a feature matrix that can be processed easily by Fraud Detection Models. It also creates training and testing sample datasets based on original datasets and synthetic datasets. The Fraud Detection Models perform scoring and classification by sequence learning the temporal patterns of the entire customer history. This involves two steps; first, multiple models are developed based on a featured data pool generated by Feature Engineering and Data Sampling module and a model pool, where each model produces a score with a range of [0, 1]. Then, the results are ensembled by a deep neural network model to produce an optimized classification. This research implements Events Sequence Model using LSTM, Markov and SVM models. Subsequently the models and their results are evaluated, and rules are stored into the Rule System.

The architecture is planned to be developed in two stages. The Deep Neural Network Ensembler module is not included in the first stage development. The IP Address model, UA model, RFM model and Benford model have been developed at the first stage but not linked

to the Model Evaluation module, so their details are not included in this section. However, their implementation can be found in Appendix – A. At the second stage, the research develops an integrated Payment Fraud Detection System (PFDS) model and implements Long Short-Term Memory (LSTM) family of Recurrent Neural Network model, SVM model and Markov model as an event sequence learner for detecting fraud in remote banking. Subsequently, the research adopts a comprehensive approach to evaluate and compare their performances based on well-defined indicators and datasets.

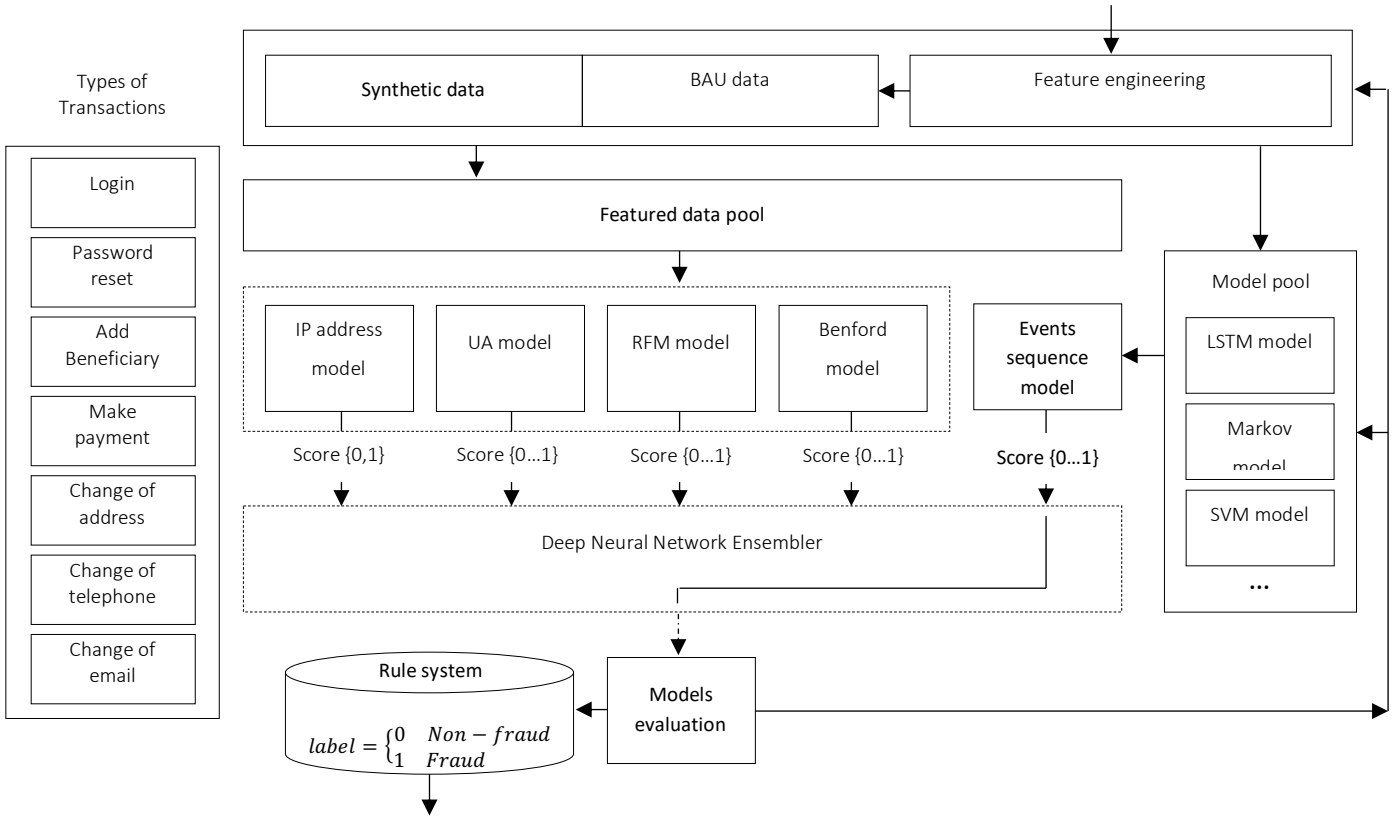


Figure 4.1. Confusion Matrix on Generated Dataset

4.2. Data and pre-processing

4.2.1. Data source and description

The datasets are obtained for event sequence learning from a large European bank. They are anonymised before taken offline for analysis. The datasets are the online payment transactions which include four web pages, i.e. Login, Second Factor Login, Adding Beneficiary, and Making Payment pages and corresponding browsing activities. For the fraud detection analysis two types of datasets can be further classified, i.e. fraudulent datasets and legitimate/non-fraudulent datasets. In total, the fraudulent datasets contain 22,880 records and the non-fraudulent data has 100,000 records. This highly unbalanced dataset fraud labels for 0.172% of all transactions. A snap of the data samples with a two-dimensional structure are shown in Figure 4.2 below:

ID	PARTYID	CHANNEL_IDENTIFIER	SUBCHANNEL_NAME	EVENT	FINANCIAL_INSTITUTE	INTERNET_SESSIONID	AUTHENTICATION_CODE	TRANSACTION_ID
0	2909988221	INTERNET_BANKING	MOBILE	NotifySessionRequest	XYZ	ab1f4f8608a111e6b61a22d49b75d30c	900	b2ded26408a111e6b61a22d49b75d30c
0	2909988221	INTERNET_BANKING	MOBILE	ScorePaymentRequest	XYZ	ab1f4f8608a111e6b61a22d49b75d30c	900	bc6267c408a111e6b61a22d49b75d30c
0	2909988221	INTERNET_BANKING	MOBILE	NotifySessionRequest	XYZ	145e5b8a096b11e6aa8aaae8038c220c	900	1c39699e096b11e6aa8aaae8038c220c
0	2909988221	INTERNET_BANKING	MOBILE	ScorePaymentRequest	XYZ	145e5b8a096b11e6aa8aaae8038c220c	900	27b3613a096b11e6aa8aaae8038c220c
0	2909988221	INTERNET_BANKING	MOBILE	NotifyPaymentArrangem…	XYZ	145e5b8a096b11e6aa8aaae8038c220c	500	6468fec8096b11e6aa8aaae8038c220c
0	2909988221	INTERNET_BANKING	MOBILE	NotifySessionRequest	XYZ	cd4a4ea6096b11e68e67aae8018a030d	900	d07b683a096b11e68e67aae8018a030d
0	2909988221	INTERNET_BANKING	MOBILE	ScorePaymentRequest	XYZ	cd4a4ea6096b11e68e67aae8018a030d	900	d74c50b6096b11e68e67aae8018a030d
0	2909988221	INTERNET_BANKING	MOBILE	NotifySessionRequest	XYZ	cd4de746097011e6b61a22d49b75d30c	900	da5c0c4c097011e6b61a22d49b75d30c

EVENT_TYPE	USERAGENT	IPADDRESS	AVAILABLE_BALANCE	ACCOUNT_NUMBER	SORT_CODE	AUTO_RESPONSE	LATENCY	TRANSACTION_AMOUNT	TRANSACTION_SPEED	TRANSACTION_REFERENCE
IDVIVR			0			false	59	0		
CustomerLogin	Mozilla/5.0 (iPhon…	188.29.165.44	0			false	103	0		
InternalTransfer			20	21995767	117361	false	812	10	REAL_TIME	
CustomerLogin	Mozilla/5.0 (iPhon…	92.40.248.125	0			false	82	0		
InternalTransfer			10	21995767	117361	false	893	10	REAL_TIME	
CustomerLogin	Mozilla/5.0 (iPhon…	2.96.237.106	0			false	105	0		
CustomerLogin	Mozilla/5.0 (iPhon…	2.96.237.106	0			false	87	0		
InternalTransfer			6.5	11807566	110772	false	576	6.5	REAL_TIME	

TRANSACTION_SPEED	TRANSACTION_REFERENCE	BENEFICARY_ACCOUNT_NUMBER	BENEFICIARY_SORT_CODE	TRANSACTION_TIME	FRAUD_INDICATOR	FRAUD_FLAGS
				2016-04-19 11:36:29	0	
				2016-04-19 15:59:01	0	
REAL_TIME		00449531	110760	2016-04-19 15:59:45	1	Tfr
				2016-04-19 18:32:07	0	
REAL_TIME		00449531	110760	2016-04-19 18:32:19	1	Tfr
				2016-04-22 16:43:33	0	
				2016-04-22 16:48:46	0	
REAL_TIME		00449531	110760	2016-04-22 16:49:02	1	Tfr

Figure 4.2. A snapshot of the data samples

The full description of the data attributes is listed here:

Features	Description
EVENT_TYPE	Type of event e.g. Customer Login, Make Payment etc
TRANSACTION_ID	Transaction ID
CHANNEL_IDENTIFIER	A way that customers can interact with a bank. This can be via telephone, internet banking, branch, mobile.
FINANCIAL_INSTITUTE	Financial Institute name
SUB_CHANNEL	Sub-channel name
PARTYID	Customer ID
PARTY_TYPE	Customer party type code
PARTY_RELATIONSHIP_TYPE	Customer's relationship type
PARTY_BIRTH_DATE	Customer's birthday date
PARTY_EMAIL_ADDRESS	Customer's email address
PARTY_MARITAL_STATUS	Customer's marital status
PARTY_FIRST_NATIONALITY	Customer's first nationality
PARTY_OCCUPATION	Customer's occupation
PARTY_REGISTRATION_DATE	
PARTY_TELESERVICEREGISTERDT	Registration date of telephone service
PARTYTL	Customer's Title
PARTY_FIRSTNAME	Customer's first name
PARTY_MIDDLENM	Customer's Middle name initial
PARTY_SURNAME	Customer's Surname
PARTY_PMRADDRESSLINE1	Customer's account primary address line 1
PARTY_PMRADDRESSLINE2TX	Customer's account primary address line 2
PARTY_PMRADDRESSTOWNNM	Town name of the account primary address
PARTY_PMRPOSTCODE	Account Postcode
PARTY_PMRCOUNTRYCD	Account Country code
PARTY_ADDRESSLINE1	Customer's address line 1
PARTY_ADDRESSLINE2	Customer's address line 2
PARTY_ADDRESSTOWNNM	Town name of the address

PARTY_POSTCD	Postcode
PARTY_COUNTRYCD	Country Code
PARTY_MAILADDRESSLINE1	Customer's mail address line 1
PARTY_MAILADDRESSLINE2	Customer's mail address line 2
PARTY_MAILADDRESSTOWNNM	Customer's town name of mail address
PARTY_MAILPOSTCODE	Customer's mail address postcode
PARTY_MAILCOUNTRY	Customer's mail address country code
PARTY_HOMEPHONENO	Customer's home phone number
PARTY_MOBILEPHONENO	Customer's mobile phone number
PARTY_OFFICEPHONENO	Customer's office phone number
CORRESPONDENCE_NAME	Account correspondence name
ACCOUNT_SORTCODE	Account's sort code
BANK_ACCOUNTNO	Account's bank account number
ACCOUNT_TYPE	Account type
ACCOUNT_AVAILABL_BALANCE	Account's available balance
ACCOUNT_CURRENCY	Currency code of the account's available balance
ACCOUNT_OPENDT	Open date of the account
RISK_INDICATOR	Risk indicator code
EXISTCAHELDIN	
COMMERCIAL_BANKINGCUSTIN	Commercial banking
CREDIT_CARDHELD	Credit card held in
INSURANCE_PRODUCTHELDIN	Insurance product held in
PD_INVESTMENTPRODUCTHELDIN	Investment product held in
LOAN_PRODUCTHELDIN	Loan product held in
MORTGAGE	Mortgage in
BUSBANK_PRODUCTHELDIN	Business bank product held in
SAVINGS_HELDCD	Savings held code
RISK_SCORE	Risk score value
BENEFICIARY_NAME	Payeename of transaction
BENEFICIARY_SORTCODE	Payeesort code

BENEFICIARY_ACCOUNTNO	Payeeaccount number
BEN_ADDRLINE1	Beneficiary's address line 1
BEN_ADDRLINE2	Beneficiary's address line 2
BEN_ADDRTOWN	Town name of beneficiary's address
BEN_POSTCODE	Postcode of payeeaddress
BEN_COUNTRY	Country code of payeeaddress
TRANSACTION_AMOUNT	Transaction amount
TRANSACTION_CURRENCY	Currency code of transaction amount
TRANSACTION_PMTDATE	Payment date of transaction
TRANSACTION_SPEED	Transaction speed (e.g. Real time, several days, faster, as soon as possible)
PAYMENT_INDICATOR	Faster or Standard Payment indicator: Faster (1) or Standard (2)
STANDING_ORDER_FREQ	Standing order payment frequency code (e.g. weekly, monthly)
FIRST_PERIODICPAYMENT_AMNT	Amount of first periodic payment
FIRST_PERIODICPAYMENT_AMNTCUR	Currency of first periodic payment amount
STANDINGORD_AMOUNT	Standing order amount
STANDINGORD_CURR	Currency of standing order's amount
LAST_AMOUNT	Last amount
LASTAMOUNT_CURR	Currency code of last amount
FINALPAYMENT_DT	Final payment date
TNSX_REFERENCE	Transaction reference
LOCAL_DIRECTOR_OFFICENM	Local director's office name
LOAN_TYPE	Loan type code
REJECT_CODE	Transaction reject code
AUTHENTICATION_CODE	Authentication code
TRANSACTION_TIMEZONE	Time zone of transaction
INTERNET_SESSIONID	Internet session ID
INTERNET_SESSION_STARTTIME	Start timestamp of internet session
CLIENT_SCREEN_RESOLUTION	Client screen resolution
PROXY_AUTHO	Proxy authorization
TRANSACTION_USER_AGENT	User agent

BROWSER_LANGUAGE	Browser language
USERID	User ID
IP_ADDRESS	IP address
DEVICEID	Device ID
DEVICE_RISKSCORE	Device risk score code
SESSIONDR	
TEL_SESSIONID	Telephone session ID
CALL_TIME	Call timestamps
CONTACT_PHONENO	Contact phone number
PARTY_INICALLIN	
TRANSACTION_TIME	Transactions timestamps
EVENT	Event of transaction
AUTO_RESPONSE	Auto response
PARTY_SMSPHONENO	Customer's SMS phone number
LATENCY	Latency
PARTY_EIAPHONENO	Phone number for Enhanced Internet Authentication (EIA)
PARTY_EMAIL2	Customer's second email address
PARTY_EMAIL3	Customer's third Email address
LOGIN_TYPE	Login Type
AUTH_TYPE1	Authentication Type 1
AUTH_DETAILS1	Authentication details
AUTH_DETAILS2	Authentication details
AUTH_DETAILS3	Authentication details
AUTH_TYPE2	Authentication Type
AUTH_TYPE3	Authentication Type
BEN_BANKID	Payeebank ID
BENF_BANKNAME	Payeebank name
BENY_ROUTINGTYPE	Payeerouting type
INTERNATIONAL_TRANSACTIONAM	International transaction amount
BENBANK_COUNTRYNM	Payeebank country name

BENBANK_ADDRLINE1TX	Payeebank's address line 1
BENFBANK_ADDRLINE2TX	Payeebank's address line 2
BENFBANK_ADDRTOWNNM	Town name of payeebank's address
BENFBANK_POSTCD	Postcode of payeebank's address
BENFBANK_COUNTRYCD	Country code of payeebank's address
TRANSACTION_CHARGES	Amount charged for transaction

Table 4.1. A full description of the data sample features extracted from real banking database

The datasets exhibit a set of the events that customers have from the time of logging into their online banking to the completion of the transactions. The fraudulent events were collected between the dates 7/09/2015 and 7/07/2016, whereas the given legitimate events were collected between the 00:00am and 01:43am on 15/10/2012.

In total there are 124 features for each row. To reduce the dimensionality and bias of the modelling, some of the features with dominant missing values are either removed or filled with well-tuned values (e.g. the last known value) during the modelling. Overall there are 47 features that do not possess any significant values and few other features with very limited values or repetitive values across all the transactions. These features are predominantly personal information about the customers such as email address, birth date, phone number, country code and the bank's detail.

The modelling favours the features such as Transaction amount, Event type, Latency, IP address, User agent, Transaction time, Payee's account number and sort code, Beneficiary's account number and sort code. Customer ID attribute is also included in the modelling process as it helps to aggregate and analyse the datasets with identified features. In this case, the personal information, mainly IDs, and the account information are anonymised for confidentiality. The data selection and pre-processing are further illustrated in more specific detail in the data pre-processing subsection and model implementation section with respect to specific models.

4.2.2. Exploratory analysis

The explanatory analysis in this section is intended to understand and determine the probability distributions of the datasets and the key data features used by the model, and subsequently identify possible outliers.

The features used for this analysis include Transaction Amount, User Agent, IP Address and the transaction datetime. Only the records tagged with 'PaymentRequest' are analysed while zero payment request transactions are ignored. In total, 18,169 legitimate transactions and 4,322 fraudulent datasets are selected. In the fraudulent datasets, however, not all the transactions are fraudulent, out of which 916 transactions are labelled as being fraudulent while the rest 3406 are considered as being legitimate. The analysis here explores all these three types of datasets.

The distributions of fraud transactions, legitimate transactions and legitimate transactions in fraudulent datasets are overlaid in Figure 4.3 for comparison. The transactions at lower amounts are more frequent for both legal and fraudulent events, as shown from their distributions. To more clearly show the frequency of the right end of the x-axis, i.e., the high amount of payment transactions, two more distribution plots without overlay are generated in Figure 4.4 and Figure 4.5, where Figure 4.5 shows the logarithm distribution by amounts.

From both figures, the payment amounts in the legal transactions are largely below £2,000 and the legal payment in the fraud data are largely than £1,000, which demonstrate that the personal customers are cautious of their online banking activities. The payment amounts of most of the fraud transactions are spread between one penny and £3,000. However, some fraud transactions occurred with very high payment up to £25,000. The distributions of legal transactions are smoother than the fraud transactions which rightly demonstrate it happens more sparely. Figure 4.4 shows that there is 0.5% more probability that a transaction with small amount of payment (i.e., < £1,000) is fraudulent than legitimate. A table below (i.e. Table 4.2) is also included to show the occurrences of fraud transactions, legitimate transactions and legitimate transactions in fraudulent datasets with payment amount ranges.

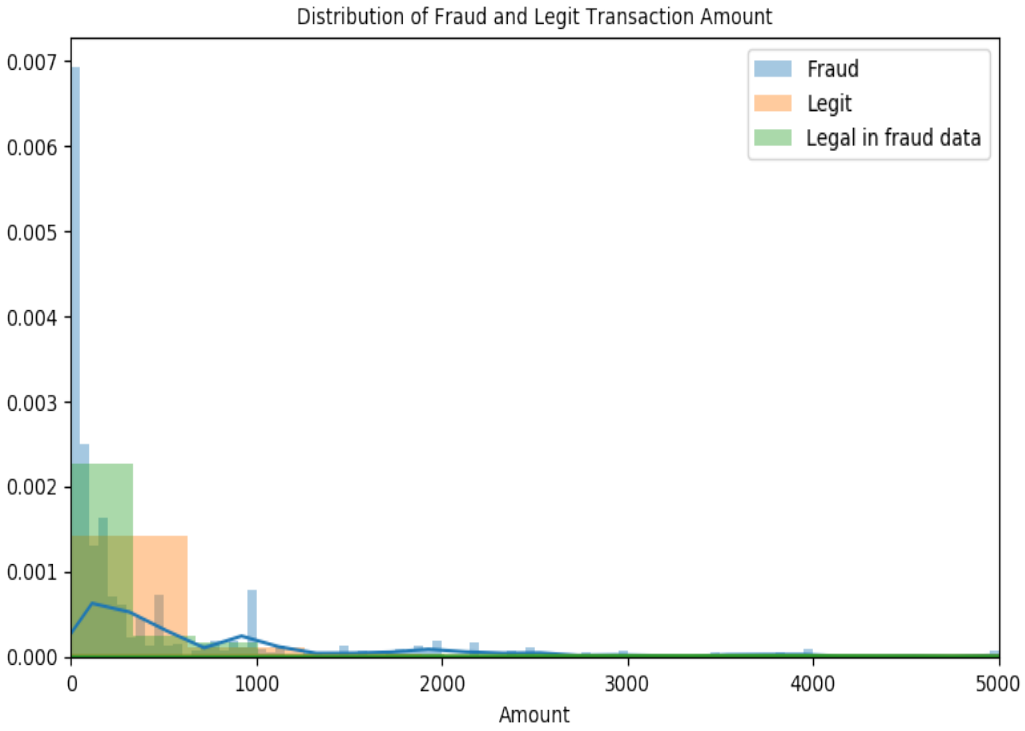


Figure 4.1. Distributions of the fraudulent and Legit transactions with payment amounts with overlay

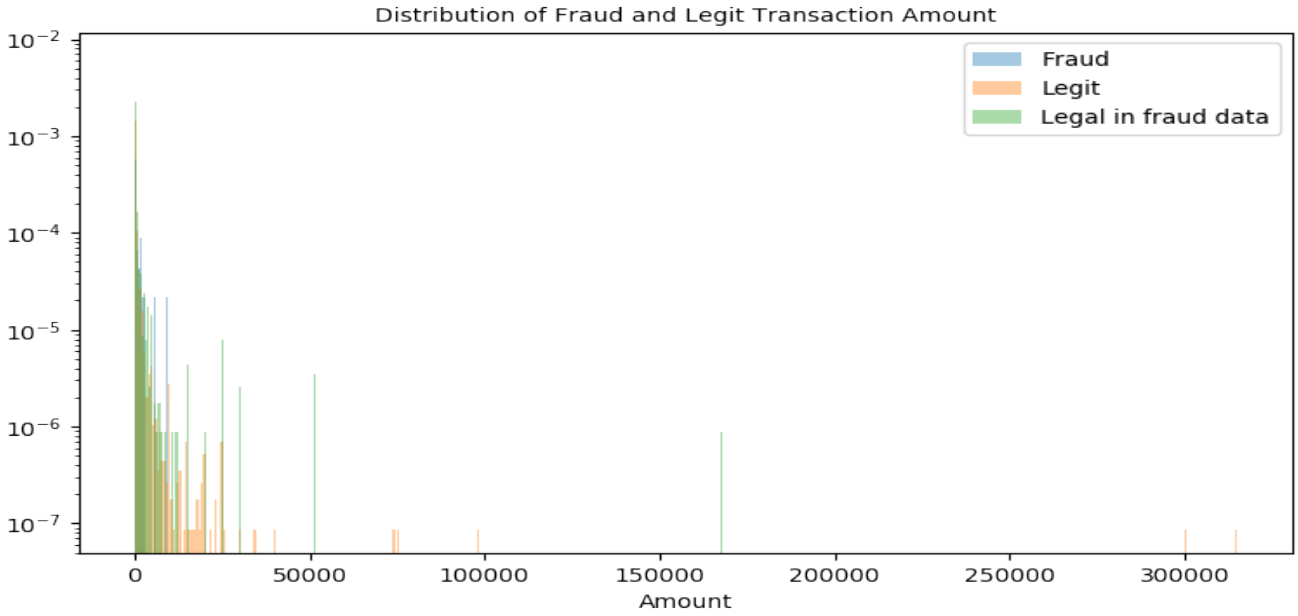


Figure 4.2. Distributions of the fraudulent and Legit transactions with payment amounts without overlay

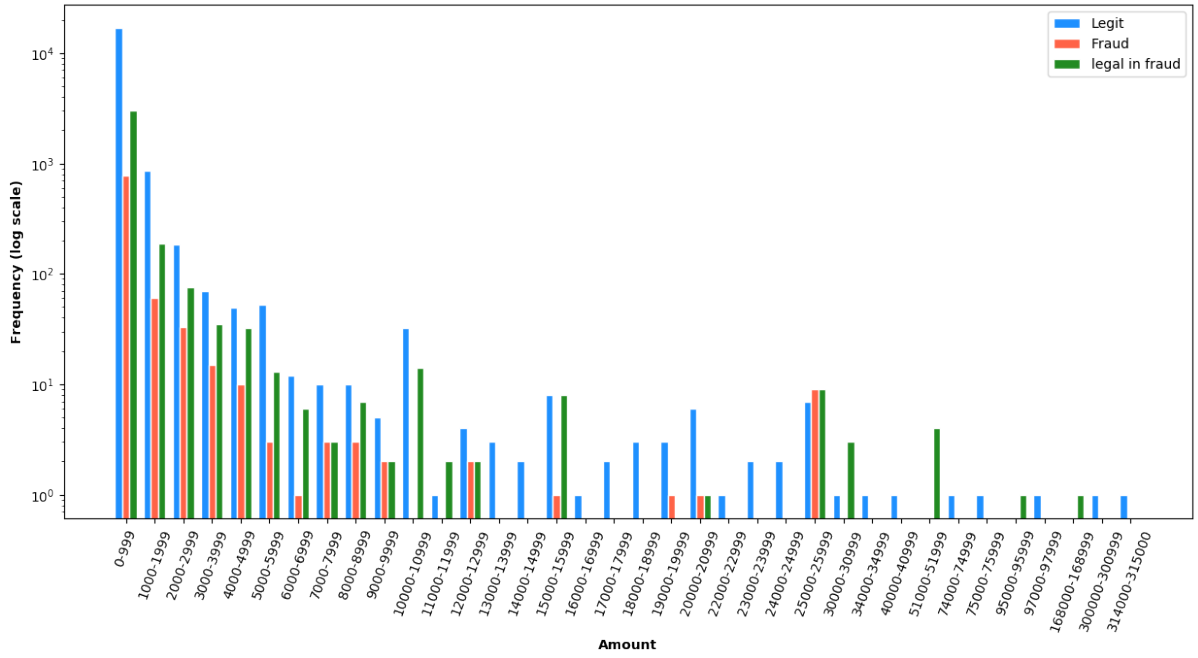


Figure 4.3. Log Distributions of the fraudulent and Legit transactions with payment amounts with overlay

Amount range	Legit Frequency	Fraud Frequency	Legal in fraud Frequency
0-999	16826	771	3001
1000-1999	866	61	187
2000-2999	183	33	75
3000-3999	70	15	35
4000-4999	49	10	32
5000-5999	53	3	13
6000-6999	12	1	6
7000-7999	10	3	3
8000-8999	10	3	7
9000-9999	5	2	2
10000-10999	32		14
11000-11999	1		2
12000-12999	4	2	2
13000-13999	3		
14000-14999	2		

15000-15999	8	1	8
16000-16999	1		
17000-17999	2		
18000-18999	3		
19000-19999	3	1	
20000-20999	6	1	1
22000-22999	1		
23000-23999	2		
24000-24999	2		
25000-25999	7	9	9
30000-30999	1		3
34000-34999	1		
40000-40999	1		
51000-51999			4
74000-74999	1		
75000-75999	1		
95000-95999			1
97000-97999	1		
168000-168999			1
300000-300999	1		
314000-315000	1		

Table 4.2. Occurrences of the fraudulent and legitimate transactions with payment amount ranges

Here, the specific time frames when the data were collected is explored. The legitimate transactions were collected between 12:00am and 1:45am on 12 October 2012, over a time of 1 hour and 45 minutes, while the fraudulent events were collected over about a year. Figure 4.4 and Figure 4.5 overlay the two pieces of the datasets with transactions spanning a day, where Figure 4.4 is a natural distribution and Figure 4.5 is a log logarithmic distribution of the transaction payment amounts. In the pictures, the red data points show the legitimate transactions. The green and blue points represent the legitimate and fraudulent transactions respectively. The figures show that both the legitimate and fraudulent transactions from fraudulent events occurs more often since the midday. The amounts of the legitimate

transactions from legitimate events in red are more evenly distributed than the amounts of the fraudulent transactions.

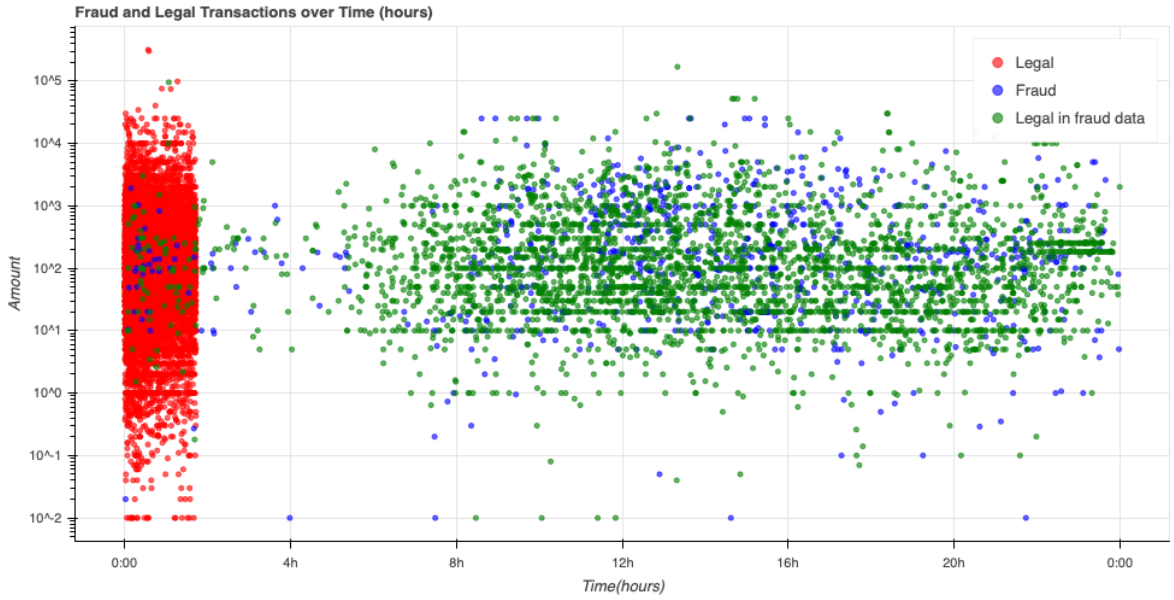


Figure 4.4. Fraud and Legal Transactions Over Time in Hours by Logarithm in Y-Axis

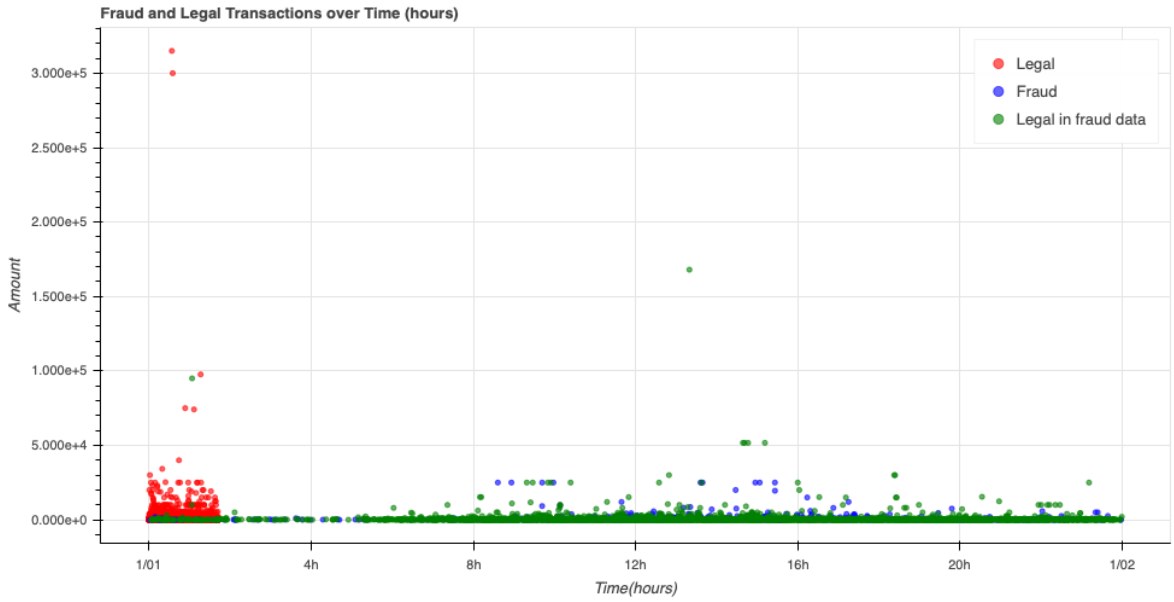


Figure 4.5. Fraud and Legal Transactions Over Time in Hours

If one plots the fraudulent data which only happens over a day, as shown in the Figure 4.7, logarithmic version is depicted in Figure 4.8, one may find that between 2:00am and 8:00am

there are the least fraud transactions, whereas from midday up to 4:00pm and after 8:00 pm the occurrences of fraudulent events are much higher.

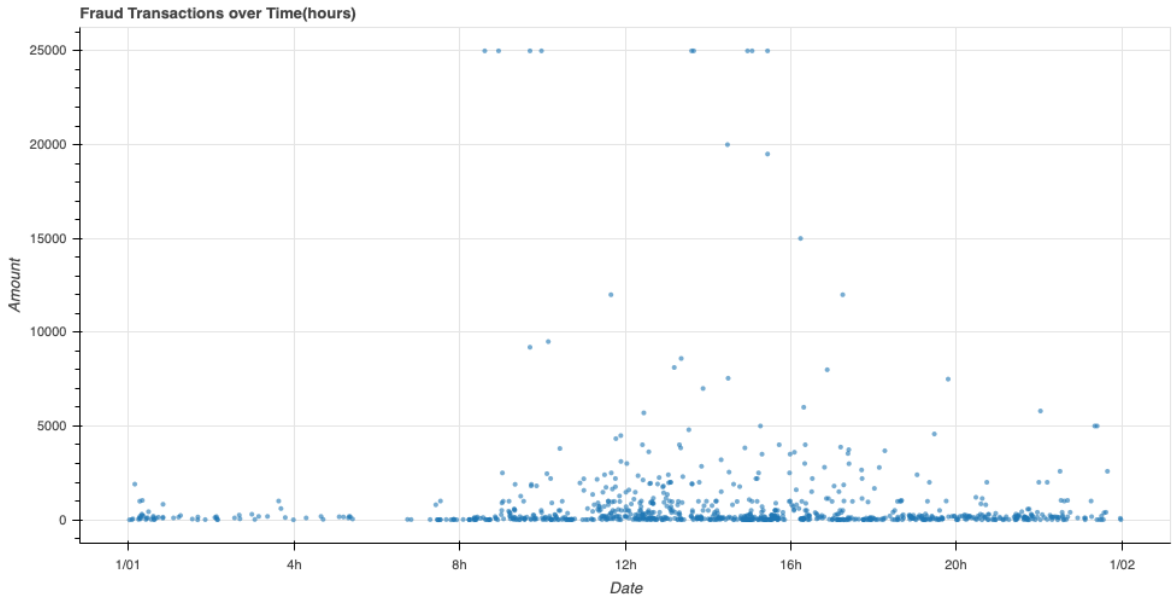


Figure 4.6. Fraud Transactions Over Hours of the Day

If one investigates deeper into the time when a fraudulent event occurs, most fraudulent events happen at the beginning and end of each month - from the 1st up to the 7th and then from 30th to 31st, as shown in Figure 4.8. The left figure shows an inverted bell shape. In the mid-month, the detected fraudulent events drop and rise again at the end of month. The figure also shows that the legitimate transactions in the fraudulent data follow the same pattern as the fraudulent events. The right figure shows that May and June are most active for fraudsters. There are more than 500 fraudulent transactions detected during June on the top list of the months, May is also highly active for the fraudsters with approximately 300 fraud transactions. On the other hand, legitimate transactions in the fraud datasets occur more often in April, March and May, exhibiting a non-correlation between this two.

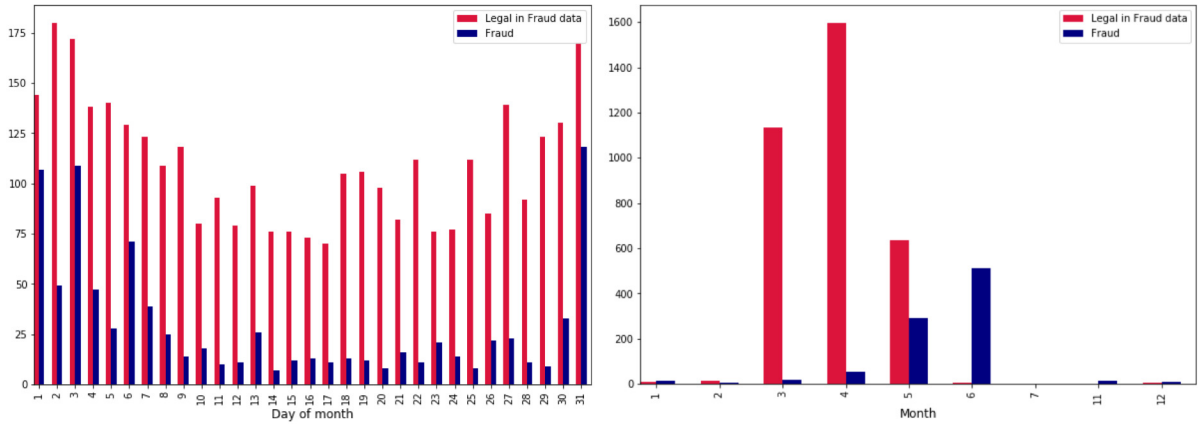


Figure 4.7. Number of Fraudulent and Legal Events in Days (left plot) and Months (right plot)

By clustering IP addresses based on the leading three bytes, one can identify the locations of fraudulent events as well as the locations of the legitimate events. As shown in Table 4.3., the large share of fraudulent events (i.e., 54) occurred in Canada followed by UK regions including Warrington, Exeter, Uxbridge Thetford and London from our data samples. One of the reasons is that the data were collected from a bank located in the UK. In the meantime, the IP addresses of the legitimate transactions from the fraudulent datasets show that they are mainly originated from the locations including Brighton, London, Wakefield, Warrington and Thetford.

Fraud Location	IP address (3 bytes)	Count	Legal Location	IP address	Count
Canada (Windsor)	135.207.48	54	**Brighton**	92.40.249	52
Warrington	211.205.252	46	**London**	81.96.8	51
Exeter	211.205.194	28	**Wakefield**	92.40.248	50
Uxbridge	84.23.59	18	**Warrington**	86.9.118	50
Thetford (Norfolk)	81.255.233	16	**Thetford (Norfolk)**	85.255.233	48
London	82.81.143	14	**Ashford**	94.197.120	45
Utley	183.69.144	12	**Manchester**	164.39.65	41
Bridport	88.132.212	10	**Derby**	86.156.151	41

Crewe	89.229.51	10	**Exeter**	213.205.194	41
London	144.92.120	9	**Warrington**	213.205.252	37

Table 4.3. Top 10 locations of fraud and legal (in fraud) events

In the legitimate datasets, most customers are from Maidenhead, Wales, Lewisham and Waterloo as shown in Table 4.4. Some of the locations appear twice in the tables such as Maidenhead and Waterloo, as these cities have more than one type of class-c networks. None of the top locations match between the two datasets. By removing another byte from the addresses, there is a wider range of the locations for each one.

Location	IP address (3 bytes)	Count
Maidenhead	93.40.254	257
Wales	215.183.128	186
Lewisham	98.197.127	176
Maidenhead	91.40.253	174
Waterloo	92.186.23	102
Waterloo	98.186.31	72
Cambridge	218.183.140	58
Maidenhead	99.41.251	46
Slough	81.132.248	46
Elmswell	198.176.105	32

Table 4.4. Top 10 Locations of Legitimate transactions

Figure 4.10 shows the transaction amounts by time for the three datasets based on their top locations. All the legitimate transactions in Maidenhead occurred between midnight and 1:40a.m and the maximum legal transaction amount is £2,500. The fraudsters in Canada tend to take smaller amounts, with three times £190, fifty times £20 and £10 once. These transactions happened between 10:00am and 3:00pm. The legitimate transactions in Brighton began from 8 o'clock in the morning up to midnight with no much variety of the amounts, whereas the legitimate transactions in Maidenhead show more diversity in terms of amounts.

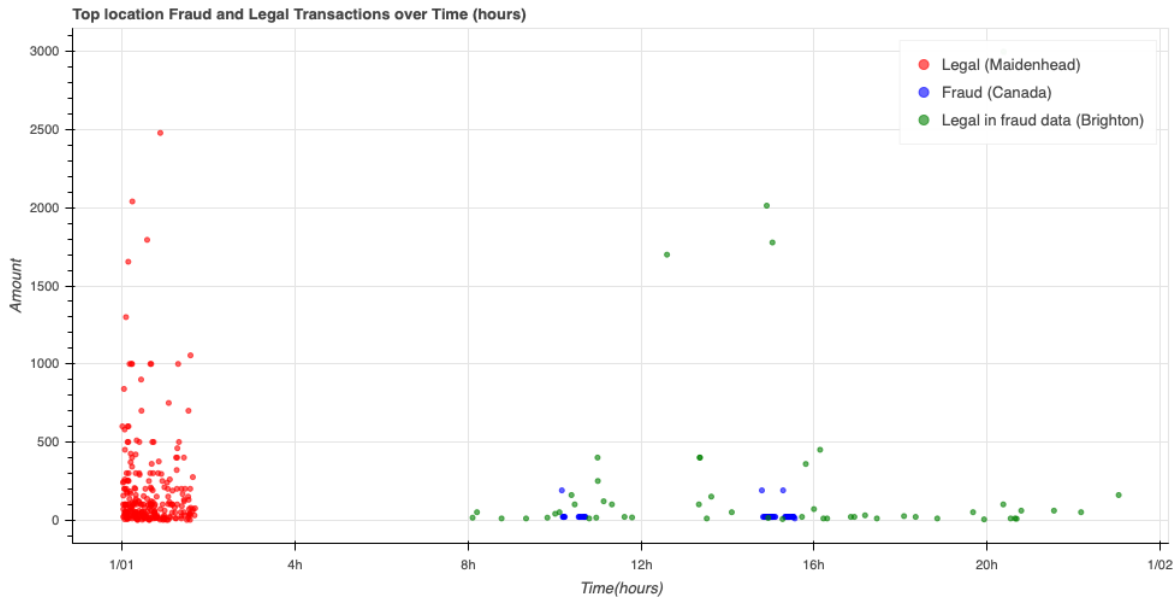


Figure 4.8. Top Location of Fraud and Legal Transactions Over Time in Hours

Another interesting variable is the User Agent. User Agent provides information about the browser and operating system of the device used to connect with the web server. The cross channel fraudulent events identified as shown in Table 4.5 (in total, 424) used Windows operating systems. The second most popular operating system for fraud is iOS with 238 user agents. Another popular operating system used by fraudster for transactions is Android on either tablet or mobile devices. Chrome OSs are not preferred by the fraudsters. From the table, Windows, iOS, Android and Mac OS are the top four most popular operating systems used by fraudsters, account for more than 98% of the total transactions.

Fraud		**Legal (in fraud)**		**Legal**	
Operating System Family	**Count**	**Operating System Family**	**Count**	**Operating System Family**	**Count**
Windows	424	**iOS**	1367	**Windows**	9530
iOS	238	**Windows**	1282	**iOS**	4275
Android	192	**Android**	577	**Android**	1524
Mac OS X	39	**Mac OS X**	101	**Mac OS X**	1195
Ubuntu	12	**Windows Phone**	58	**BlackBerry OS**	357

Windows Phone	7	**Chrome OS**	14	**Windows Phone**	49
Linux	3	**Linux**	7	**Ubuntu**	38
Chrome OS	1			**Linux**	37

Table 4.5. Popular operating systems used by customers in the datasets

With further investigation on the datasets, one finds that none of the detected events for both datasets occurred by a bot; 63% of legitimate transactions were carried out on PCs by customers; 53% of fraudulent records were made using PC and the rest are split between the mobile and tablet devices, with mobile accounting for the 39% and tablet for 8.6%.

In general, the preferred mobile devices are iPhone and iPad. However, a quite large proportion of the transactions occurred on some unidentified devices. In total there are about 30 different devices in the fraudulent dataset, 57 devices used for the legal transaction in the fraud dataset and 218 unique devices appear in the legitimate datasets.

4.2.3. Data pre-processing

As part of the data pre-processing, Feature Engineering technique is used for feature selection and development. The Feature Engineering building block examines and analyses the events collected from remote banking channel and create latent variables that will form a feature matrix that can be processed easily by intelligent algorithms such as neural networks. The features are generated through an iterative extraction-selection-validation process. First, a set of features are developed based on the raw data and RFM technique. Then the features are selected based on a set of rules. Finally, the features are analysed and validated by running a set of experiments. The process is repeated, and more features are subsequently identified.

The iterative process is applied to define and represent features of the online payment transactions. Let $X = \{x_1, x_2, \dots, x_t, \dots x_n\}$ be the set of all possible events of a customer in remote banking. Mathematically, it is the vector representation set of the information available for n events. Let f be a classifier with its binary range $C = \{trueFraud, falseFraud\}$, then one can define the payment fraud detection model as a functional mapping, $f: X \to Y$.

If one considers a customer journey on the Internet banking site such as customer login, adding a payeeor making payment, then the possible events X can be visualised as shown in Figure 3.1 of chapter 3. Each page shown as a circle is represented as a “state” of a customer and each “event” is represented as “state transition” which is shown as an arrow. The description of the states for an online payment transaction is shown in Table 3.5 of Chapter 3.

The information describing the events in Remote Banking Fraud Detection Framework includes transactional data, personal and account data, and behavioural data. Transactional data captures key attributes of a customer transaction that defines their characteristics and generally stored in Online Transaction Processing (OLTP) relational databases. This includes monetary events (i.e. Account Bill Pay) as well as service events (i.e. change of address). When applied to the fraud domain, this type of data can be used in the setting of RFM where data can be aggregated in accordance with a various time dimension (109).

The input data have a different range of values and types. These ranges could vary widely, which may impact the model performance. Data normalisation is then carried out by fine-tuning the input variables to bring the entire probability distribution values into alignment. Different data normalisation techniques such as shifting, and scaling can be applied to eliminate the effects of certain gross influences. Feature set generation is a process of selecting a set of features that are relevant for the model to perform better. The technique of RFM is used to create non-normalised data that can be used to facilitate further analysis. According to the data model, there are different data types associated with different attributes such as numeric, categorical, nominal. Page and transition times are numeric whereas page id is a categorical value. One-hot encoding (117) is applied for categorical values by listing down all the different states or web pages of a transaction. Further illustration based on specific model cases is included in the next section.

Subsequently, an object-oriented data model is developed to provide a unified view of the dataset. Four data classes to model the states, events, page navigations, customers and remote banking transactions have been identified, i.e., WebPage, PageNavigationGraph, Customer and BankingTransaction as shown in Figure 4.11.

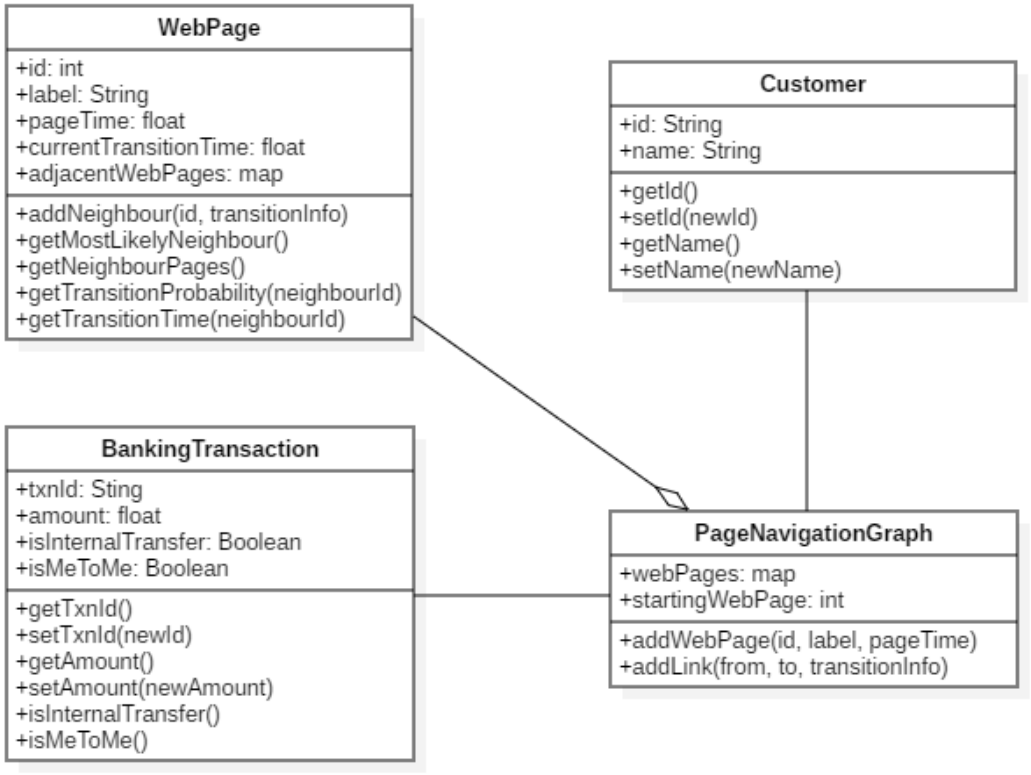


Figure 4.9. Class Diagram for Remote Banking Fraud Detection

Here, WebPage models the different states of a remote banking transaction. It stores information such as page id, page latent time, page label and information about the neighbour web pages and their corresponding transition time and probability. It has operations for adding neighbour pages, retrieving the most likely neighbour and all potential neighbours, retrieving transition probability and time of a neighbour. PageNavigationGraph is a set of related web pages that represent a remote banking transaction such as payment transaction. It stores the starting web page and other related web pages are retrieved traversing the page navigation graph.

For the numeric values such as page duration time and transition time (i.e., latent time), the mean, rescaling and decorrelation techniques are applied so that the input sequence data can be modelled by a function approximator, i.e., Gaussian Mixtures Models (GMM) (109) to provide smoother standard distribution with zero mean and standard deviation as one. The Min/Max Standardising technique (7) to scale the numeric features such as page and transition times is also used. In the datasets, there are different data types associated with different attributes such as numeric, categorical and nominal data types, where page and

transition times are numeric whereas page id is categorical. One-hot encoding (134) is applied for categorical values by listing down all the different states or web pages of a transaction.

From the proposed conceptual framework for remote banking fraud detection, an online payment transaction is a sequence of states and state transitions. Each web page is a state. The events (e.g., submitting the form or clicking on links to navigate from one web page to another web page) are state transitions. In our case here, a state is defined by dwell time which is the time spent by a customer on a web page and a state transition is represented by flight time which is the time taken to navigate from one state to another. Therefore, a transaction sample that includes four web pages is a sequence of eight-time steps represented by either a dwell time feature or flight time feature as shown in Figure 4.12.

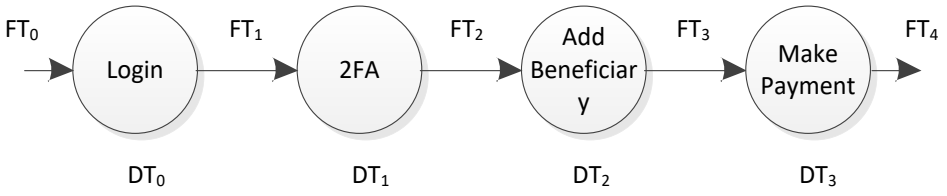


Figure 4.10. Representation of the Payment Transactions as a Sequence of States and Event Transitions

As the learning task of an LSTM model is to classify a payment transaction sample, X, into trueFraud or falseFraud, each sample is labelled using probabilistic score either as [1, 0] for a falseFraud sample or [0, 1] for trueFraud as shown below.

$$X = \{FT_0, DT_0, FT_1, DT_1, FT_2, DT_2, FT_3, DT_3\}$$

$$Y = \begin{cases} [1,0], falseFraud \\ [0,1], \ trueFraud \end{cases} \quad \text{(eq. 4.1)}$$

Where Y is the class label of sample, DTi is the dwell time or time spent on page i, FTi is the flight time or transition time from page i-1 to page i.

Most of the features in both datasets have missing data. Observations with just few missing values (e.g. < 0.1%) are not removed in order to avoid bias in the models. Bespoke data pre-processing is also applied to specific models.

More specifically, for Support Vector Machine (SVM) model, one-hot encoding is used mainly for features whose data types are string. One-hot encoding is a method that maps each category to a vector that represents value of 1 when the feature is presence and value of 0 otherwise. The number of vectors defer based on the number of categories (135). This method has a numerical nature; hence a machine learning model can easily incorporate such categorical feature information by learning a separate parameter for each dimension (136). One-hot encoding is applied on IP Address and User Agent attributes. Table 4.7 shows a sample of one-hot encoding representing the information of IP addresses.

IP Address	90.205.141.119	93.186.22.232	217.41.36.115
90.205.141.119	1	0	0
93.186.22.232	0	1	0
217.41.36.115	0	0	1

Table 4.7. One-hot encoding representation of IP Address

Transaction Amount is another feature applied for identifying fraudulent transactions. It is used by a model in its original numerical format. The four features for SVM model are applied with different combinations in order to find the best combination to predict fraudulent transactions.

For Markov Model, the data is extracted vertically as a sequence of time stamped events based on individual customers and the time of each event occurring. Combining fraudulent and non-fraudulent event sessions grouped by customer are aggregated to 64,538 sessions. Table 4.8 shows the most frequently occurring event sequences for both fraudulent and non-fraudulent transactions. It is observed that both datasets have the same sequences at the top four positions. Hence, it can be said that these four sequences would not be able to predict if a transaction is fraudulent.

Event Sequence	Frequency
Fraud	
Customer Login	11601
Customer Login -> Internal Transfer	1078
Customer Login -> MakeFasterPayment	616
Customer Login -> MakePaymentP2P	250
Customer Login -> AddPayeeP2P -> MakeFasterPayment	200
Customer Login -> Amend Telephone Number	187
Non-Fraud	
Customer Login	61046
Customer Login -> Internal Transfer	6044
Customer Login -> MakeFasterPayment	1697
Customer Login -> MakePaymentP2P	1598
Customer Login -> Internal Transfer -> Internal Transfer	658
Customer Login -> MakeFasterPaymentBillPay	389

Table 4.8 Most frequent occurring sequences in the fraudulent and non-fraudulent data

Further observation shows that the fraudulent data has 324 unique sequences whereas the legitimate data has 494 unique sequences. In order to limit the number of states in the model, event sequences exceed the 97th percentile of the observations are obtained from the data. Overall, there are 42 fraudulent and 14 non-fraudulent sequences that exceeded the 97th percentile. Using the 42 fraudulent sequences, frequent state of events is extracted in order to construct a mapping. These state events are shown in Table 4.9.

State Events	
Add Payee	Make Faster Payment P2P
Add Payee Recurring	Add Payee P2P
Customer Login	Make Payment P2P
Make Faster Payment Bill Pay	Amend Telephone Number
Device Registration	Amend Password
Session Update	Successful Password Reset

Add Payee Bill Pay	Internal Transfer
Amend Email Address	Make Faster Payment Bill Pay

Table 4.9. State events above the 97th percentile

This mapping is implemented on the training data by encoding the states and in case there are additional states, which are not included in the table above, are grouped together and labelled as zero.

For the Markov model to make predictions, it is necessary to calculate the dwell times between two state events in a sequence over a specified sampling period. However, the dwell time for the final event session is not given, thus for those events a dwell time value is assumed. The value is set to 1 second as it is the minimum dwell time that exists in the datasets.

For the Long Short-Term Memory (LSTM) model, a sequence of events based on each customer is extracted for both fraudulent and non-fraudulent data. Because not all the events are fraudulent in the fraudulent data, only the events flagged with 'Fraud' are considered in order to limit the sequence length of each customer. Therefore, the fraudulent data contains 4,945 rows and the legitimate data 100,000. Figure 4.13 shows the sequence length for each customer from both datasets. Most of the customers associated with fraudulent datasets have the sequence length less than 100 events, with some outliers that exceed the number. The legitimate customers all have sequence length up to 40 events except one customer with 80 events.

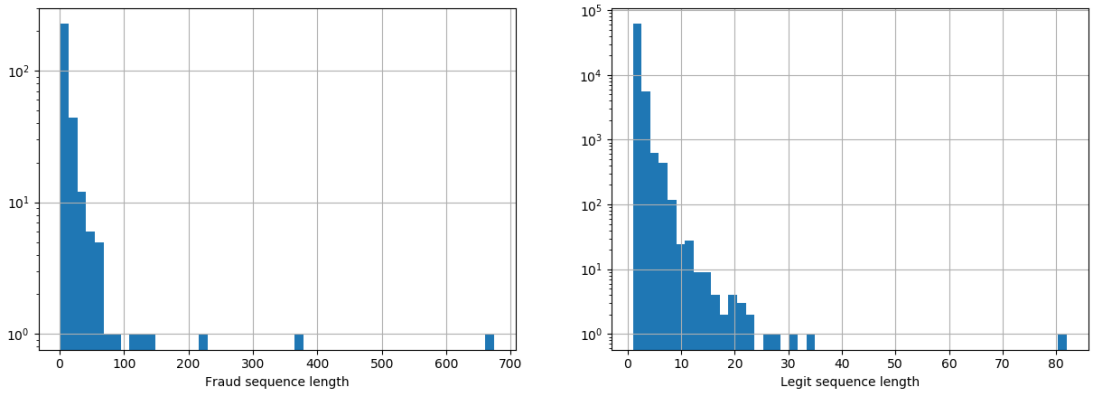


Figure 4.11. Fraudulent and Legitimate Sequence Length

Comparing the two figures shows that the lengths of the most fraud event sequences are longer than most of the legitimate sequences. In the model, a maximum sequence length is set to reduce the complexity of the modelling and meanwhile have a good comparability without jeopardizing its prediction accuracy. A sequence length of 40 is set because most non-fraudulent event sequence lengths lie between 1 and 40.

Before being fed into the model, the data is randomly shuffled so that the fraudulent and non-fraudulent sequence categories are not clustered. One-hot encoding is used to transform the event types from categorical data into integer. The encoding size needs to be consistent between runs. However, due to the randomness of fraudulent path selections the encoding size varies. Therefore, the maximum possible number of unique event types (here, 100) is applied to keep consistency. After applying the one-hot encoder, the dataset is normalized into a three-dimensional format. The number of the rows is 68,727 with 40 columns or so-called sequence length and 100 one-hot encoded events.

In practice, TensorFlow library is used for the model development. A two-dimensional array is used to store all customer sequences with fix length, where the sequences whose lengths are less than the array length are filled with zeros at the right side to meet the length.

4.2.4. Data sampling

To overcome the lack of large dataset, especially the scale of fraudulent dataset, a sampling technique is applied. In practice, a synthetic dataset which combines the real and simulated datasets are developed. The simulated dataset is generated based on the real remote banking events using probabilistic generative models on the set of features that are time dependent such as time spent on a page and time taken between the pages.

The data sampling technique is developed based on the GAN described in Chapter 3, that uses Gaussian Mixture Model. It supports unsupervised methods by applying the Expectation Maximisation Algorithm (EM) to adjust component distributions, which is typically Gaussians (137). The EM-algorithm is derived from Maximum Likelihood Estimation (MLE) given by;

$$L(\theta) = \prod_j^n P(\theta_j|x_j) \quad \text{(eq. 4.2)}$$

Where, x_j is the training data and θ_j is individual Gaussian components. The use of the product rule is to ensure that conditional probabilities are uncorrelated. EM is then given by,

$$\underset{\theta_j}{\operatorname{argmax}}[L(\theta_j)] \quad \text{(eq. 4.3)}$$

A typical application of the Gaussian Mixture Model (GMM) is to approximate probability distributions from sample data, with the resolution being set by the number of components (Gaussians) being fitted along with a variance regularizer. Each data point is treated as a component with independent mean and variance (138). By sampling the model's output represented as probability distribution function of real user's data, the simulated transactional dataset samples are generated as shown in the Figure 4.14.

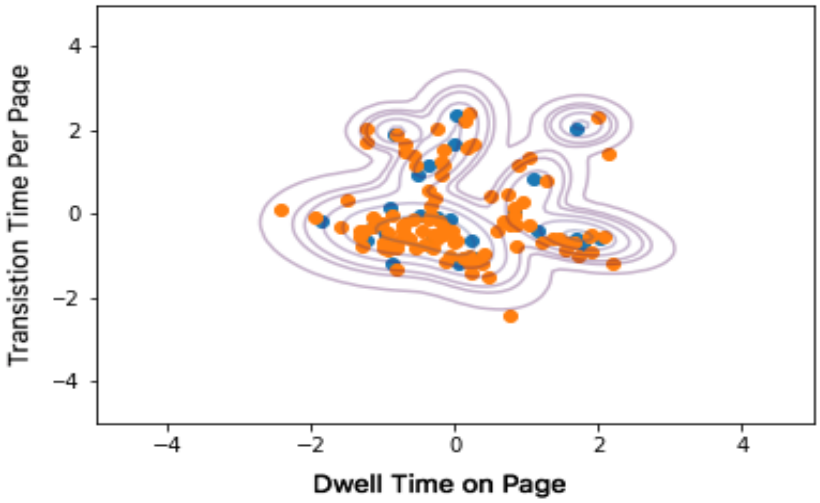


Figure 4.12. Example of a Generated Sample Base on GMM

The blue points represent original data and the red points represent generated data produced through random sampling. The contours are drawn to highlight the learnt probability distribution from the probability distribution function at different level (i.e. 0.001, 0.01, 0.02, 0.5). This technique resulted in the generation of 500,000 transactions that was representative of 80% of legitimate transactions and 20% of fraudulent transactions.

To compare the classification performance of LSTM and SVM, three data scenarios are developed by varying the size and non-linearity complexity of the generated datasets. The non-linearity complexity of the generated dataset is controlled by two parameters, i.e., starting point (b1) and width (b2) of a truncated normal distribution. Three different combinations of b1 and b2 are used to set up the three scenarios. In each group of the modelling, the size of the sample varies from 20,000 to 500,000. An example of the generated datasets is shown in Figure 4.15.

Figure 4.13. Example Demonstration of Dataset Generated

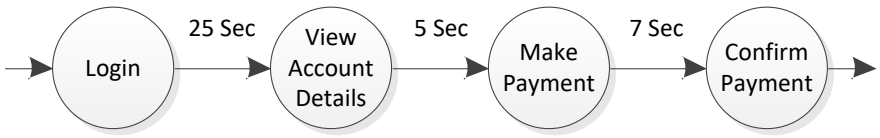
In the above graph, red dots represent the real user journeys in terms of time spent on a page and time spent on navigating to a different page. The green dots represent synthetic data derived based on GMM. The sample ratio of fraud to non-fraud and test to train ratio may vary for the three groups. The method gives a well coverage of data points centred at the mean as well as outliers.

4.3. Model implementation

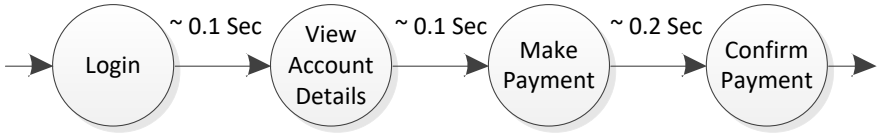
Three fraud detection algorithms in the proposed conceptual framework are developed, i.e., Long Short-Term Memory (LSTM) as a type of Recurrent Neural Networks (RNN), Support Vector Machine (SVM) and Markov Model. Event sequences are extracted from the fraudulent and non-fraudulent datasets. The models then perform the scoring and classification by sequence learning the temporal patterns of the entire customer history. Subsequently, the modelling results are evaluated, compared and analysed in the following chapter.

4.3.1. Recurrent Neural Networks

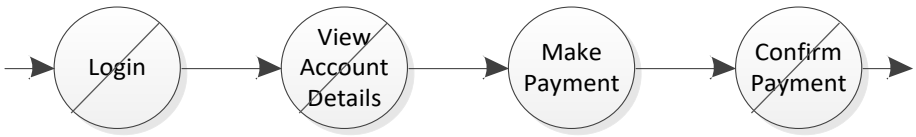
Recurrent Neural Networks can efficiently deal with temporal dependencies of the sequence classification problem which can be used for dealing with events from internet banking that is part of an internet banking session. The Universal Approximation Theorem for Recurrent Neural Network highlights that in the case of multiple hidden units it can arbitrarily map the input sequences to output sequences with reasonable accuracy (120). The usefulness of this theorem is shown in the following examples in Figure 4.16, which highlight how RNN can be modelled conceptually to map the input sequences and to perform fraud detection.



Example 1. Normal Customer journey for online banking



Example 2. Malware journey for online banking



Example 3. Non-Repudiation attack on online banking

Figure 4.14. Examples of Non-Fraud and Fraud Events

For normal customer behaviours, one may measure the information of an event, time spent at the event and latency introduced while moving from one event to another. The Example 1 here depicts a user navigating from one page to another on the bank's online portal. For example, the time taken for a user to log into a bank and navigate to Account Overview is approximately 25 seconds. Subsequently, the user navigates from Account Overview page to Make Payment page in approximately 5 seconds and so on. The Example 2 depicts a malware

journey for online banking, where malware logins and subsequently go through the same events as normal user in quick succession. Time spent on each event and in-between event here is minimum. A non-repudiation attack is when only the making payment event is observed as depicted in Example 3. This is also known as the "man in the middle attack" where an attacker captures the event and replays over after a while.

Let $X = \{x_1, x_2, \dots, x_t, \dots x_n\}$ be the vector representation set of the information available for n events of a customer in remote banking, where the event label t corresponds to the time t. Let us assume that the feature vector size of each event x_t is m, the input $X = (\mathbb{R}^m)$ are drawn from a distribution D, and the output space $Y = C^n$ be the sequences of $C = \{trueFraud, falseFraud\}$. The objective is then to use X to train fraud detection algorithm $f: X \rightarrow Y$ to label outputs in a test set $X' \subseteq D$ in a way that minimises a defined error measure. Here, the representation of the typical Recurrent Neural Networks (RNN), which calculates such a function, is shown in Figure 3.18 of chapter 3.

The RNN architecture is similar to a Neural Network architecture but weights are shared across evolving time stamp. As shown in Figure 3.18, RNN consists of an input layer that contains event at discrete time stamp (ts), a memory or hidden layer and an output layer. The number of input nodes is created based on the number of features in a feature set (x), the number of hidden nodes is created based on trial and error and the output is a single node (y). For online banking, the time-series of input feature vectors in matrix is shown below.

$$\bar{X} = \begin{bmatrix} id & latency & CFV & & & \\ id & latency & CFV & AFV & & \\ id & latency & CFV & AFV & TFV & \\ id & latency & CFV & AFV & TFV & \dots \end{bmatrix} \quad \text{(eq. 4.4)}$$

where id refers to a transition state such as login page, view bank statements page, make payment to new payeepage and change details page, and latency is the time difference between page navigation. The Context Feature Vector (CFV) feature includes IP address and malware flag etc., the Authentication Feature Vector (AFV) feature includes authentication type etc., and the Transaction Feature Vector (TFV) feature include transactional data such as transaction amount, transaction type and payeeaccount number etc.

The current state (t) of the hidden nodes depends on the objective function (122) that takes in the previous state (t-1) of the hidden nodes and the input vector at time t, as shown below;

$$h_t = f_w(h_{t-1}, x_t) \quad \text{(eq. 4.5)}$$

Where h_t is the new state, h_{t-1} is the previous state, x_t is the input vector at time t. This provides a feedback into the network and can be considered as introducing memory to the network. Since the network can hold memory it can be better utilised for classification and prediction tasks, when compared to other neural networks. Several research studies (122), (123), (124) showed that hyperbolic tangent have been widely used as an objective function. For the hidden layer of RNN above, one may rewrite the equation 5 as follows;

$$h_t = (\tanh(w_{hh}h_{t-1} + w_{hx}x_t)) \quad \text{(eq. 4.6)}$$

Where w_{hh} and w_{hx} are the weights between the previous state and current hidden nodes, and between input nodes and hidden nodes respectively. Tanh is a logistic function that is non-linear and continuously differentiable. This enables the study of the properties of neural networks for developing learning algorithms. The values of the output layer y(t) are predicted based on the current input, weighted by the coefficients wi of all the past input. The aim of the learning algorithm is to minimise the error by incrementally adjusting the parameters of the algorithm to optimise the objective function h_t during training. The prediction error, e(t), is;

$$e(t) = y(t) - \hat{y}(t) \quad \text{(eq. 4.7)}$$

For regression models, typically the mean squared error or squared Euclidean distances, i.e., E$[e^2(t)]$) is used, where the error is assumed to be statistically independent (122).

The classifier outputs 1 (i.e. trueFraud) if the fraud is detected or 0 otherwise (i.e., falseFraud). A probabilistic classification approach is preferred to calculate conditional probabilities $P(y|x)$. This allows the probabilities to be retained in a consistent manner. The classifier accepts xt and produces a probability score that belongs to the class C as defined previously. The output is based on the SoftMax function which outputs a cross-entropy loss by the equation as follows;

$$P(y = fraud\ |x_t) = \frac{e^{w_{hy}x_t}}{\sum e^{w_{hy}x_t}}, (0 \leq P \leq 1) \quad \text{(eq. 4.8)}$$

The output of the above equation outlines the probability of the event. If the probability is above a defined threshold the system will output 1 as fraud or 0 otherwise.

4.3.2. Long Short-Term Memory

For RNN to learn the sequences of events and classify them accurately, an error needs to be calculated and propagated back through the network using the gradient decent algorithm known as Back Propagation Through Time (BPTT) (123). The advantage of using BPTT is that linear and non-linear constraints between the shared weights can be easily incorporated. It will allow the model to generalise well to temporal dependencies of different states. For examples, let's say one started off satisfying the constraint of $h_1 = h_2$, and one needs $\Delta h_1 = \Delta h_2$. One can compute $\frac{\delta E}{\delta h_1} + \frac{\delta E}{\delta h_2}$ based on the model and error term for h_1 and h_2.

Based on the above, one may build the forward pass that determines user state at each timestamp and followed by a backward pass by computing the error derivatives at each time stamp. The derivatives are added together for all of the timestamps for each weight and then all the copies of the weight are updated by the same amount which is proportional to the sum of those derivatives.

As highlighted in the research (120), (123) the above algorithm fails to learn where there are long-time lags between the input events and their associated errors flowing back through time or when the network grows from t to $t + n$ (120), (123).

The LSTM network is used alongside hidden layer or as a hidden layer to overcome the vanishing and exploding gradient problem. The properties of the LSTM memory block are outlined as depicted in Figure 3.20 of chapter 3.

The architecture of LSTMs consists of three gates, i.e., input, forget and output gates. The purpose of the input gate is to control the input to a memory cell. Its core function is to allow data into the memory cell and to protect the memory cell from noise. An output gate is responsible for regulating the flow of cell activations into the rest of the network. The purpose of the forget gate is to forget the memory that has been acquired during earlier time stamp as the content becomes irrelevant.

The first step of LSTM is to update the forget gate that decides which information to remember and forget from the cell state (139). The equation for the forget gate is given by;

$$f_t = \sigma\,(W_f.[h_{t-1}] + b_f) \quad \text{(eq. 4.9)}$$

The second step is to update the cell state. First, a sigmoid layer decides which values of the state cell to update and then a hyperbolic tan (tanh) layer creates a vector of new candidate values that can be added to the cell state (139). The update equation is formulated as follows,

$$i_t = \sigma(W_i.[h_{t-1}, x_t] + b_i) \quad \text{(eq. 4.10)}$$

$$\hat{\mathrm{C}}_t = \tanh\,(W_c.[h_{t-1}, x_t] + b_c \quad \text{(eq. 4.11)}$$

The previous cell state (Ct-1) is updated to a new cell state (Ct) by combining equations (4–5). First by multiplying the previous cell state by ft (i.e., forgetting factor) and then add the new candidate values (Ĉt) scaled by it (i.e., the amount of update to each state value).

$$C_t = f_t * C_{t-1} + i_t * \hat{\mathrm{C}}_t \quad \text{(eq. 4.12)}$$

The final step of LSTM is to determine the output based on the new cell state and a sigmoid layer,

$$O_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \quad \text{(eq. 4.13)}$$

$$h_t = O_t * \tanh\,(C_t) \quad \text{(eq. 4.14)}$$

In practice, LSTM classifies web session event sequences into fraudulent and non-fraudulent categories, where fraudulent event sequences can be further classified as one of the types below.

Fraudulent type	Description
CoT	Change of Telephony
Sim swap	Fraudster contacts mobile network and obtains new sim in the name of the user, uses it to gain access to account
Social engineering - Device takeover	Remote access Trojan
Social engineering - Account at risk	Fraudster impersonates bank staff to gain access

Table 4.10. Fraudulent types for event sequence data modelling

For each customer a sequence of events is extracted for both fraudulent and non-fraudulent data. Because not all the events are fraudulent in the fraudulent datasets, only the events flagged as 'Fraud' is considered to limit the sequence length of each customer. Hence the final fraudulent datasets contain 4,945 rows and the legitimate datasets contain 100,000 rows. As the lengths of the fraud event sequences are usually much longer than the legitimate sequence lengths, a maximum sequence length is set to reduce the complexity of the modelling and meanwhile have a good comparability without jeopardizing its prediction accuracy. A sequence length of 40 is set because most non-fraudulent event sequence lengths lie between 1 and 40 in fraudulent datasets.

In practice, 80% of the datasets are taken as training datasets and 20% as testing datasets. The parameters of the model are tuned based on model performance. Other proportions and alternative model parameters are also tested at the model evaluation stage through a systematic approach in order to find an optimal performance in next chapter. The table below shows the parameters used by the model.

Parameter	Value	Parameter	Value
Learning rate	0.0014	**Keep probability**	0.5
Training steps	1000	**Hidden neurons**	85
Batch size	267	**Hidden layers**	2

Display step	10	Number of classes	2

Table 4.11 Tuned parameters of model

4.3.3. Support Vector Machine (SVM)

SVM is a supervised non-probabilistic machine learning method that acts as a discriminative classifier by a separating hyperplane (140) and outputs an optimal hyperplane when classifying new data given labelled training data. It is a popular machine learning technique used in the fraud detection domain and often ensembled with other machine learning techniques (73), (74), (75). SVMs are useful alternatives to neural networks when the learning objective is a non-convex problem represented as multiple local minima. In such circumstances, neural networks tend to get stuck in an area known as saddle point (76), (137). Another advantage of SVM over neural networks is the large number of hyperparameters that are required to be tuned in neural networks (137) which often slows down the training. A hyperplane is defined by the following function (140).

$$f(x) = \beta_o + \beta^T x \qquad \text{(eq. 4.15)}$$

Where, β is the *weight vector* and βo is the *bias*. An absolute value of the above function can be used to get an optimal hyperplane and can be expressed as;

$$\left| \beta_o + \beta^T x \right| = 1 \qquad \text{(eq. 4.16)}$$

where x is the closest training dataset to the hyperplane. This training dataset is called support vectors. The distance between x and a hyperplane (β, βo) is given by;

$$distance = \frac{\left|\beta_o + \beta^T x\right|}{\|\beta\|} = \frac{|1|}{\|\beta\|} \qquad \text{(eq. 4.17)}$$

The objective of the SVM algorithm is to find an optimal separating hyperplane that maximises the margin of the training dataset [39]. The maximum margin, for an optimal hyperplane, is given by,

$$M = \frac{|2|}{\|\beta\|} \quad \text{(eq. 4.18)}$$

Finally, in order to maximise M, one need to minimize an objective function $L(\beta)$;

$$L(\beta) = \frac{1}{2} * \|\beta\|^2 \quad \text{(eq. 4.19)}$$

Subject to constraints (140);

$$y_i(\beta^T + \beta_o) \geq 1 \ \forall_i \quad \text{(eq. 4.20)}$$

Where, y_i represents the label for each training data. The constraints imposed on the hyperplane is used to accurately classify training samples X_i.

The above minimization problem can be solved using Lagrange multipliers to obtain the weight vector β and the bias βo of the optimal hyperplane (140).

The implementation of SVM model is to find a linear separator in the feature space that can best separate legitimate and fraudulent training data. The data used for fraud detection is the payment records. In the fraud datasets, that is the payments flagged as 'Payment Fraud'. There are 770 fraudulent records and 8,350 legitimate records in total. Fraud detection is an imbalance class problem. Commonly, the majority of the data are legitimate transactions, and only a small proportion is fraudulent. One approach to rebalance the dataset and have equal classes is to use Synthetic Minority Over-sampling (SMOTE) technique (105). By using this technique, the minority class will be oversampled with synthetic samples.

Not all features are included in the model. Some of the features are not relevant to the fraudulent behaviours but only widen the high dimensional search space. To find the best combinations of the attributes for the model, a forward stepwise predictor selection process is executed by starting with one attribute/variable and continuing adding one by one, until the best performance is identified. For the variables with their data types being strings such as IP Address and User Agent, one-hot encoder is used. The encoder takes a column of categorical values and then converts them into a vector of 1s and 0s based on their presence in a scenario.

4.3.4. Markov Model

In general Markov property assumes the probability of an event depends only on the state attained in the previous event.

Let us assume the stochastic process $\{X(t), t \in T\}$ and its state space I. Then, the Markov property is expressed in conditional probability as,

$$P(X(t_n) = x_n | X(t_{n-1}) = x_{n-1}, \dots, X(t_1) = x_1) = P(X(t_n) = x_n | X(t_{n-1}) = x_{n-1}) \quad \text{(eq. 4.21)}$$

Where $x_i \in I$. A stochastic process $\{X(t), t \in T\}$satisfying Markov property is called Markov process. Markov chain is a special case of Markov process dealing with discrete time and states, which is which is often seen in applications. The equation above implies a transition per t or per step. A more general case may contain m steps.

$$P(X(t_n) = x_n | X(t_{n-m}) = x_{n-m}, E) = P(X(t_n) = x_n | X(t_{n-m}) = x_{n-m}) \quad \text{(eq. 4.22)}$$

For simplicity, here notion E is used to represent a subset of earlier events. Obviously, Markov chain is a one-dimensional random walk. The process starts from one state to another with a probability. The transitions between pair of states can be represented by a so-called transition matrix;

$$\begin{array}{c} \\ a_1 \\ a_2 \\ \vdots \\ a_q \end{array} \begin{array}{c} \begin{array}{cccc} a_1 & a_2 & \cdots & a_q \end{array} \\ \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1q} \\ p_{21} & p_{22} & \cdots & p_{2q} \\ \vdots & \vdots & \vdots & \vdots \\ p_{q1} & p_{q2} & \cdots & p_{qq} \end{bmatrix} \end{array} = P(m) \qquad \text{(eq. 4.23)}$$

Here state space $I = \{a_1, a_2, \dots, a_q\}$ and m represents the number of steps. $P(m)$ can also be written as $\left(p_{ij}(m)\right), i, j \in [1, \dots, q]$. To calculate $P(m)$, according to the Chapman-Kolmogorov formula (122), for any time $u, v \in T$,

$$P_{ij}(u+v) = \sum_{k=1}^{q} P_{ik}(u) P_{kj}(v) \qquad \text{(eq. 4.24)}$$

Which states that the probability of going from state a_i to state a_j through $u+v$ steps can be found from the probabilities of going from a_i to an intermediate state a_k through u steps and then from a_k to a_j through v steps. In a matrix format,

$$P(u+v) = P(u)P(v) \qquad \text{(eq. 4.25)}$$

Based on the equation above, one obtains;

$$P(m) = P(1)P(m-1) = PP(m-1) = \cdots = P^m \qquad \text{(eq. 4.26)}$$

That is, the calculation of the transition matrix of steps m is equal to the one-step transition matrix P to the power of m.

One may also be interested in the state probability distribution along the time (i.e., steps), m. For each state a_i along the timeline m, one has the state probability (represented by lower case p) after the time $t = m$,

$$p_i(m) = p\{X(m) = a_i\} \quad \text{(eq. 4.27)}$$

Obviously,

$$\sum_{i=1}^{q} p_i(m) = 1 \quad \text{(eq. 4.28)}$$

The equation gives probability of states. The equation can be further deduced based on Bayes Theorem,

$$\begin{aligned} p\{X(m) = a_i\} &= \sum_{k=1}^{q} p\{X(0) = a_k, X(m) = a_i\} \\ &= \sum_{k=1}^{q} (p\{\, X(m) = a_i \mid X(0) = a_k\} \cdot p\{X(0) = a_k\}) \end{aligned} \quad \text{(eq. 4.29)}$$

Based on the transition matrix, the equation can be written in short,

$$p_i(m) = \sum_{k=1}^{q} \left(P_{ki}(m) \cdot p_k(0)\right) \quad \text{(eq. 4.30)}$$

Or vector-wise,

(eq. 4.31)

$$\boldsymbol{p}(m) = \boldsymbol{p}(0)P(m)$$

Where, $\boldsymbol{p}$ is the vector of states and P is the transition matrix. It shows that the distribution of Markov chain at time m is determined by its initial distribution and transition matrix.

Markov Model is often used for modelling time series data, where it represents the probability distribution over sequences of observations. In our model, the data is extracted as a sequence of time stamped events based on individual customers and the time of each event occurred. Combining fraudulent and non-fraudulent event sessions grouped by customer aggregate 64,538 sessions. Examining the most frequently event sequences, there are 324 fraudulent and 494 non-fraudulent frequent sequences. The best way to limit the number of states in the model is to obtain event sequences above the 97 percentiles according to occurrences in an incremental order. Based on that, default states are extracted in order to construct a mapping and implement it on the training data. These state events are depicted in Table 4.12 below;

No. of states	Names of states
1	AddPayeeP2P
2	AddPayeeRecurring
3	CustomerLogin
4	MakeFasterPaymentBillPay
5	DeviceRegistration
6	AddPayeeBillPay
7	SessionUpdate
8	AmendEmailAddress
9	MakeFasterPayment
10	AddPayeeP2POnUs
11	MakePaymentP2P
12	AmendTelephoneNumber

13	AmendPassword
14	SuccessfulPasswordMIReset
15	InternalTransfer
16	MakeFasterPaymentBillPayOnUs

Table 4.12. States of online banking event sequences

The states are encoded in the Markov model. In case there are additional states, which are not included in the list above, they are grouped together and labelled as zeros. In order to estimate the model, it is necessary to calculate the dwell times between two state events over a specified sampling period. For the final event sessions whose dwell time is not given, the minimum dwell time value of the datasets (i.e.1 second) is used.

The model predicts the fraudulent events by using the average log likelihood $\mu_\theta(X_{1:N}) = \frac{1}{N}\sum_{n=1}^{N} \log P\,(X_n|\theta)$ of the observed sequence $X_{1:N}$ in a length normalised manner. If the scores are not normalized, there is a high probability for the classifier to predict fraud and non-fraud based on the sequence length due to its larger values. The use of log likelihood for the predictions is because it simplifies the mathematical analysis and signifies the small probabilities that would otherwise be difficult to manipulate and represent.

Classifying the observed sequence, the Markov model is applied for each class of the observed training data. Using fraudulent and non-fraudulent training data the normalised log likelihoods $\mu_{\theta_{fraud}}(X_{1:N})$ and $\mu_{\theta_{nonfraud}}(X_{1:N})$ are calculated respectively. And the classification score is obtained by the normalised log likelihood ratio $\mu_{\theta_{fraud}}(X_{1:N}) - \mu_{\theta_{nonfraud}}(X_{1:N})$. For each of the two classes in both training and testing data, an arbitrary count value is set to 1 for each event in order to overcome the problem where an event has zero frequency.

4.3.5. A systematic modelling process

The models are developed based on LSTM, SVM and Markov techniques for detecting fraud from event sequences. And subsequently, their performance is analysed and evaluated. In practice, Python with Google's Tensor Flow library is used to build the integrated Payment Fraud Detection System. The general modelling process to implement the models for

detecting fraudulent remote banking transactions is shown in Table 4.13 and a corresponding flowchart outlining the general flow of the implementation is also given in Figure 4.17.

1	*Generate datasets based on Feature Engineering and prior knowledge of the data.*
2	*Data normalization based on each feature.*
3	*Split dataset into train, validation, and testing.*
4	*Define model structures and functions, including input and output variables.*
5	*Initialize model parameters (memory size, learning rate, batch size and epochs).*
6	*Compute the outputs based on training and validation datasets.*
7	*Repeat:*
8	*1. Compute training error.*
9	
10	*2. Compute validation error.*
11	*3. Minimize errors.*
12	
13	*4. Update model parameters.*
14	*While epochs are less than maximum.*
15	*Predict for testing dataset using trained models.*
16	*Produce model performance indicators, e.g., confusion matrix and F1 score.*
	Iterative model evaluation (steps 3-15) using various datasets.
	Model selection for prediction.

Table 4.13. Modelling Process

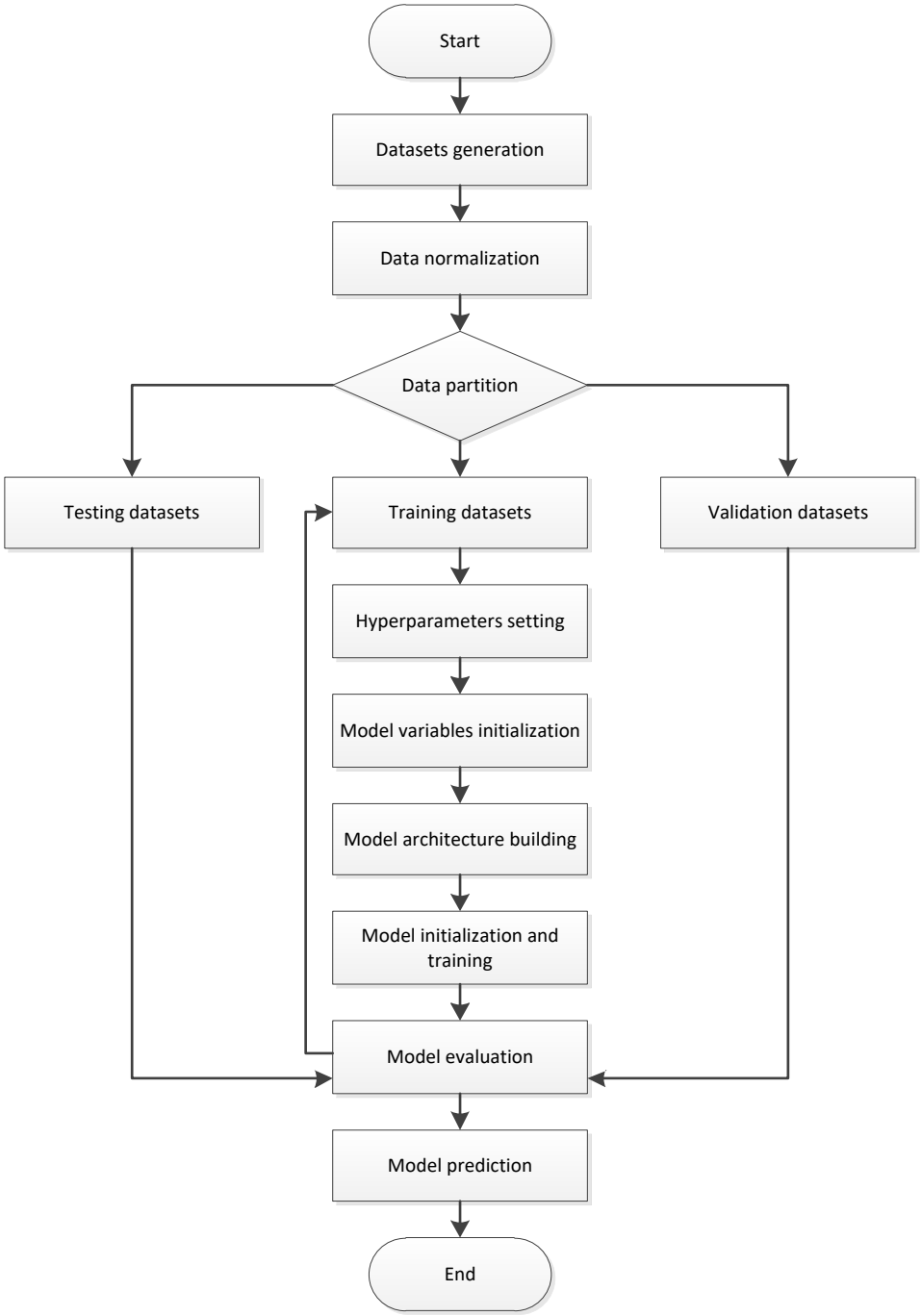


Figure 4.15. Modelling Process for Remote Banking Fraud Detection

4.4. Conclusion

The sophistication of the fraud Modus Operandi (MO) requires us to rethink on the Payment Fraud Detection System approaches. Meanwhile, intelligent methods such as SVM and LSTM are not well utilized in remote banking fraud detection. In this chapter an integrated Payment Fraud Detection System with three models, i.e., Support Vector Machine, Markov Model and LSTM model are implemented based on the framework developed in the previous chapter. It

is capable of automated training and tuning of the models based on thresholds with high F1 scores and novel analytics.

Detecting fraud is a no-easy computational task and the searching for the optimal model is complex, which can be influenced by model types, model characteristics and the data itself. In this research, the data collected for the analysis is imbalanced, with 22,880 rows of fraud datasets and 100,000 rows of legitimate datasets. The exploratory analysis is applied to both legitimate and fraudulent datasets for feature investigation and selection. Only features exhibiting the identities of fraudulent and legitimate events such as Amount, Time, IP address and user agents are selected.

In the exploratory analysis, the distributions of the datasets show a set of characters. For example, most of the fraud happened between noon and 4:00pm; the popular months for fraudulent activities are April, March and May, most frequently at the start and end of the months; and the transactions turn to be specific country-oriented and device-specific.

The **SVM** model is trained by starting with one variable and continue by adding one by one based on features, Amount, User agent and IP address. The model performed better with Amount as the only input. **It gave an F1 score 71%, an EER of 0.3 and the lowest False Positive Rate of 27%.** The overall time to run the specific model was 26 seconds. **Markov model** uses frequent sequences of states of events classifying fraud and non-fraud by calculating the normalised log likelihood of each class. The model is compared among 5 sampling rates – 20s, 10s, 5s, 2s and 1s. **The lowest EER (≈ 0.07)** is obtained with the sampling rates 5s and 2s, whereas **the highest accuracy (≈ 0.97)** is acquired with the sampling rate of 20s. Lastly, the **LSTM-based Recurrent Neural Network model** is used to uncover suspicious activities on the retail banking channel. The LSTM classified event sequences by customer and achieves an EER of 0.05 approximately and an **overall accuracy of 97%. The lowest False Positive rate is 3.6%.** The overall running time of this model is 3 seconds whereas the Markov model runs 55 minutes.

4.5. Key Summary

This section describes the high-level summary of the content of this chapter. This chapter follows on from Chapter 3 where a conceptual framework was highlighted and in this chapter, it is applied on real dataset.

- Discussed framework in detail for developing CCFDS with a real-life scenario dataset from a large European bank. The conceptual framework was described in Chapter 3.
- Details data source and outlines exploratory analysis to understand and determine the probability distributions of the datasets and the key features used by the model.
- As part of pre-processing, latent features are created based on feature engineering technique described in Chapter 3.
- LSTM, SVM and Markov Model are explored in this chapter and the real dataset can be applied to these models.
- A systematic modelling process is created to evaluate the performance of the models.

5.

CCFDS – Evaluation and Validation (Comprehensive comparative analysis and evaluation of the sequence learners)

5.1. Modelling results and analysis

Applying the technique SMOTE for balancing the data between the two classes before training the SVM model, the following sets are obtained as depicted in Table 5.1 below:

	Fraud	Non-Fraud	Total
Training	6680	6680	13360
Testing	1670	1670	3340

Table 5.1 Datasets split into training and testing

The feature selection is based on a forward selection process by adding the features one by one to the SVM model. The features are normalized before fed into the model. The evaluation (5.1) of results is based on the computation of the Receiver Operating Characteristic (ROC) curves and F1 score. The F1 score gives the harmonic mean of precision and recall based on the equation,

$$F_1 = \left(\frac{precision^{-1} + recall^{-1}}{2}\right)^{-1} = 2 \cdot \frac{precision \cdot recall}{precision + recall} \qquad \text{(eq. 5.1)}$$

where,

$$Precision = TP/(FP + TP)$$

$$Recall = TP/(TP + FN)$$

Here TP, FP and FN represent True Positive, False Positive and False Negative respectively. The scores corresponding to each class give the accuracy of how well the SVM model classifies the data of a class compared to another class. Table 5.2 shows the F1 scores fraud and non-fraud class for individual or combinational features. The features are analysed in the exploratory analysis section.

As it can be noticed from the table, as more variables are added to the model, the F1 score drops for both classes. The experiments show that the classifier is capable of detecting non-fraudulent transactions with good accuracy. The combinational features, Amount and IP address, score the lowest. The best performance according to the F1 score is given by the individual feature, Amount with 72%. However, observing the ROC curves in Figure 5.1, the model with combinational inputs, i.e., Amount and User Agent, achieves highest accuracy with a moderate Equal Error Rate (EER) (≈ 0.327). It also suggests that the Amount and IP address variables should be better used with the User Agent variable, which scores 68% accuracy and with 0.366 EER. The optimal thresholds for all the cases are between 0.45 and 0.51.

F1 score	Amount	Amount, User agent	Amount, IP address	Amount, User Agent, IP address
Non-Fraud	0.72	0.67	0.60	0.63
Fraud	0.72	0.61	0.62	0.61
Average	0.72	0.66	0.61	0.62

Table 5.2. F1 score of fraud and legit class over a combination of features

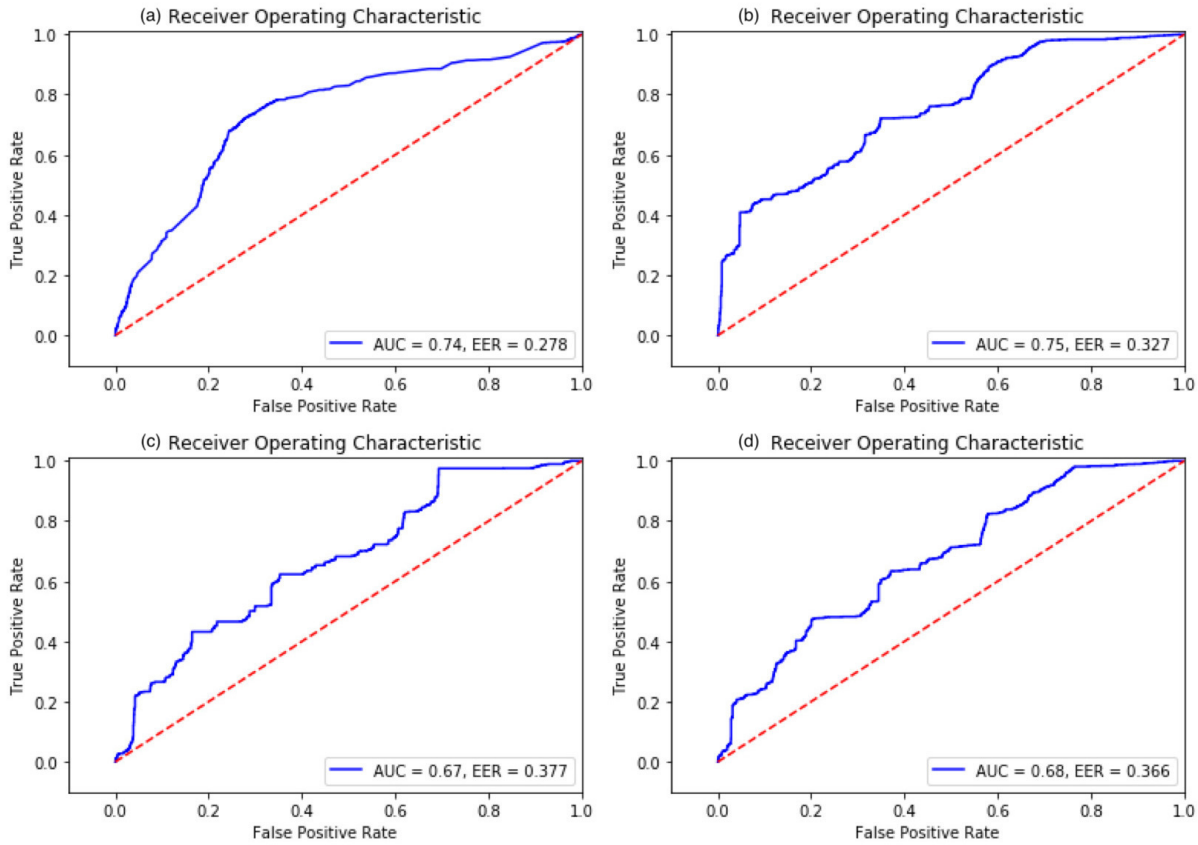


Figure 5.1. ROC curves of SVM model with inputs (a) Amount, (b) Amount and User agent, (c) Amount and IP address and (d) Amount, User agent and IP Address

Markov Model and LSTM used a different approach for fraud detection. Sequences of events are taken from data and imported into the models. The Markov model was evaluated using the 3-fold cross validation in an arbitrary selection of set of sampling rates, i.e., 20s (i.e., seconds), 10s, 5s, 2s and 1s. A rate lower than 1ms does not provide significant results as the sampling rate of 1ms is the lowest time span in the dataset. Figure 5.2 presents the ROC curves with the five sampling rates. The difference between sample rates are not significant, where the sampling periods of 20ms and 10ms achieve slightly better Area Under the Curve (AUC), equivalent to 96.8% and 96.3% respectively. However, the two lowest EERs, both approximately 0.066, are obtained with 2ms and 5ms.

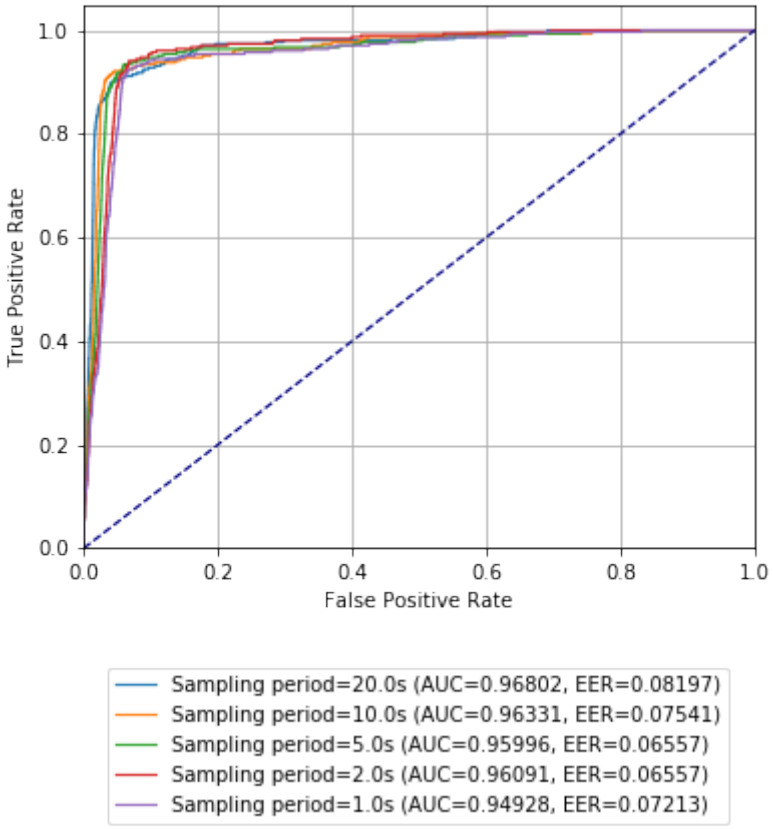


Figure 5.2. ROC curves of Markov model for sampling rates 20s, 10s, 5s, 2s and 1s

The bidirectional LSTM model is developed with specified tuned parameters. The testing datasets contain 13,690 legitimate events and 56 fraudulent events. With the data the LSTM model running time is 120 seconds to produce the prediction results. The results show that it performs better on legitimate sequence events with an F1 score close to 99% as shown in the confusion matrix in Figure 5.3. The F1 score on the fraudulent events is 92%. Out of the 56 fraudulent events, 52 events are predicted as fraud and the other 4 as legitimate. The overall model prediction rate is 97%.

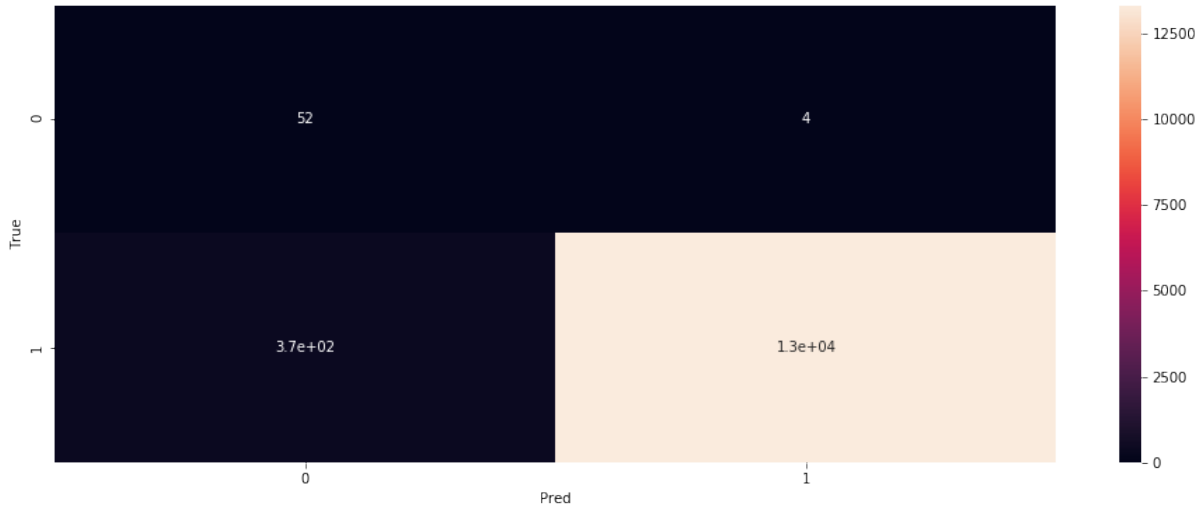


Figure 5.3. Confusion matrix of the LSTM model

The classifier achieves an AUC (Area Under the Curve) equal to 99% and an error rate of 5%, which is lower than the Markov Model error, as illustrated in Figure 5.4.

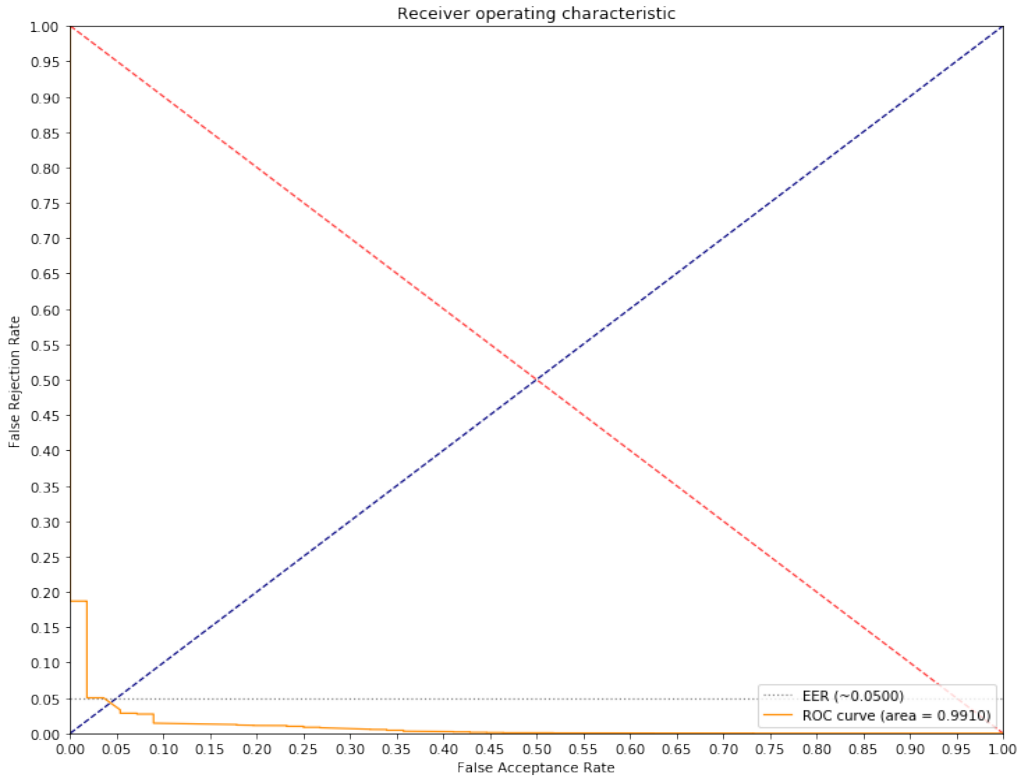


Figure 5.4. ROC of the LSTM model

The highest performance of the model is achieved when the threshold is 0.6, giving the lowest rates in False Positive and False Negative. False Positive Rate with 3.6% means the model classifies a small proportion (i.e. 3.6%) of the fraudulent events as legitimate. This rate is

equivalent to the 95% of True Positive Rate (i.e., the proportion of the legitimate sequence events are classified correctly as legal). The model classified only 5% of the legal events as fraud (False Negative Rate).

5.2. Model evaluation (SVM, RNN, Markov Model)

5.2.1. Model validation

The model validation may refer to verify model performance and limitations, and justify the model results based on data, statistical and modelling techniques. In the research, a grid search technique is used to find optimal parameters of the SVM models. The grid search technique builds a model on each possible parameter combinations. It iterates through every parameter combination to find the optimal model (141). The learning parameter ranges set for SVM in prior are shown in Table 5.3.

Parameter Name	Value
Kernel	Radial Basis Function (RBF)
Cost (C)	[0.001, 0.01, 0.1, 1, 10]
Gamma coefficient	[0.0001, 0.001, 0.01, 0.1, 1]
K-fold cross validation	5

Table 5.3. Learning parameters of SVM

The Radial Basis Function (RBF) kernel is fixed in SVM model as it is useful for the nonlinear classification and meanwhile the linear kernels used by SVM model are often computationally expensive and time consuming. Five possible values are set for gamma coefficient i.e. 0.001, 0.01, 0.1, 1 and 10 and five cost values or Cs, i.e. 0.0001, 0.001, 0.01, 0.1 and 1 are also set. Then the optimal values of C and Gamma are determined automatically by the grid search technique which searches all possible combinations to the optimal based on cross validation (here, 5-fold). The best performance is achieved by a parameter combination with Gamma 1 and C 10.

The datasets are then split into training, validation and testing dataset by 80%*80%-80%*20%-20%. Table 5.4 shows the F1 scores of the four SVM models based on the validation and testing datasets, where SVM 1 models the Transaction Amount on, SVM 2 models the

Transaction Amount and IP address features, SVM 3 takes features Transaction Amount and User agent and SVM 4 has all the three features i.e. Transaction Amount, IP address and User agent. SVM 1 has the lowest performance in all models while model 2 produces the best performance. It indicates that the two features, Transaction Amount and IP address, are most relevant to the identification of fraudulent and legitimate events. All these models outperform the results of the default model as shown in Table 5.4.

Model name	Features	Validation dataset	Testing dataset
SVM 1	Amount	0.7219	0.7155
SVM 2	Amount, IP address	0.9463	0.9225
SVM 3	Amount, User agent	0.9378	0.9364
SVM 4	Amount, IP address, User agent	0.9418	0.9384

Table 5.4. F1 scores of four SVM models based on features, validation and testing datasets

The LSTM model learning parameters are similarly tuned to determine the optimal parameter combination, as shown in Table 5.5.

Parameter Name	Value
Learning rate	[0.1, 0.01,0.001, 0.0001]
Epochs	1000
Batch size	100
Number of layers	2
Number of hidden units	85
Loss function	Soft-max Cross Entropy
Optimiser	Adam Optimiser
Number of features	100
Number of classes	2

Table 5.5. Learning parameters of LSTM model

In LSTM model, the Soft-max Cross Entropy and Adam Optimiser are used as loss function and optimiser respectively. The learning rate is chosen from a range of values i.e. (0.1, 0.01, 0.001, 0.0001). By testing, the optimal learning rate 0.001 is obtained.

Figure 5.5 shows the accuracy and loss levels over the number of training steps (here, 0 – 1000 epochs) that the model takes. As the number of epochs increases, the training and validation loss instability decreases, and the loss value reaches zero. It means that the over fitting does not occur. On average, the accuracy is above 80% and the loss is close to zero. Fitted with the unseen datasets, the LSTM model manages to correctly predict almost all fraudulent transactions and a big proportion of the legitimate transactions.

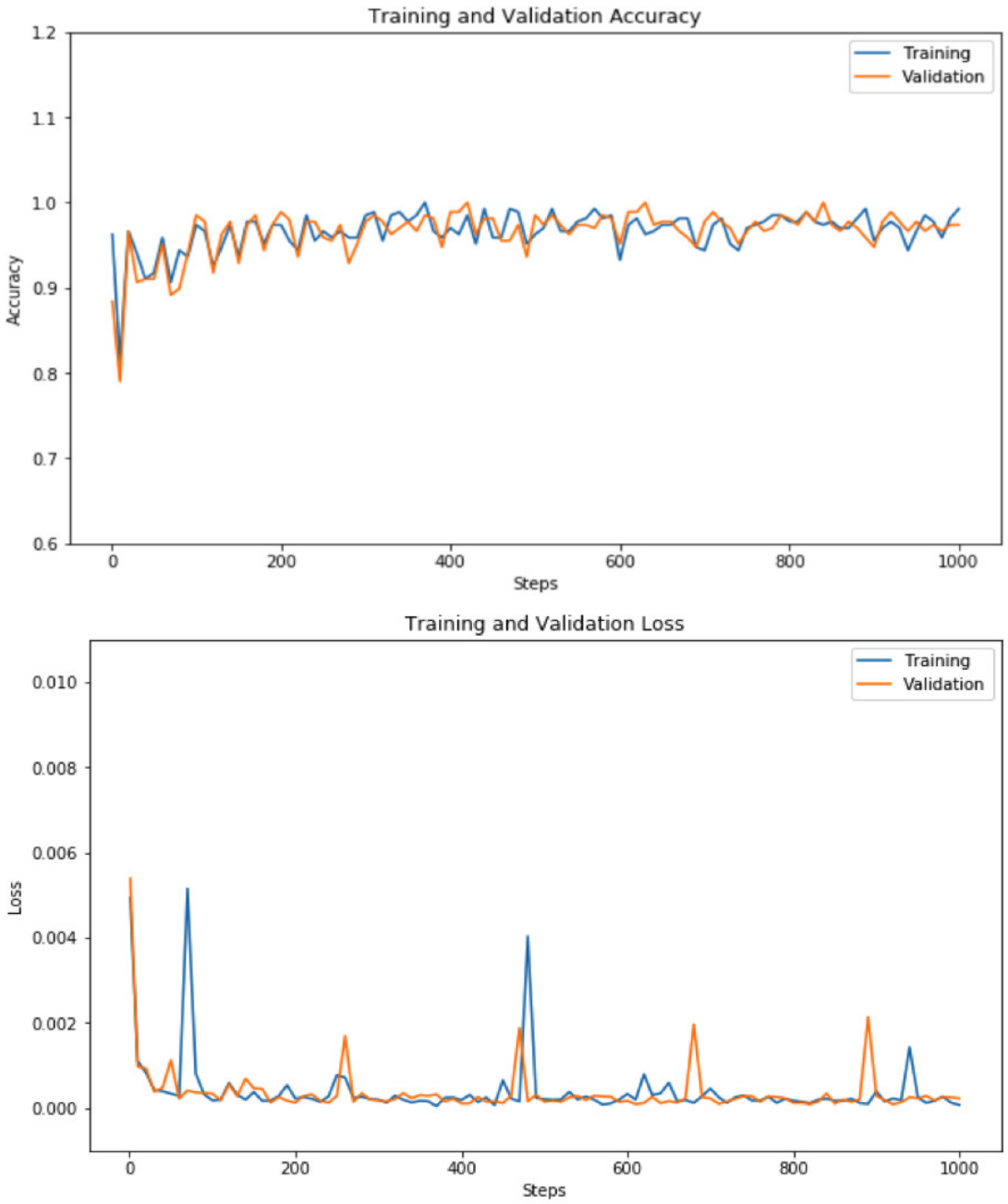


Figure 5.5. Accuracy and Loss of LSTM for training and validation datasets

The Markov model is evaluated with a set of sampling rates, i.e., 20s (i.e. second), 10s, 5s, 2s and 1s. The selection of sampling rates is arbitrary. However, a sampling rate lower than 1s would provide no significant results as the sampling rate is the lower time boundary in the dataset. Table 5.6 shows the F1 scores of the model by sampling rates. The performance

range of the model is between 92.5% and 93.5%. The highest score is achieved with a sampling rate of 5s.

	Sampling rate = 20s	Sampling rate = 10s	Sampling rate = 5s	Sampling rate = 2s	Sampling rate = 1s
F1 Score	0.9279	0.9246	0.9345	0.9312	0.9259

Table 5.6. F1 Score of a selection of sampling rates

The performances of the three models, i.e., LSTM, SVM and Markov models, are also evaluated with experiments based on a range of k-fold cross validation from 2 to 10. That is, the data is split into k subsets where each exhibits the same class distribution as the full datasets. The models are then trained on k-1 subsets and evaluated on the testing datasets. The average cross validation accuracy for each k is shown in Figure 5.6 for all the models.

The average cross validation score for LSTM increases as the number of k increases. For k equal to 8, 9 and 10, the average score is approximately the same, with 9-folds giving slightly higher score (0.9727). The four SVM models have a very stable performance across the k range without much fluctuation. Here the 9- and 10-folds have better performances than the smaller ones for SVM models. On the other hand, the Markov model exhibits more fluctuation, with the highest average score 0.94 at 4-fold with sampling rate 5s. The sampling rates 5s and 2s show better performance than the other sampling rates.

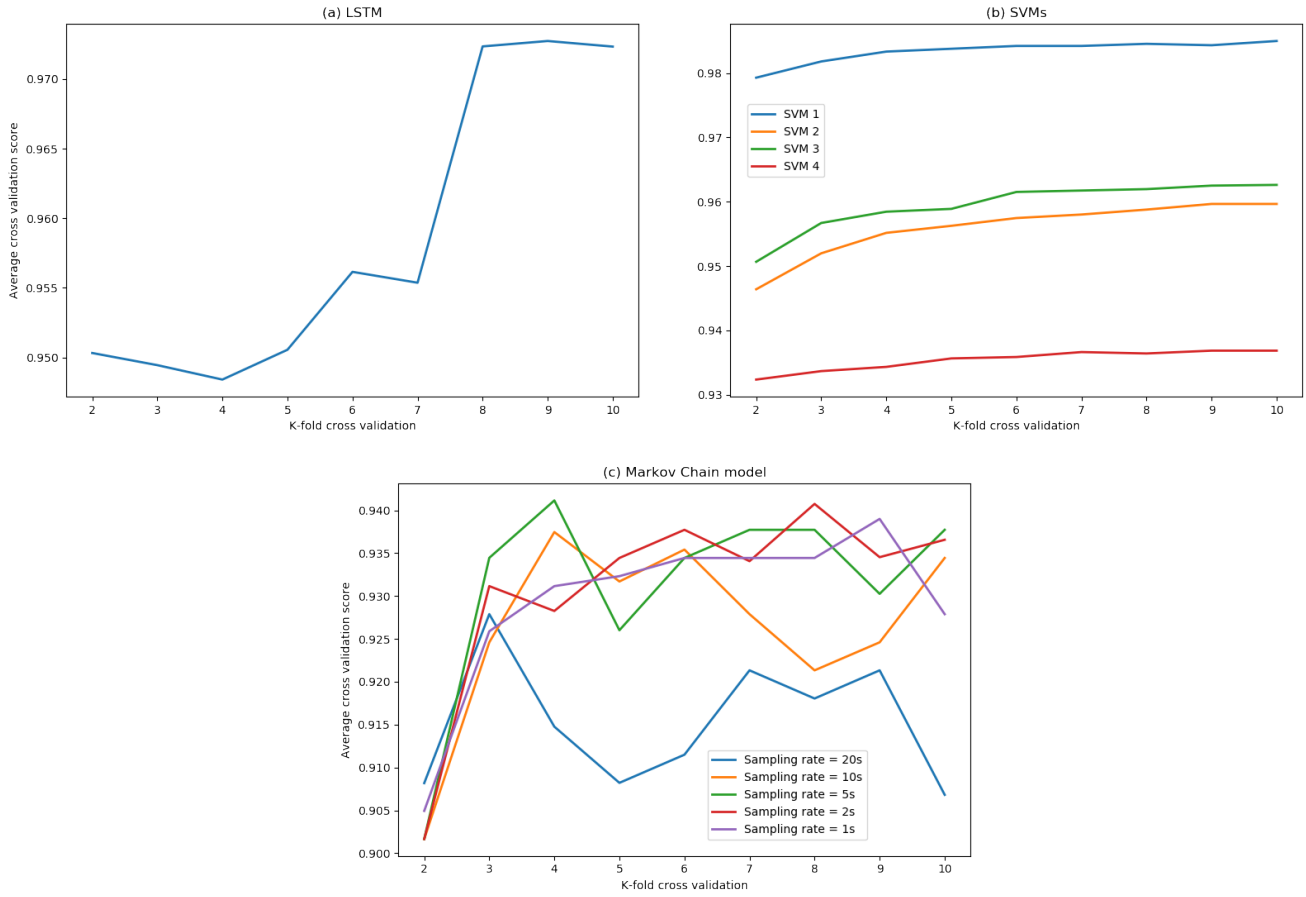


Figure 5.6. K-fold cross validation accuracy for (a) LSTM model, (b) SVM models and (c) Markov model

The most efficient model is LSTM model. It only takes the model 5 minutes on training and testing. With the same number of datasets, however, it takes the Markov model approximately 1 hour to complete the training and testing. The time varies for the SVM model series, dependent on the numbers of features used. The SVM 1 model, with the Amount feature only as the input, spends 44 seconds to achieve the same target. The SVM 2 with Amount and IP address features needs approximately 32 minutes for the training and testing, while the SVM 3 model with Amount and User agent features needs 28 minutes. The model that spends the longest time on the task is the SVM 4 model, approximately, 1 hour and 40 minutes. It is also the model that takes on the most features, including Amount, User agent and IP address.

5.2.2. Model robustness

For robustness test and reducing generalization errors, various proportions of noises are added into the datasets before training the models. The effect of mislabelled data, more specifically, the robustness of the three models in the presence of noisy class labels is evaluated.

Since the labels in the dataset are either 1s or 0s, the label noise is modelled by randomly selecting a percentage of samples and flipping their labels. In practice, the robustness of the models is evaluated with 0.1% and 1% added noises to the label. The F1 score results of the SVM, LSTM and Markov models against random label flips are shown in Table 5.7. It shows that the performance of all models drops when the noise rates increase. The SVM models and LSTM model perform better than the Markov models at 1% noise rate of the label. The Markov models' performance deteriorates with increased noise rates, showing less robustness than other models. Individually, the SVM 3 model outperforms the other three SVM models for any of the experimental percentages of noises. Between the five sampling rates in the Markov models the best performance is achieved by the model with T=5s.

Model name	**Noise rate**		
	0%	0.1%	1%
SVM 1 (Features: Amount)	0.7251	0.6620	0.7001
SVM 2 (Features: Amount, IP address)	0.9518	0.9493	0.9243
SVM 3 (Features: Amount, User agent)	0.9515	0.9556	0.9577
SVM 4 (Features: Amount, IP address, User agent)	0.9590	0.9413	0.9446
LSTM	0.9771	0.9654	0.9654
Markov Model (*T* = 20s)	0.9279	0.7661	0.5456
Markov Model (*T* = 10s)	0.9246	0.7635	0.5489
Markov Model (*T* = 5s)	0.9345	0.7688	0.5432
Markov Model (*T* = 2s)	0.9312	0.7715	0.5319
Markov Model (*T* = 1s)	0.9259	0.7661	0.5426

Table 5.7. F1 score for different percentage of noise of SVM, LSTM and Markov Model

Another way to establish the reliability or uncertainty of a model is to calculate the results' confidence intervals. Confidence intervals provide a range about model's performance and a likelihood that the performance of the model will fall between the ranges when making predictions on new data. A robust way to calculate confidence intervals for machine learning algorithms is to use the bootstrap. This is a general technique for estimating statistics that can be used to calculate confidence intervals by building a sampling distribution for a statistic by resampling the data. Suppose a random sample $S = \{X_1, X_2, \ldots, X_n\}$ is drawn from a population $P = \{x_1, x_2, \ldots, x_N\}$. Now suppose that one is interested in some statistic $\hat{\theta} = t\ (S)$ as an estimate of the corresponding population parameter $\theta = t\ (P)$ where θ is a vector of parameters and $\hat{\theta}$ is the corresponding vector of estimates. The traditional approach to find confidence intervals is to make an assumption that the population is normally distributed and along with the random sampling, to derive the sampling distribution of $\hat{\theta}$. However, the distribution of $\hat{\theta}$ can be complex hence the non-parametric bootstrap is applied as it allows estimating the sampling distribution of statistic $\hat{\theta}$ without making any assumptions about the distribution of the population. The way non-parametric bootstrap works is elegantly simple. First, a random sample of size n is drawn, with replacement, from S. It is necessary to sample with replacement otherwise the original sample S will be reproduced. Since S approximates P, the unknown distribution of $\hat{\theta}$ is approximately the distribution of the resulting bootstrap sample, based on S. That approximating distribution is used to set confidence intervals. This procedure is repeated a number of times R selecting R bootstrap samples (142).

Next, follows the construction of a confidence interval for the parameter θ. Since the bootstrap samples are treated as if it is the population, the $(100 - \alpha)\%$ bootstrap percentile interval contains the middle $(100 - \alpha)\%$ of the bootstrap distribution taking the $\alpha/2$ and $(100 - \alpha/2)$ percentile of the bootstrap distribution as interval endpoints:

$$\left[q^*_{\alpha/2},\ q^*_{100-\alpha/2}\right] \qquad \text{(eq. 5.2)}$$

However, this does not take full account of the difference between θ and $\hat{\theta}$. But assuming that the percentiles of the sampling distribution and the bootstrap distribution is approximately the same, the confidence interval is obtained by the endpoints (143):

$$\left[2\hat{\theta} - q^*_{\alpha/2},\ 2\hat{\theta} - q^*_{100-\alpha/2}\right] \quad \text{(eq. 5.3)}$$

For the 95% confidence interval for the proposed models, this means that with 95% of the chance, the true statistic falls between the 2.5th percentile of the bootstrap samples and the 97.5th percentile. By taking into account the time each model needed to run and empirical evidence (144), here R was set to 50. The statistic parameter over which the confidence intervals are estimated is the F1 score. Table 5.8 lists the 95% confidence interval of each model's F1 score, which most likely lies between the lower and upper boundaries. The F1 scores for SVM 2 model and Markov model with sampling rate equal to 20s are outside their confidence intervals though by a very small amount, indicating unreliable predications. The width of a confidence interval indicated how reliable the estimated performance of a model is. Observing the table, one sees that the LSTM model has a narrower confidence interval than the other models, indicating that the estimated F1 score (97.71%) is fairly reliable, with 95% probability of being between 97.6% and 97.9%.

	F1 Score	Confidence Interval	
SVM 1 (Features: Amount)	72.51%	69.8%	72.8%
SVM 2 (Features: Amount, IP address)	95.18%	93.7%	94.5%
SVM 3 (Features: Amount, User agent)	95.15%	95.1%	96.1%
SVM 4 (Features: Amount, IP address, User agent)	95.9%	94.5%	95.9%
LSTM	97.71%	97.6%	97.9%
Markov model (*T* = 20s)	92.79%	90.5%	92.6%
Markov model (*T* = 10s)	92.46%	91.1%	93.6%
Markov model (*T* = 5s)	93.45%	91.8%	94.1%
Markov model (*T* = 2s)	93.12%	92.5%	93.8%
Markov model (*T* = 1s)	92.59%	91.8%	93.4%

Table 5.8. 95% confidence intervals of model statistic F1 score

5.3. Conclusion

Detecting fraud is a no-easy computational task and the searching for the optimal model is complex, which can be influenced by model types, model characteristics and the data itself. In this research, the data collected for the analysis is imbalanced, with 22,880 rows of fraud datasets and 100,000 rows of legitimate datasets. The exploratory analysis is applied to both legitimate and fraudulent datasets for feature investigation and selection. Only features exhibiting the identities of fraudulent and legitimate events such as Amount, Time, IP address and user agents are selected.

Based on the integrated Payment Fraud Detection System model developed, three models, i.e., Support Vector Machine, Markov Model and LSTM model are implemented for fraud detection in Chapter 4. Subsequently, a comprehensive model evaluation is carried out in this chapter.

Various tests are carried out to evaluate the performance and robustness of the proposed models. In the tests, indicators such as F1 score, False Positive Rate (FPR), False Negative Rate

(FNR) and confidence intervals are used. Among all the SVM models the SVM 4 model achieves the best performance, based on the features including Transaction Amount, IP address and User agent, while among all the Markov type models the model with sampling rate of 5s provides the best results. Table 5.9 below compares the indicators among the best types of models developed in the modelling.

	SVM 4	Markov model (T=5s)	LSTM
F1 Score	0.9590	0.9345	0.9772
False Positive Rate (FPR)	4.67	6.55	6.78
False Negative Rate (FNR)	8.02	6.56	2.27
Number of attributes	3	4	1
Speed of learning	1h 40min	58 min	5 min
Tolerance to noise (1% Noise)	0.9446	0.9612	0.9963
95% Confidence Interval	[0.945, 0.959]	[0.918, 0.941]	[0.976, 0.979]
Data size	9120	64538	68727

Table 5.9. Comparison table of SVM, Markov Chain and LSTM

Here, the FPR measures the likelihood the model incorrectly classifies the fraud events as the legitimate events, while the FNR measures the likelihood that the framework incorrectly classifies the legitimate events as the fraud events.

From the table, one can see that the SVM 4 model has the lowest FPR, but the LSTM model is with the lowest FNR. The SVM 4 model is the most expensive model to run among all the models, whereas the LSTM model spends the least amount of time for learning. The optimal models concluded in the table are competitive and able to tolerate a small percentage of label noises in the data. The LSTM model achieves a F1 score of 97.7% whereas the SVM 4 model and Markov model achieve 93.5% and 95.0% respectively. Overall, the LSTM model performs better than the SVM and Markov models in terms of all the indicators listed here, which indicate a huge potential to be used in Payment Fraud Detection System.

5.4. Key Summary

This chapter provides comprehensive approach to evaluate and compare the performances of the models based on well-defined indicators and datasets.

- SMOTE technique is implemented in this chapter for balancing the data between the two classes before training for all the models.
- All models are validated using K-fold validation technique. The grid search technique is used to build a model on each possible parameter combinations.
- Optimal model hyper parameters are highlighted and discussed in detail for each of the model.
- To test the robustness of all the models, noise is introduced to reduce generalization errors. The effect of mislabelled data, more specifically, the robustness of the three models in the presence of noisy class labels is evaluated.
- This chapter concludes that LSTM is the most performant model as a sequence learner.

6.

Conclusion and Future Work:

6.1. Introduction of the main achieved work

This research seeks to advance the field of payment fraud detection by creating a novel framework with LSTM RNN architecture. The approach adopted was to design and implement this framework by prototyping in order to provide evidence on the feasibility of this framework. Introduction of the LSTM model demonstrated its advantage over several other traditional classifier such as SVM and Markov Model. Introduction of a unique feature engineering technique that allows the creation of temporal features, which allows LSTM to train directly on the raw data. In this chapter, this study will be concluded and will provide a closing remark on the problem statement as well as the effectiveness of proposed solution. Furthermore, recommendations and future research areas that can improve the fraud detection are discussed as well.

With the surge of internet usage, identify theft has become more efficient. Advance fraud MO as seen in this research demonstrates that fraudsters can mask themselves by faking the data points sent to the bank and defrauding the bank's security and fraud controls. Our research in Chapter 1 has also showed that simply rejecting any suspicious looking activities comes at a high operational cost. Our research also showed that traditionally online payments were limited to credit card payments only; however, there are now a number of different payment channel for customers to interact with due to innovation in technology. However, because of this there is also an increase in fraudulent activities such as social engineering and malware attacks. Fraudsters no longer have to walk into a bank's branch to carry out an attack they can simply use someone's identity or steal the information through means of hacking. The damage caused by these fraudulent activities goes beyond direct monetary loss for any financial institutions and affects their brand and market share.

As discussed in Chapter 2 many of the existing fraud detection system, such as those that are based on rule based systems and suffer limitations such as low detection rates and high false positive rates that is often caused by either a delay in a feedback loop as a result of human

investigative process involved or purely due to the fact of human error by miss-classifying the fraud labels. The limitations are further compounded by fraudsters' ability to continuously change their tactics in order to avoid detection and the machine learning models to adapt to these changing tactics in a timely manner. Research carried out in this study provided a proof of our hypothesis that retail banking transactions can indeed be modelled as sequences of time series events and generalised model can be achieved with a good model accuracy and lower false positive rates. After several empirical and analytical analysis as described in Chapter 3, Figure 6.1 below summarises different phases as the output of the framework that lead to the proposed better fraud detection model.

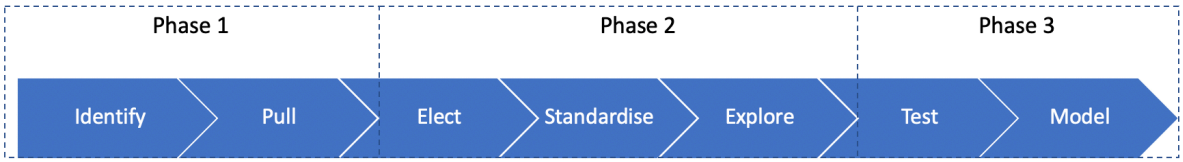


Figure 6.1. Implementation Phases

- **Phase 1** was focused on gathering dataset that is relevant to this research. This phase included:
    - Identifying dataset: identifying and labelling events belonging to victims of confirmed Social Engineering and Malware fraud for a given timeframe. The dataset contained 916 events that were labelled as being fraudulent, while the rest 3406 are considered as being legitimate.
    - Pull: retrieving and extracting all session-related information (session journey) from historical dataset.
- **Phase 2** included pre-processing, and feature extraction.
    - Elect: creating a sample by randomly selecting from a current time window to extract live transaction. The sampling rate was limited to 5 minutes so that a broad set of coverage can be gathered. For prototyping, 100,000 random samples were selected that contained completed sessions. Additional label field was added so that fraud and non-fraud dataset can be easily separated.

    - Standardise: this pre-processing step was used for organising, sorting and normalising the data so that user's journey can be easily visualised and analysed.
    - Explore: involved examining the data for the presence of indicative fraud patterns. This focused specially on social engineering and malware fraud. For each fraud account, transactions in the fraud sessions were compared with transactions of other legitimate accounts. Features were created by applying dimensionality reduction techniques.
- **Phase 3** provided details on design and implementation of testing and validation strategy, using predefined performance metrics such as accuracy, precision, recall and F1-score.
    - Test: this included dataset division strategy was defined by splitting the dataset into train, test and validate. Hypotheses were expressed in a form of a model and features and testing was done on both fraud and non-fraud accounts. An estimated performance was measured using performance metrics.
    - Model: when enough features demonstrating their abilities to sperate frauds from non-frauds to a sufficient degree were obtained, they were combined in the form of an ensembled model whose performance was estimated on the historical data.

The design and implementation defined in the above phases was used to test the performance of three different inference classifiers, namely SVM, Markov Model and LSTM using varying dataset defined by dataset division strategy, explained in detail in Chapter 5. The best performing model was LSTM with an accuracy of 97.7% when compared other classifiers. The worst performing model was Markov classifier and SVM performed relatively close to LSTM, however speed of training became a bottleneck as the data grew. The results prove that LSTM has advantages in modelling sequential data such as customer behaviours over other models used in the research. Based on the data analysis in Chapter 4, it was apparent that Fraud MOs such as social engineering is sensitive to the temporal impact and hence the performance of LSTM is not surprising since problems such as vanishing and exploding gradings weren't observed.

Overall, Table 6.1 below provides an overview of how the aim, objectives and research questions proposed in Chapter 1 are addressed.

Aim & Objectives	Status	Summary
To investigate the current state of research in fraud detection and to identify the main problems, existing approaches and available methods for achieving fraud detection with improved performance.	Met	Details can be found in Chapter 2. Existing research spans across traditional statistical rule based expert system, classical machine learning systems and deep neural network based. Not a lot of research has been done on using deep learning concepts to detect fraudulent events on retail banking. Very few papers exist on using RNN on credit card data. the main goals of this research will remain un-changed as the research showed a significant amount of work is still required to look at the financial data set for efficient fraud detection.
To investigation data availability across multiple channels to produce appropriate data sets and criteria that will form to be subject to the analysis. Also, to investigate how to process raw dataset for fraud detection.	Met	Further details can be found in Chapter 3. Generic data model described in this chapter can be used to model data from any channel. The feature engineering section describes how latent variables can be created and techniques for creating a set of features. Exploratory analysis can be found in Chapter 4.
To research into the methods for machine learning for analysing of the multidimensional banking data represented as profiles and real-time transaction data. For example, anomaly detection on the customer spends pattern and behaviour could be analysed using techniques such as neural networks.	Met	Further details can be found in Chapter 4. In this chapter an integrated Payment Fraud Detection System with three models, i.e., Support Vector Machine, Markov Model and the LSTM model are implemented based on the framework developed in the previous chapter 3.
To determine the best design ensemble and select the best ensemble classifier. To implement and execute fraud framework that ensembles deep learning and other deterministic models on a suitable platform in a context of fraud detection. Demonstrating that customer behaviour during a banking session can be used to detect social engineering and malware fraud.	Partially Met	Details on the design on design can be found in Chapter 3. Implementation is left for future work
To analyse the use of the constructed fraud model, the efficiency of the selected methods and compare the results obtained at the end of the ensemble with the results obtained from individual algorithm. Evaluate the efficiency by performing validation, offline and online testing. Validation of the developed framework will be performed	Met	A comprehensive model evaluation is carried out in chapter 5. Various tests are carried out to evaluate the performance and robustness of the proposed models. In the tests, indicators such as F1 score, False Positive Rate (FPR), False Negative Rate (FNR) and confidence intervals are used. Overall, the conclusion is that LSTM model performs better than the

using both benchmark data and real-life data; offline testing will be carried out using historical data and then online testing will be performed with real-time data.		SVM and Markov models in terms of all the indicators, which indicate a huge potential to be used in the Payment Fraud Detection System.
Research Questions		
Can events across multiple channels be considered as time-series events in order to build a classifier that can detect inherent anomaly regardless of the skewness, noise in the data?		Chapter 3 provides a conceptual framework for cross channel fraud detection system and demonstrates that classifier can be built using sequence learning technique. For more information see Section 3.3.1
How to reduce uncertainty of the model prediction by reducing false negative rate when there are only a few labelled data, or the observed labels are noisy as a result of manual measurement imprecision during labelling?		Chapter 5 provides a technique on how noise can be introduced and its impact on model performance. Section 5.2.2 highlights LSTM as the best performant model, when 1% of noise is introduced
Can a correct use of model and technique be used (i.e. sequence-based machine learning techniques such as HMM or RNN) to represent customer behaviour as they interact with the online channel?		Chapter 4 provides details on the model implementations of several sequence learners. Sequence of events can be modelled as time-series and evaluation of such models are described in Section 4.3.5
Can a model structure be able to generalise to new abnormal behaviour not seen during training? Thus, using point estimates to model deterministic function that contains uncertainty information in order to reduce false negative rate.		Generalisation of models and its effectiveness are described in Chapter 5. The model validation technique described in Section 5.2.1 is used to verify model performance and limitations, and justify the model results based on data, statistical and modelling techniques.

Table 6.1. An Overview of the aim, objectives and research questions achieved

6.2. Research Contribution

This section gives a list of contributions as outcomes to this research. The core contribution of this research is to formalise a conceptualisation of the fraud detection framework that can be generalised across different channels for banks. The heart of the concept consists of:

- Feature engineering technique that combines contextual and temporal data points such as monetary data, behavioural data, and personal data. a bi-directional LSTM inference model that will out a score between 0 to 100.
- A logical data model for cross channel fraud detection system that can cater for static, contextual, transactional and reference data from different channel, namely remote banking and cards.
- LSTM model to demonstrate that sequence learners are useful for learning patterns of unknown length such as our social engineering use case, as they have ability to operate over sparse dataset that have higher-level temporal features such as our banking dataset.

- Synthetic data generation technique: Another major contribution is providing a novel approach of generating synthetic that can generate data quickly and efficiently. The quality of generated data is a close representation of the real data for fraud detection domain. This novel approach makes use of GAN network to generate a better dataset that can be used to augment original dataset for training and evaluating. As seen in Chapter 3, this output of the GAN closes matches the real dataset while increasing the number of samples.
- Model evaluation technique to verify model performance and limitations, and justify the model results based on data, statistical and modelling techniques. Core focus is based on utilising the F1 score to measure the effectiveness.

6.3. Recommendations for future research

Avenues for future work are many and varied. The conceptualisation of the fraud framework that is proposed in this thesis can be used for various different data points across the various bank's channel. However, the overall fraud management tasks are challenging, it would be interesting to model fraud management using NIST's cyber security framework [145] across identity, protect, detect, respond and recover functions.

Furthermore, in relation to fraud detection itself one could further explore the domain by:
Increasing data set: in order to compensate for lack of amount of data, synthetic data technique was used, however to in order to truly evaluate the model performance, real data is required over a longer period for training, testing and evaluating. Also, more data means more complex model can be evaluated such as Convolution Recurrent Neural Network and more complex latent features can be extracted by convolution kernels before applying to recurrent neural networks.

Implementing ensemble technique: a technique to create ensemble classifier was discussed in Chapter 3, however due to time limitation the implementation was incomplete. This technique is focused on using a weak classifier. Using such weak classification technique overall framework's performance can be improved as weak algorithm such as RFM and Benford's law for fraud detection can be easily plugged in.

Adding additional dataset such as credit card (card present and card not present data): Conceptualisation of the fraud framework described in Chapter 3 is a generic framework that can generalised over different cross channels for a bank. However, due to time limitation only the retail banking channel data were analysed.

References

1. On the draft Regulatory Technical Standards specifying the requirements on strong customer authentication and common. **European Banking Authority.** Augest 2016. [Cited: 12 September2016.] https://www.eba.europa.eu/documents/10180/1548183/Consultation+Paper+on+draft+RTS+on+SCA+and+CSC+(EBA-CP-2016-11).pdf. EBA-CP-2016-11.

2. *January to June 2016 fraud update: Payment cards, remote banking and cheque.* **Financial Fraud Action UK.** s.l. : Financial Fraud Action UK, 2016.

3. FICO Enterprise Fraud Management. **Falcon Fraud Manager, FICO.** [Online] 2016. [Cited: 21 December 2016.] http://www.fico.com/en/latest-thinking/white-papers/fico-enterprise-fraud-brochure.

4. An Enterprise Approach to Fighting Online Fraud. **SAS Fraud Management, SAS Institute.** [Online] [Cited: 21 December 2016.] http://www.sas.com/en_us/whitepapers/fighting-online-fraud-107854.html.

5. Fraud Monitoring and Mitigation Strategies by Channel. **ACI World Wide.** [Online] 2014. [Cited: 15 July 2016.] http://www.aciworldwide.com/what-we-know/insights/-/media/58c9fae2dcd544dabcd6e5610d40d2a5.ashx.

6. *Fraud detection in online reviews using machine learning techniques.* **Shivagangadhar, K., Sagar H, Sathyan, S. & Vanipriya, C. H.** 5, May 2015, International Journal of Computational Engineering Research (IJCER), Vol. 5, pp. 52-56.

7. *Fraud Analytics Using Descriptive, Predictive and Social Network Techniques.* **Bart Baesens, Veronique Van Vlasselaer, Wouter Verbeke.** s.l. : Wiley Book, 2015.

8. *Detecting Fraudulent Behaviour Using Recurrent Neural Networks.* **Yoshihiro Ando, Hidehito Gomi, Hidehiko Tanaka.** s.l. : Computer Security Symposium 2016, Octomber 2016.

9. Fraud Prevention and Detection for Credit and Debit Card Transactions. **WebSphere Software IBM.** [Online] August 2009. [Cited: 02 Feburary 2016.] ftp://public.dhe.ibm.com/software/cn/websphere/industries/government/WSW14053USEN.pdf.

10. *Neural Network Predictor for Fraud Detection: A Study Case for the Federal Patrimony Department.* **Antonio Manuel Rubio Serrano, João Paulo Carvalho Lustosa da Costa, Carlos Henrique Cardonha, Ararigleno Almeida Fernandes and Rafael Timóteo de Sousa Júnior.** s.l. : THE INTERNATIONAL CONFERENCE ON FORENSIC COMPUTER SCIENCE (ICOFCS), 2012.

11. *Credit Card Fraud Detection Using Neural Network.* **Raghavendra Patidar, Lokesh Sharma.** NCAT2011, June 2011, International Journal of Soft Computing and Engineering (IJSCE), Vol. 1, pp. 2231-2307.

12. Fraud Nelson Report. **David Robertson.** - *https://www.nilsonreport.com/upload/content_promo/The_Nilson_Report_10-17-2016.pdf.* s.l. : The Nelson Report, 2017.

13. A-Z of fraud. *Action Fraud UK.* **Action Fraud UK.** [Online] [Cited: 24 12 2016.] http://www.actionfraud.police.uk/a-z_of_fraud.

14. *Statistical Fraud Detection: A Review.* **Hand, Richard J. Bolton and David J.** 3, 2002, Statistical Science, Vol. 17, pp. 235–255.

15. *Other People's Money: Study in the Social Psychology of Embezzlement.* **Cressey, Donald R.** April 1972.

16. *A Conceptual Model For Proactive Detection Of Potential Fraud In Enterprise Systems: Exploiting SAP Audit Trails To Detect Asset Misappropriation.* **Singh, K Harichunder.** s.l. : University of Southern Queensland, 2012.

17. *The Physiology of Fraud. In terms and issues in Crime an criminal Justice.* **P, Duffield G and Grabosky.** s.l. : Australian Institute of Criminology, 2001.

18. *The Definitive Overview of Payment Industry Fraud, Fraud the Facts 2016.* **Financial Fraud Action UK.** s.l. : Financial Fraud Action UK, 2016.

19. *Unsupervised Profiling Methods for Fraud Detection.* **J., Richard Richard J. Bolton and David.** s.l. : Hand Department of Mathematics Imperial College London, 2001.

20. *APATE: A Novel Approach for Automated Credit Card Transaction Fraud Detection using NetworkBased Extensions.* **V. Van Vlasselaer, C. Bravo, O. Caelen, T. Eliassi-Rad, L. Akoglu, M. Snoeck, and B. Baesens.** 2015, Decision Support Systems, Vol. 75, pp. 38-48.

21. *The development of a rule based expert system model for fraud alert in consumer credit.* **K., Leonard.** 1995, European Journal of Operational Research, Vol. 80, pp. 350-356.

22. *Fuzzy Darwinian Detection of Credit Card Fraud.* **Peter J. Bentley, Jungwon Kim, Gil-Ho Jung and Jong-Uk Choi.** s.l. : Department of Computer Science, University College London; Department of Computer Science, SungKyunKwan University and Sangmyung University, 2000.

23. *Fraud Detection in Telecommunications: History and Lessons Learned.* **Richard A. BECKER, Chris VOLINSKY, and Allan R. WILKS.** 1, Feb 2010, American Statistical Association and the American Society for Quality TECHNOMETRICS, Vol. 52.

24. *Pattern Recognition and Machine Learning.* **Bishop, Christopher M.** s.l. : Sprinter, 2007. Vol. International Edition.

25. *Introduction to Information Retrieval. Chapter 17.* **Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze.** s.l. : Cambridge University Press, 2008, pp. 377 - 385.

26. *Survey of Clustering based Financial Fraud Detection Research.* **Sabau, Andrei Sorin.** 1, 2012, Informatica Economica, Vol. 16.

27. *Utility-Based Fraud Detection.* **Lopes, Luis Torgo and Elsa.** s.l. : Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence, 2010.

28. Density based clustering and radial basis function modeling to generate credit card fraud score. **V.Hanagandi, A. Dhar, K.Buescher.** *IEEE International Conference.* Februray 1996, pp. 247-251.

29. *Credit Card Number Fraud Detection Using K-Means with Hidden Markov Method.* **Pooja Bhati, Manoj Sharma2.** 3, June 2015, SSRG International Journal of Mobile Computing & Application (SSRG-IJMCA), Vol. 2.

30. *Some Methods for classification and Analysis of Multivariate Observations.* **MacQueen, J. B.** s.l. : Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability, Berkeley, University of California Press, 1967.

31. *Genetic K-means Algorithm for Credit Card Fraud Detection.* **Pooja Chougule, A.D. Thakare, Prajakta Kale, Madhura Gole, Priyanka Nanekar.** 2, 2015, International Journal of Computer Science and Information Technologies, Vol. 6.

32. *Transaction Fraud Detection Based on Total Order Relation and Behavior Diversity*. **L. Zheng, G. Liu, C. Yan, and C. Jiang**. IEEE Trans. Comput. Soc. Sy*st.*, Vol. 5, no. 3, pp. 796–806, Sep. 2018

33. *An Adaptive and Real-Time Fraud Detection Algorithm in Online Transactions*. **J. Batani**. Int. J. Comput. Sci. Bus. Informatics, Vol. 17, no. 2, pp. 1–12, 2017

34. Fraud Detection Using Predictive Modelling **Gopinathan, Krishna M. and Louis S., Biafore, William M. Ferguson, Michael A. Lazarus, Anu K. Pathria, Allen Jost.** *PCT/US1993/008400* US, Oct 1998.

35. *Neural Networks and Learning Machines.* **Haykin, Simon.** s.l. : Pearson, 2009.

36. *Neural Networks - A systemitic introduction.* **Rojas, Raul.** s.l. : Springer, 1996. pp. 392-412.

37. *Unsupervised clustering with growing self-organizing neural network – a comparison.* **Martin Hynar, Michal Burda, and Jana Sarmanov.** s.l. : CEUR Workshop Proceedings, 2005.

38. *CREDIT CARD FRAUD DETECTION USING SELF- ORGANIZING MAPS.* **STRIZHAK, Vladimir ZASLAVSKY and Anna.** 2013.

39. *Fraud detection using self-organizing map visualizing the user profiles.* **Olszewski, Dominik.** 2014, Knowledge-Based Systems, Vol. 70, pp. 324-334.

40. *Behaviour mining for fraud detection.* **Xu, Jianyun, Sung, Andrew H.Liu, Qingzhong.** 2007, Journal of Research and Practice in Information Technology.

41. *Probabilistic Approaches to Fraud Detection.* **Hollmén, Jaakko.** s.l. : Helsinki University of Technology Department of Computer Science and Engineering, 1999.

42. *Implementing social network analysis for fraud prevention.* **CGI Group Inc.** [Online] [Cited: 03 January 2017.] https://www.cgi.com/sites/default/files/white-papers/Implementing-social-network-analysis-for-fraud-prevention.pdf.

43. Entity Centric and Social Network Analysis for an effective Financial Crime strategy. **Detica Inc.** [Online] [Cited: 01 January 2017.] http://ari.uitm.edu.my/main/images/download/icftcf2011/day1-12.pdf.

44. *Neural Network Fraud Detection in Credit Card Operations.* **Dorronsoro, J., Ginel, F., Sanchez, C. & Cruz, C.** 4, 1997, IEEE Transactions on Neural Networks, Vol. 8.

45. *A Survey on Hidden Markov Model for Credit Card Fraud Detection.* **Anshul Singh, Devesh Narayan.** 3, February 2012, International Journal of Engineering and Advanced Technology (IJEAT), Vol. 1.

46. *Adaptive Fraud Detection Using Benford's Law.* **Fletcher Lu, J. Efrim Boritz, and Dominic Covvey.** 2006, Advances in Artificial Intelligence.

47. *Financial fraud detection: a new ensemble learning approach for imbalanced data.* **Bian, Yiyang; Cheng, Min; Yang, Chen; Yuan, Yuan; Li, Qing; Zhao, J. Leon; and Liang, Liang** (2016). *PACIS 2016 Proceedings.* 315

48. *Fraud Detection by Monitoring Customer Behaviour and Activities.* **Parvinder Singh, Mandeep Singh.** 11, February 2015, International Journal of Computer Applications, Vol. 111.

49. *Fraud Detection in Credit Card by Clustering Approach.* **Vaishali.** 3, July 2014, International Journal of Computer Applications, Vol. 98.

50. *A comparative study of efficient initialization methods for the K-Means clustering algorithm.* **Celebi, Kingravi and Vela.** 1, 2013, Expert systems with applications, Vol. 40, pp. 200-210.

51. *Applications of partition based clustering algorithms: A survey.* **A. Dharmarajan, T. Velmurugan.** s.l. : IEEE International Conference on Computational Intelligence and Computing Research, 2013.

52. *A Survey on Statistical Relational Learning.* **Bina, Hassan Khosravi and Bahareh.** June 2010, Canadian Conference on Artificial Intelligence.

53. *Credit Card Fraud Detection Using Advanced Combination Heuristic and Bayes' Theorem.* **Sahil Hak, Suraj Singh, Varun Purohit.** April 2015, IEEE.

54. *Credit Card Fraud Detection via Kernel-Based Supervised Hashing*. **Li Z, Liu G, Wang S, Xuan S, Jiang C**. IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation*),* 2018, pp. 1249–1254

55. *Detecting credit card fraud by genetic algorithm and scatter search.* **Duman and M. H. Ozceli**k. Expert Syst. Appl., vol. 38, no. 10, pp. 13057–13063, Sep. 2011.

56. *Anomaly Detection with Machine Learning and Graph Databases in Fraud Management.* **S. Magomedov, S. Pavelyev, I. Ivanova, A. Dobrotvorsky, M. Khrestina, and T. Yusubaliev**. 2018.

57. *Enterprise Financial Crimes Framework for Banking Prevent, detect and manage fraud across the enterprise.* **SAS.** s.l. : SAS INC., 2011.

58. *A Comprehensive Survey of Fraud Detection Techniques.* **Lutfun Nahar Lata, Israt Amir Koushika and Syeda Shabnam Hasan.** December 2015, International Journal of Applied Information Systems (IJAIS), Vol. 1.

59. *A logical calculus of the ideas immanent in nervous activity.* **Pitts, Warren S. McMuclloch and Walter H.** 1943, Bulletin of Mathematical Biophysics, Vol. 5, pp. 115-133.

60. *Perceptrons - An Introduction to Computational Geometry.* Expanded Edition. **Papert, Marvin Minsky and Seymour A.** s.l. : MIT Press, 1987.

61. *Deep Sparse Rectifier Neural Network.* **Xavier Glorot, Antoine Bordes, Yoshua Bengio.** 2011, International Conference on Artificial Intelligence and Statistics (AISTATS), Vol. 15.

62. *Rectified Linear Units Improve Restricted Boltzmann Machines.* **Vinod Nair, Geoffrey E. Hinton.** 2010, International Conference on Machine Learning.

63. *Rectifier Nonlinearities Improve Neural Network Acoustic Models.* **Andrew L. Maas, Awni Y. Hannun, Andrew Y. Ng.** 2013, International Conference on Machine Learning.

64 . Evaluating Consumer Loans Using Neural Networks Ensembles. **Maher Alaraj, Maysam Abbod, and Ziad Hunaiti.** *International Conference on Machine Learning, Electrical and Mechanical Engineering.* 08 01 2014.

65 . An Overview of Personal Credit Scoring: Techniques and Future Work. *International Journal of Intelligence Science.* **Li, X.L., & Zhong, Y.** 2012, pp. 181-189.

66. *Credit Card Fraud Detection using Artifical Neural Networks Tuned by Genetic Algorithms.* **Paasch, Carsten A. W.** 2011.

67. *Credit card fraud detection with a neural network.* **S. Ghosh, D. L. Reilly, and N. Inc.** 1994, Proceedings of the Twenty-Seventh Hawaii International Conference on, Vol. 3.

68. *Cluster Analysis and Artificial Neural Networks A Case Study in Credit Card Fraud Detection.* **Emanuel Mineda Carneiro, Luiz Alberto Vieira Dias, Lineu Fernado Stege Mialaret.** April 2015, IEEE.

69. *Novel Artificial Neural Networks and Logistic Approach for Detecting Credit Card Deceit.* **Ganesh Kumar Nune, P. Vasanth Sena and T.P. Shekhar.** 3, Octobor 2012, International Journal of Computer Science and Management Research, Vol. 1.

70. *Parallel granular neural networks for fast credit card fraud detection.* **M. Syeda, Y.Q. Zhang, Y. Pan.** 2002, Proceedings of the IEEE International Conference on Fuzzy Systems, pp. 572-577.

71. *Credit card fraud detection using Bayesian and neural networks.* **S. Maes, K. Tuyls, B. Vanschoenwinkel, B. Manderick.** 2002, Proceedings of the First International NAISO Congress on Neuro Fuzzy Technologies.

72. *Real-Time Credit-Card Fraud Detection using Artificial Neural Network Tuned by Simulated Annealing Algorithm.* **Azeem Ush Shan Khan, Nadeem Akhtar and Mohammad Naved Qureshi.** s.l. : Recent Trends in Information, Telecommunication and Computing, ITC, Association of Computer Electronics and Electrical Engineers, 2014.

73 *Supervised Machine (SVM) Learning for Credit Card Fraud Detection.* **Sitaram Patel, Sunita Gond.** 3, Feb 2014, International Journal of Engineering Trends and Technology (IJETT) , Vol. 8.

74. *Fraud Detection Using Reputation Features, SVMs, and Random Forests.* **Dave DeBarr, Harry Wechsler.** s.l. : World Congress in Computer Science, Computer Engineering, 2013.

75. Detecting Credit Card Fraud by Decision Trees. **Duman, Y. Sahin and E.** *Internatinal MultiConference of Engineers And Comput Scieentists.* 2011.

76. *An Overview of Statistical Learning Theory.* **Vapnik, Vladimir N.** 5, 1999, IEEE TRANSACTIONS ON NEURAL NETWORKS, Vol. 10.

77. *An Intrdouction to Support Vector Machines and Other Kernal Based Learning Methods.* **Cristianini, N., Taylor J. S.** s.l. : Cambridge University Press, 2000.

78. *A brief introduction to kernel classifiers.* **Johnson, Mark.** Lecture Notes. s.l. : Brown University - Computer Science Department, 2009.

79. Personalized approach based on SVM and ANN for detecting credit card fraud. **R.C. Chen, S.T. Luo, X. Liang, V.C.S. Lee.** *Proceedings of the IEEE International Conference on Neural Networks and Brain.* pp. 810-815.

80. *Data Mining With Decision Trees: Theory And Applications.* **Maimon, Lior Rokach and Oded Z.** 2nd. s.l. : World Scentific Publishing Co, 2014.

81. *Credit Card Fraud Detection Using Decision Tree for Tracing Email and IP.* **GAYATHIRI.P, R. DHANAPAL.** 5, 2012, IJCSI International Journal of Computer Science, Vol. 9.

82. *Classification and Regression Trees.* **Breiman, L., Friedman, J. H., Olshen, R. A., Stone, C. J.** s.l. : Wadsworth & Brooks, 1984.

83. *A Hybrid Approach of Stepwise Regression, Logistic Regression, Support Vector Machine, and Decision Tree for Forecasting Fraudulent Financial Statements.* **Suduan Chen, Yeong-Jia James Goo, and Zone-De Shen.** 2014, The Scientific World Journal, Vol. 2014.

84. *Credit Card Fraud Detection Using Decision Tree Induction Algorithm.* **Snehal Patil, Harshada Somavanshi, Jyoti Gaikwad, Amruta Deshmane, Rinku Badgujar.** 4, 2015, International Journal of Computer Science and Mobile Computing (IJCSMC) , Vol. 4, pp. 92-95.

85. *A fast learning algorithm deep belief nets.* **Hinton, G., Osindero, S., and Naïve, Y.** 2006, Neural Computation, Vol. 18, pp. 1527-1554.

86. *Deep Learning of Representations for Unsupervised and Transfer Learning.* **Bengio, Yoshua.** s.l. : JMLR: Workshop and Conference Proceedings, 2012.

87. *Representation learning: A review and new perspectives.* **Bengio Y., Courville, A., and Vincent, P.** 2013, IEEE.

88. *Deep Learning NIPS'2015 Tutorial.* **Geoff Hinton, Yoshua Bengio & Yann LeCun.** s.l. : CIFAR, 2015.

89. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition.* **David E. Rumelhart, James L. McClelland.** s.l. : MIT Press, 1986.

90. *A Deep Non-Linear Feature Mapping for Large-Margin kNN Classification.***Renqiang Min, David A. Stanley.** s.l. : University of Toronto, 2009.

91. *Deep Learning.* **Ian Goodfellow, Yoshua Bengio and Aaron Courville.** MIT Press, 2016.

92. *Network anomaly detection with the restricted Boltzmann machine.* **Ugo Fiore, Francesco Palmier, Aniello Castiglione, and Alfredo De Santis.** October 2013, Elseviee.

93. *Gradient-Based Learning Applied to Document Recognition.* **Yann Lecun, Leon Bottou, Yoshua Bengio and Patrick Haffner.** 1998, IEEE.

94. Credit Card Fraud Detection Using Convolutional Neural Networks. **Kang Fu, Dawei Cheng, Yi Tu, and Liqing Zhang.** *International Conference on Neural Information Processing .* 26 September 2016, pp. 483-490.

95. Training Recurrent Neural Network. **Sutskever, Ilya.** s.l. : University of Toronto, 2013.

96. *Credit Card Transactions, Fraud Detection, and Machine Learning: Modelling Time with LSTM Recurrent Neural Networks.* **Wiese, Benard Jacobus.** s.l. : Department of Computer Science, University of the Western Cape, 2007.

97. *Graph analytics for healthcare fraud risk estimation.* **L. K. Branting, F. Reeder, J. Gold, and T. Champney.** IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), 2016, pp. 845–851

98. *Discovering important nodes through graph entropy the case of Enron email database*. **J. Shetty and J. Adibi**. Proceedings of the 3rd international workshop on Link discovery - LinkKDD, 2005, pp. 74–81

99. *A comparative survey of artificial intelligence applications in finance: artificial neural networks, expert system and hybrid intelligent systems.* **Bahrammirzaee, Arash.** 8, November 2010, Neural Computing and Applications, Vol. 19, pp. 1165 - 1195.

100. *Data mining techniques and its applications in banking sector.* **Chitra, K. & Subashini, B.** 8, Augest 2013, International Journal of Emerging Technology and Advanced Engineering, Vol. 3, pp. 219-226.

101. *A Review of Financial Accounting Fraud Detection based on Data Mining Techniques.* **Anuj Sharma, Prabin Kumar Panigrahi.** Feburary 2012, International Journal of Computer Applications.

102. *Big Data and Specific Analysis Methods for Insurance Fraud Detection.* **Ana-Ramona BOLOGA, Razvan BOLOGA, Alexandra FLOREA.** 1, 2010, Database Systems Journal, Vol. 1.

103. *Large scale distributed deep networks.* **J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, Q. Le, M. Mao, M. Ranzato, A. Senior, P. Tucker, K. Yang, and A. Ng.** 2012, In Proceedings of Neural Information Processing Systems (NIPS).

104. *How PayPal uses deep learning and detective work to fight fraud.* **Harris, Derrick.** s.l. : Gigaom, March 2015.

105. *SMOTE: Synthetic Minority Over-sampling Technique.* **Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, W. Philip Kegelmeyer.** 2002, Journal of Artificial Intelligence Research, Vol. 16, pp. 321–357.

106. A conceptual basis for feature engineering. **C. Reid Turner, Alfonso Fuggetta, Luigi Lavazza, Alexander L. Wolf.** *The Journal of Systems and Software.* 49, 1999, Vol. 3, 15.

107. *Picking Them by Their Batting Averages' Recency-Frequency-Monetary Method of Controlling Circulation.* **Cullinan, G.J.** s.l. : New York Direct Mail/Marking Association, 2013.

108. A Study and Comparative Analysis of Conditional Random Fields For Intrusion. **Deepa Guleria, M.K.Chavan.** 4, 2012, International Journal of Research in Computer Science elSSN , Vol. 2, pp. 31-38.

109. *Finding Rare Classes: Adapting Generative and Discriminative Models in Active Learning.* **Timothy Hospedales, Shaogang Gong and Tao Xiang.** 2, s.l. : IEEE Transactions on Knowledge and Data Engineering, 2013, Vol. 25.

110. *Effective detection of sophisticated online banking fraud on extremely imbalanced data.* **Wei Wei, Jinjiu Li, Longbing Cao, Yuming Ou, Jiahang Chen.** 4 pp. 449-475, s.l. : Springer Inc - World Wide Web, 2013, Vol. 16.

111. **L***earning, Pattern Recognition and Machine.* **Christopher Bishop.** s.l. : Springer Science + Business Media, 2007.

112. PCA Whitening. *Unsupervised Feature Learning and Deep Learning.* **Deep Learning, Computer Science Department.** [Online] Standford University. [Cited: 31 01 2017.] http://ufldl.stanford.edu/tutorial/unsupervised/PCAWhitening/.

113. *Batch Normalized Recurrent Neural Network.* **Cesar Laurent, Gabriel Pereyra, Philemon Brakel, Ying Zhang, Yoshua Bengio.** s.l. : UIniversity of Montreal, 2015.

114. *Adaptive Machine Learning for Credit Card Fraud Detection***. Andrea Dal Pozzolo, Prof. Gianluca Bontempi.** s.l. : University of Brusells , 2015.

115. Transaction aggregation as a strategy for credit card fraud detection. **C. Whitrow, D. J. Hand, P. Juszczak, D. Weston, N. M. Adams.** *Data Mining and Knowledge Discovery.* Feb 2009, Vol. 18, 30-55.

116. *Learning Deep Generative.* **Salakhutdinov, Ruslan.** s.l. : Annual Review of Statistics and Its Application, 2015, Vol. 2. 361-385.

117. L*ecture Notes: CSC321: Intro to Machine Learning and Neural Networks.* **Guerzhoy, Michael.** s.l. : University of Torronto, 2016.

118. Feature selection, L1 vs. L2 regularization, and rotational invariance. **Ng, Andrew Y.** *International Conference on Machine Learning.* 2004, Vol. 21.

119. *Supervised Sequence Labelling.* **Graves, Alex.** s.l. : Springer, 2012.

120. *Recurrent Neural Networks. Lecture 12.* **Bullinaria, John A.** s.l. : University of Birmingham, 2015. Vol. Neural Computation

121. *Recurrent Neural Network for Prediction.* **Danilo P Mandic, Jonathon A Chambers.** s.l. : John Wiley & Sons LTD, 2001.

122. *Recurrent Neural Networks for Prediction: Learning Algorithms, Architectures and Stability.* s.l. : **Wiley**, 2001.

123. The Vanashing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions. **Hochreiter, Sepp.** *International Journel jof Uncertainity, Fuzziness and Knowledge Based Systems.*

124. *Long Short-Term Memory. Technical Report FKI-207-95.* **Hochreiter, S. & Schmidhuber, J.** 1996, Vol. 3.0.

125. *Performance criteria for plastic card fraud detection tools.* **D. Hand, C. Whitrow, N. M. Adams, P. Juszczak, and D. Weston.** 7, s.l. : Journal of the Operational Research Society, May 2007, Vol. 59.

126. *A Review of Synthetic Data Generation Methods for Privacy Preserving Data Publishing*. **Surendra H, Mohan S.** International Journal of Scientific & Technology Research. Vol: 6. Issue: 03. March 2017.

127. *ADASYN: Adaptive Synthetic Sampling Approach for Imbalanced Learning*. **Haibo H, Yang B, Edwardo G, and Shutao L**. IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence). 2008.

128. *ADASYN: Kernel Based Adaptive Synthetic Data Generation for Imbalanced Learnin*g. **Bo T, and Haibo H. Kernel.** IEEE Congress on Evolutionary Computation (CEC). 2015.

129. *A Synthetic Fraud Data Generation Methodology.* **Emilie L, Håkan K, Erland J**. International Conference on Information and Communications Security. 2002.

130. *Synthesizing test data for fraud detection systems*. **Emilie L, Håkan K, Erland J**. 19th Annual Computer Security Applications Conference. 2003.

131. L*earning from simulated and unsupervised images through adversarial training*. **Shrivastava A, Pfister T, Tuzel O, Susskind J, Wang W, Webb R.** Computer Vision and Pattern Recognition. 2017.

132. *Synthetic Data Augmentation Using GAN for Improved Liver Lesion*. **Maayan F, Eyal K, Michal A, Jacob G, Hayit G**, IEEE International Symposium on Biomedical Imaging. 2018

133. *Generative adversarial nets.* **Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y.** *In: Advances in neural information processing systems*. pp. 2672–80 (2014)

134. Improving Credit Card Fraud Detection with Calibrated Probabilities. **Bahnsen, A and Stojanovic, A and Djamila Aouada, D and Ottersten, B.** Journal SIAM

135 *Overview of Encoding Methodologies (article) - DataCamp*. 2018. **N. Singh**. [Online]. Available: https://www.datacamp.com/community/tutorials/encoding-methodologies. [Accessed: 24-Jun-2019].

136. *An investigation of categorical variable encoding techniques in machine learning: binary versus one-hot and feature hashing*. **C. Seger**. 2018.

137. An Introduction to Support Vector Machines and Other Kernal Based Learning Methods. **Cristianini, N., Taylor J. S**. s.l. : Cambridge University Press, 2000.

138. *Information Theory, Inference, and Learning Algorithms.* **Mackey, D**.

139. *Understanding LSTMs*. [Cited: 06/07/2017] http://colah.github.io/posts/2015-08-Understanding-LSTMs/

140. *Introduction to Support Vector Machines (SVM)*. [Cited: 03/08/2017] http://docs.opencv.org/2.4/doc/tutorials/ml/introduction_to_svm/introduction_to_svm.html

141. *Grid Searching in Machine Learning: Quick Explanation and Python Implementation*. **E. Lutins**, 2017. [Online]. Available: https://medium.com/@elutins/grid-searching-in-machine-learning-quick-explanation-and-python-implementation-550552200596. [Accessed: 04-Jul-2019].

142. *Bootstrapping Regression Models*. **Fox J.** 2002, [Online]. Available: http://statweb.stanford.edu/~tibs/sta305files/FoxOnBootingRegInR.pdf. [Accessed July 22, 2019].

143. *ON TEACHING BOOTSTRAP CONFIDENCE INTERVALS.***Engel J**. 2010, [Online]. Available: www.stat.auckland.ac.nz/~iase/. [Accessed July 22, 2019].

144. *Bootstrap Confidence Intervals: When, Which, What? A Practical Guide for Medical Statisticians*. **Carpenter J, Bithell J**. Vol 19, 2000. [Online]. Available: https://m.tau.ac.il/~saharon/Boot/10.1.1.133.8405.pdf. [Accessed July 23, 2019].

145. Natioal Institute of Standards and technology (NIST) Cyber Security Framework .[Online]. Available: https://www.nist.gov/cyberframework. [Accessed February 06, 2020]

Appendix - A

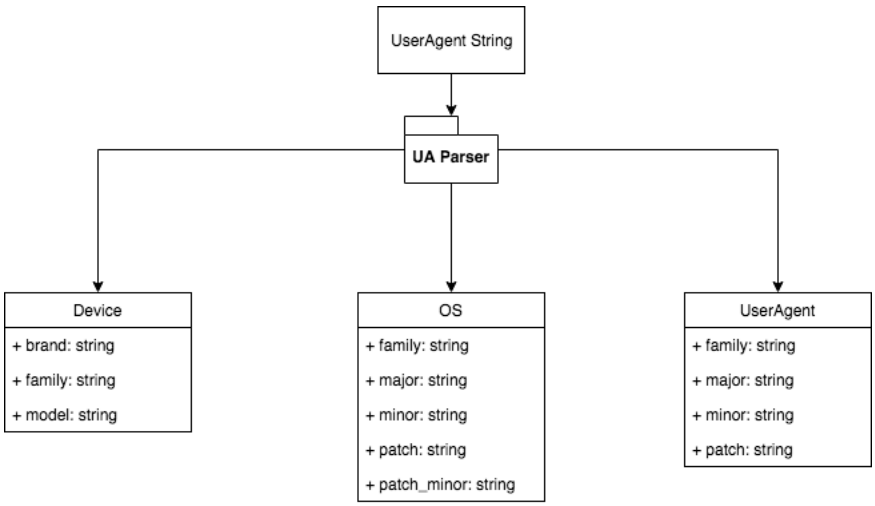
IP address-UA model

During the exploratory analysis it was discovered that a probability of a fraudulent transaction is high when there is a password reset event and then immediately a login event. Therefore, all the event types that have "PasswordReset" followed by "CustomerLogin" were extracted from the dataset. In total, there are 110 customers that follow this sequence where 77 of the customers were flagged as fraud and the remaining 40 as genuine. (There were 220 fraudulent types and 80 genuine event types.) Additional attributes such as IP Address, User Agent and latency and others are deemed good indicators for fraud identification and prevention, hence they are used for this model. Table 1 shows an example of how the data looks like.

Customer ID	Event Type	Session ID	Latency	IP address	User agent	Flag
276506	PasswordReset	…	345	100.35.356.35	…	1
276506	CustomerLogin	…	135	100.35.356.35	…	1
932765	PasswordReset	…	235	123.45.57.68	…	0
932765	CustomerLogin	…	109	123.45.57.68	…	0
…	…	…	…	…	…	…

Sample of raw data based on Password Reset - Customer Login sequence

Before fitting the data into the model, a pre-processing stage occurs to some of the features in order to be transformed into a suitable format for the model. At first user-agent string was analysed using user agent parser library. Once the user agent string is passed through the parser, it provides the information shown in figure below.



Information of user agent string passed through a user agent parser

An example of the parsed string for fraud and non-fraud label is shown below:

Non-Fraud
{ 'device': {'brand': None, 'family': 'Other', 'model': None}, 'os': { 'family': 'Windows', 'major': '7', 'minor': None, 'patch': None, 'patch_minor': None}, 'string': 'Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like ' 'Gecko) Chrome/50.0.2661.102 Safari/537.36 CACHE CLEAR 1440x900', 'user_agent': { 'family': 'Chrome', 'major': '50', 'minor': '0', 'patch': '2661'}}
Fraud
{ 'device': {'brand': 'Spider', 'family': 'Spider', 'model': 'Desktop'}, 'os': { 'family': 'Other', 'major': None, 'minor': None, 'patch': None, 'patch_minor': None}, 'string': 'Jambot/0.1.x (Jambot; http://www.jambot.com/blog; ' 'crawler@jambot.com)', 'user_agent': { 'family': 'Jambot', 'major': '0', 'minor': '1', 'patch': None}}

An assumption of using the user agent string is that it should stay consistent across different events within a session and for sessions that are close enough in time. However, during account take over, the malware it operating as a “man in the browser” or session replication attack is taking place, the user agent string will change. One need to be able distinguish

between genuine user agent updates (i.e. customer using a different browser or a browser upgrade) to a fraudulent attack such as Bot performing account takeover.

In order to prove the above hypothesis, Locality Sensitive Hashing (LSH) algorithm is applied to cluster different browsers and OS, along with the score card to include as prior knowledge such as minor forward upgrade to browser is fine but backward upgrade is not. For each of the session the user agent will be examined using the above parser. For each of the class such as device, OS, Browser a Jaccard distance algorithm will be applied to measure any deviation. The Jaccard similarity function is given by:

$$Sim_{Jaccard}(UA_1, UA_2) = 1 - \frac{|UA_1 \cap UA_2|}{|UA_1 \cup UA_2|}$$

Once the Jaccard similarity is calculated, the following score card is applied to the similarity score:

If the device family is off type Spider add 60 base points to the device score.
If the device family is off type Spider and model is Desktop add 10 base points to the device score.
If the OS family is Windows add 10 base points
If the current browser major version is upgraded minus 5 base points
If the current browser minor version is upgrade minus 20 base points
If the current browser major version is degraded add 20 base points
If the current browser minor version is degraded add 30 base points

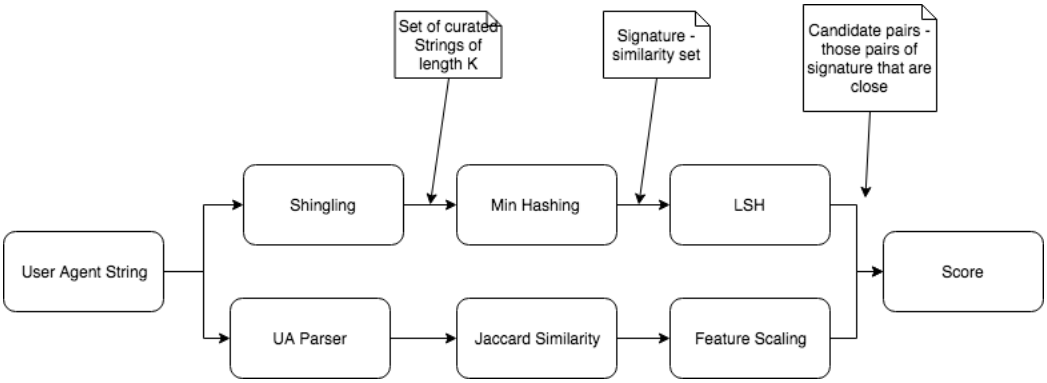
The overall sum is then calculated using the weighted sum, where the weights are the base points and all the distance output from device, OS and browser.

$$f(x) = \sum_{i}^{N} x_i w_i$$

Once the overall scores are computed, the score along with the original user agent string are clustered using the LSH. The final piece is to normalise the scores by applying the feature-scaling algorithm.

$$Z' = \sum_{i}^{N} S^i = \left(\frac{S - S_{min}}{S_{max} - S_{min}}\right)$$

The overall process is as follows:



Hence the overall score can be given as:

$$S(x) = \sum f(x)_i + J(x)_i + b_i$$

where $f(x)$is the Jaccard distance, $J(x)$ is the LSH and b is the overall biased. Picking from a sample populate the effect of the above can be seen as follows (Table 2):

CustomerID	Event	UserAgent String	Score
894993166	CustomerLogin	Mozilla/5.0 (Linux; Android 5.1.1; SM-G920F Build/LMY47X; wv) AppleWebKit/537.36 (KHTML, like Gecko) Version/4.0 Chrome/49.0.2623.105 Mobile Safari/537.36 CACHE CLEAR 1440x2560	0
894993166	CustomerLogin	Mozilla/5.0 (compatible; 008/0.83; http://www.80legs.com/spider.html;) Gecko/2008032620	0.99039
894993166	MakePayment	Mozilla/5.0 (compatible; 008/0.83; http://www.80legs.com/spider.html;) Gecko/2008032620	0.99039
1112788017	CustomerLogin	Mozilla/5.0 (Linux; Android 4.4.2; en-gb; SAMSUNG GT-I9505 Build/KOT49H) AppleWebKit/537.36 (KHTML, like Gecko) Version/1.5 Chrome/28.0.1500.94 Mobile Safari/53 CACHE CLEAR 1080x1920	0

1112788017	Makepayment	Mozilla/5.0 (compatible; Butterfly/1.0; +http://labs.topsy.com/butterfly.html) Gecko/2009032608 Firefox/3.0.8	0.99039

Overall score on user agent string on a sample of the dataset

Lists for IP address and user-agents are gathered from the Internet including fraudulent IP address and user agents' families of operating system, device and browser. A filter is built to check if an IP address in the dataset exists in the bad list. In the case there is a bad IP address in the dataset, the filter outputs 1 for fraud otherwise is 0. The IP and UA filters are used for both PasswordReset and CustomerLogin events separately. Table 3 below presents a sample of the new dataset after the filter are applied.

Password Reset				**Login**				
IP	**Latency**	**UA**	**UA score**	**IP**	**Latency**	**UA**	**UA score**	**Flag**
0	233	0	0	0	100	0	0	0
0	345	0	0.43	0	130	0	0	1
0	180	0	0	0	67	0	0	1
0	547	0	0.023	0	480	0	0	1

Sample of the dataset after the IP address and UA filters are applied

The new processed data were feed into an LSTM model with a many-to-one architecture. A many-to-one architecture takes a sequence of values as input and generates an output. The first input is the password reset with sequence of 4 and then customer login with the same length sequence as password reset.

The LSTM model learning parameters are tuned to determine the optimal parameter combination, as shown in Table. In LSTM model, the Soft-max Cross Entropy and Adam Optimiser are used as loss function and optimiser respectively. The optimal learning rate is 0.0014.

Parameter Name	**Value**
Learning rate	0.0014
Epochs	1000
Batch size	100
Number of layers	1
Number of hidden units	85
Loss function	Soft-max Cross Entropy
Optimiser	Adam Optimiser

Number of features	4
Number of inputs	2
Number of classes	2

LSTM tuned parameters of IP address and UA model

The figure below shows the loss levels over the number of training steps that the model takes. As the number of epochs increases, the training and validation loss instability decreases, and the loss value reaches zero. To evaluate the performance of the models F1 score, False Positive Rate (FPR) and False Negative Rate (FNR) are used. Fitted with the unseen datasets, the LSTM model managed to achieve a F1 score of approximately 98% as shown in Table 5. The FPR is zero meaning that the model classifies all the legitimate events as legitimate. Very close to zero is FNR, which indicates that very few fraudulent events are classified incorrectly as legitimate events.

Training and Validation loss of LSTM model

	LSTM
F1 Score	0.9772
False Positive Rate (FPR)	0
False Negative Rate (FNR)	0.045

F1 score, FPR and FNR of LSTM model

Benford's Law model

Benford's law states that if one randomly select a number from any naturally occurring tables of numerical data, the significant digits are not uniformly distributed as might be expected, but instead follow a particular logarithmic distribution. It states that the leading digit is not equally likely to be any one of the nine digits 1,2,..,9, but is 1 more than 30% likely to be and 9 less than 5% of the time. The probabilities for each of the nine digits to be the leading digit decrease monotonically. The probability for the first significant digit is

$$P(D_1 = d) = log_{10}(1 + {}^{1}/_{d}) \text{ for all d = 1,2,…,9}$$

The total fraudulent transactions used were 916 and the legitimate transactions were 21,733. The probability of a specific digit to be found in the dataset at the first, second, third and fourth position was calculated. Based on the probabilities found in the dataset and the expected Benford's law probabilities the Euclidean distance and chi-square was computed. A sample of the new dataset is shown in the Figure 3 below.

	TRNSD_TRNSAM	Found_1st	Found_2nd	Found_3rd	Found_4th	Ben_1st	Ben_2nd	Ben_3rd	Ben_4th	chi	distance	fraud_flags
0	10.0	0.325754	0.555452	0.0	0.0	0.301030	0.119679	0.0	0.0	0.0	0.460497	0
1	10.0	0.325754	0.555452	0.0	0.0	0.301030	0.119679	0.0	0.0	0.0	0.460497	0
2	6.5	0.049715	0.000000	0.0	0.0	0.066947	0.000000	0.0	0.0	0.0	0.017232	0
3	20.0	0.201510	0.555452	0.0	0.0	0.176091	0.119679	0.0	0.0	0.0	0.461192	0
4	20.0	0.201510	0.555452	0.0	0.0	0.176091	0.119679	0.0	0.0	0.0	0.461192	0

Data frame of Benford's law model

The data was imported into an LSTM and SVM model in order to test their performance. But before that the features were normalised and sampled. The sampling was made because the fraudulent transactions were very limited and for that reason the results of the model would not be skew.

The tuned parameters of LSTM and SVM model are shown in the table below:

LSTM		**SVM**	
Parameter Name	Value	**Kernel**	Radial Basis Function (RBF)
Learning rate	0.0014	**Cost ©**	10
Epochs	1000	**Gamma coefficient**	1
Batch size	100	**K-fold cross validation**	5
Number of layers	1		

Number of hidden units	85		
Loss function	Soft-max Cross Entropy		
Optimiser	Adam Optimiser		
Number of features	11		
Number of classes	2		

Tuned parameters of LSTM and SVM model

The table below shows the results of the two models. The SVM model has the lowest FPR as well as the lowest FNR. The low rates indicate that the classification of fraudulent and legitimate transactions was mostly correct. Also, the SVM has a better performance than the LSTM model with a F1 score of 70% whereas the LSTM model achieved 62%.

	LSTM	**SVM**
F1 Score	0.6250	0.7028
False Positive Rate (FPR)	0.404	0.289
False Negative Rate (FNR)	0.346	0.313

F1 score, FPR and FNR of LSTM and SVM

RMF

RFM (Recency, Frequency and Monetary) model is an analysis tool used to analyse the behaviour of a customer and then make predictions based on the behaviour in the dataset. Recency represents the time length since the last transaction, while frequency indicates the number of transactions within the specified time period and monetary denotes the amount of money spent in the specified time period.

In the fraud domain, transactional data such as transaction amounts can be used in the FRM technique where data can be aggregated in accordance with various time dimensions. Then different learning algorithms can be applied to identify fraud.

Due to time constrain the RFM model is not complete but some features were developed based on the raw data. From the dataset, the features TRANSACTION_AMOUNT and TRANSACTION_TIME were transformed in order to give the recency, frequency and monetary value for each customer. From TRANSACTION_TIME two new features were extracted: the day of the year and the time based on the 24-hour clock. TRANSACTION_AMOUNT was aggregated according to time resulting to the total amount of money and number of transactions each customer had in the specified time. Figure 4 below shows a sample of the new features grouped by the customer ID.

	party_id	day	time_hour	amount_sum	no_tnxs	fraud_flags
0	001221125	116	14	10.00	2	0
1	001221125	120	5	1120.00	3	0
2	001221125	120	9	190.00	1	0
3	001221125	124	11	164.00	2	0
4	001221125	126	8	224.00	3	0
5	001221125	126	9	22.00	3	0
6	001221125	128	2	100.00	1	0
7	001221125	131	1	90.00	1	0
8	001221125	132	12	20.00	1	0
9	001221125	134	7	12.00	1	0
10	001221125	139	15	19.00	2	0
11	001221125	142	17	1887.64	3	1
12	001221125	148	15	1200.00	1	0
13	001221125	148	16	2000.00	2	0
14	001221125	148	18	200.00	1	0
15	001221125	151	14	100.00	1	0
16	001221125	156	14	1075.00	1	1
17	001221125	156	15	100.00	1	1

RFM data frame