A VOTING-BASED HYBRID MACHINE LEARNING APPROACH FOR FRAUDULENT FINANCIAL DATA CLASSIFICATION

KULDEEP KAUR A/P RAGBIR SINGH

FACULTY OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY
UNIVERSITY OF MALAYA
KUALA LUMPUR

2019

A VOTING-BASED HYBRID MACHINE LEARNING APPROACH FOR FRAUDULENT FINANCIAL DATA CLASSIFICATION

KULDEEP KAUR A/P RAGBIR SINGH

THESIS SUBMITTED IN FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF COMPUTER SCIENCE

FACULTY OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY
UNIVERSITY OF MALAYA
KUALA LUMPUR

2019

UNIVERSITY OF MALAYA

ORIGINAL LITERARY WORK DECLARATION

Name of Candidate: Kuldeep Kaur A/P Ragbir Singh

Matric No: WMA180010

Name of Degree: Masters of Computer Science

Title of Project Paper/Research Report/Dissertation/Thesis ("this Work"):

A Voting-Based Hybrid Machine Learning Approach for Fraudulent Financial Data Classification

Field of Study: Computer Science

I do solemnly and sincerely declare that:

(1) I am the sole author/writer of this Work;
(2) This Work is original;
(3) Any use of any work in which copyright exists was done by way of fair dealing and for permitted purposes and any excerpt or extract from, or reference to or reproduction of any copyright work has been disclosed expressly and sufficiently and the title of the Work and its authorship have been acknowledged in this Work;
(4) I do not have any actual knowledge nor do I ought reasonably to know that the making of this work constitutes an infringement of any copyright work;
(5) I hereby assign all and every rights in the copyright to this Work to the University of Malaya ("UM"), who henceforth shall be owner of the copyright in this Work and that any reproduction or use in any form or by any means whatsoever is prohibited without the written consent of UM having been first had and obtained;
(6) I am fully aware that if in the course of making this Work I have infringed any copyright whether intentionally or otherwise, I may be subject to legal action or any other action as may be determined by UM.

Candidate's Signature Date: 23 August 2019

Subscribed and solemnly declared before,

Witness's Signature Date: 23 August 2019

Name:

Designation:

A VOTING-BASED HYBRID MACHINE LEARNING APPROACH FOR FRAUDULENT FINANCIAL DATA CLASSIFICATION

ABSTRACT

Credit card fraud is a growing concern in the financial industry. While financial losses from credit card fraud amount to billions of dollars each year, investigations on effective predictive models to identify fraud cases using real credit card data are limited currently, mainly due to confidentiality of customer information. To bridge this gap, this research embarks on developing a hybrid machine learning approach to identify credit card fraud cases based on both benchmark and real-world data. Standard base machine learning algorithms, which include a total of twelve individual methods as well as the AdaBoost and Bagging methods, are firstly used. The voting-based hybrid approach consisting of various machine learning models with the ability to tackle issues related to missing and imbalanced data is then developed. To evaluate the efficacy of the models, publicly available financial and credit card data sets are evaluated. A real credit card data set from a financial institution is also analysed, in order to evaluate the effectiveness of the proposed hybrid approach. In addition to the standard hybrid approach, a sliding window method is further evaluated using the real-world credit card data, with the aim to simulate and assess the capability of real-time identification of fraud cases at the financial institution. The empirical results positively indicate that the hybrid model with the sliding window method is able to yield a good accuracy rate of 82.4% in detecting fraud cases in real-world credit card transactions.

Keywords: Classification; fraud detection; hybrid model; credit cards; predictive modelling.

PENDEKATAN PEMBELAJARAN MESIN HIBRID BERASASKAN PENGUNDIAN UNTUK PENGKLASIFIKASIAN DATA KEWANGAN YANG PALSU

ABSTRAK

Penipuan kad kredit dalam industri kewangan amat membimbangkan. Walaupun kerugian kewangan dari penipuan kad kredit berjumlah berbilion ringgit setiap tahun, siasatan terhadap model ramalan yang berkesan untuk mengenal pasti kes-kes penipuan menggunakan data kad kredit sebenar adalah terhad, terutamanya kerana kerahsiaan maklumat pelanggan. Untuk merapatkan jurang ini, penyelidikan ini membangunkan pendekatan pembelajaran mesin hibrid untuk mengenal pasti kes-kes penipuan kad kredit berdasarkan data awam dan data dunia sebenar. Algoritma pembelajaran mesin asas piawai yang merangkumi sejumlah dua belas kaedah individu serta kaedah AdaBoost dan Bagging, digunakan terlebih dahulu. Pendekatan hibrid yang terdiri daripada pelbagai model pembelajaran mesin dengan keupayaan untuk menangani isu-isu yang berkaitan dengan data hilang dan tidak seimbang kemudiannya dibangunkan. Untuk menilai keberkesanan model, set data kewangan dan kad kredit awam yang dinilai. Data kad kredit sebenar yang ditetapkan dari institusi kewangan juga dianalisis, untuk menilai keberkesanan pendekatan hibrid yang dicadangkan. Di samping pendekatan hibrid piawai, kaedah tetingkap gelongsor dinilai dengan menggunakan data kad kredit dunia sebenar, dengan matlamat untuk mensimulasikan dan menilai keupayaan pengenalpastian masa nyata kes-kes penipuan di institusi kewangan. Keputusan empirikal secara positif menunjukkan bahawa model hibrid dengan kaedah tetingkap gelongsor mampu menghasilkan kadar ketepatan yang baik sebanyak 82.4% dalam mengesan kes-kes penipuan dalam transaksi kad kredit dunia sebenar.

Kata Kunci: Klasifikasi; pengesanan penipuan; model hibrid; kad kredit; pemodelan ramalan.

ACKNOWLEDGEMENTS

First and foremost, I offer my sincerest gratitude to my supervisor, Prof. Dr. Loo Chu Kiong who has supported me throughout my thesis with his patience, motivation and knowledge. One simply could not wish for a better or friendlier supervisor. My husband and my sister provided countless support, which was much appreciated. I would like to also thank my family, extended family and friends for all their support. Last, but not least, I would like to thank the internal and external examiners for their comments in improving my thesis.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS AND ABBREVIATIONS

AUC	:	Area Under the Curve
CV	:	Cross-Validation
DS	:	Decision Stump
DT	:	Decision Tree
DL	:	Deep Leaning
FN	:	False Negative
FP	:	False Positive
GBT	:	Gradient Boosting Tree
LIR	:	Linear Regression
LOR	:	Logistic Regression
NB	:	Naïve Bayes
NNBP	:	Neural Network with Back Propagation
RI	:	Rule Induction
RF	:	Random Forest
RM	:	Ringgit Malaysia
RT	:	Random Tree
SVM	:	Support Vector Machine
TN	:	True Negative
TP	:	True Positive
USD	:	United States Dollar

CHAPTER 1: INTRODUCTION

In this chapter, an overview of the research is first given. This is followed by the problem statement and research objectives. Organization of this thesis is given at the end of this chapter.

1.1 Overview

Fraud is a wrongful or criminal deception aimed to bring financial or personal gain (Sahin et al., 2013). To prevent loss from fraud, two types of methods can be utilized; fraud prevention and fraud detection. Fraud prevention is a proactive method, in which the fraud is stopped from its occurrence, while fraud detection aims to detect a fraudulent transaction by a fraudster as soon as possible.

A variety of payment cards, which include credit, charge, debit, and prepaid cards, are widely available nowadays. They are the most popular means of payments in some countries (Pavia et al., 2012). Indeed, advances in digital technologies have paved the way we handle money, especially in payment methods that have changed from being a physical activity to digital transactions over electronics means (Pavia et al., 2012). This has revolutionized the landscape of monetary policy, including business strategies and operations of both large and small companies.

Credit card fraud is an unlawful use of information from the credit card for the purpose of purchasing a product or service. Transactions can be either done physically or digitally (Adewumi & Akinyelu, 2017). In physical transactions, the credit card is present physically during the transactions. On the other hand, digital transactions take place over the internet or telephone. A cardholder normally gives the card number, card verification number, and expiry date through website or telephone.

With the rapid rise of e-commerce in the past years, usage of credit cards has tremendously increased (Srivastava et al., 2008). In Malaysia, the number of credit card transactions were about 317 million in 2011, and increased to 447 million in 2018 (BNM FSPSR, 2018). As reported by The Nilson Report (2016), the global credit card fraud in 2015 reached to a staggering USD $21.84 billion. The number of fraud cases has been rising with the increased use of credit cards. While various verification methods have been implemented, the number of credit card fraud cases have not been effectively reduced.

The potential of substantial monetary gains, combined with the ever-changing nature of financial services, creates a wide range of opportunities for fraudsters (Edge & Sampaio, 2012). Funds from payment card fraud are often used in criminal activities, e.g., to support terrorism acts which are hard to prevent (Everett, 2003). The internet is a place favoured by fraudsters as their identity and location are hidden.

The increase in credit card fraud directly hits the financial industry hard. Losses from credit card fraud mainly affects merchants, in which they bear all costs, including card issuer fees, charges, and administrative charges (Quah & Sriganesh, 2008). As merchants need to bear the loss, this comes with a price to the consumer where goods are priced higher, and discounts reduced. Hence, it is vital to reduce the loss. An effective fraud detection system is needed to eliminate or at least reduce the number of cases.

Numerous studies on credit card fraud detection have been conducted. The most commonly used methods are machine learning models, which include Artificial Neural Networks, Decision Trees, Logistic Regression, Rule-Induction techniques, and Support Vector Machines (Sahin et al., 2013). These methods can be either used standalone or merged in forming hybrid models.

Over the years, fraudulent mechanisms have evolved along with the models used by the banks in order to avoid detection (Bhattacharyya et al., 2011). Therefore, it is imperative to develop effective and efficient payment card fraud detection methods. The developed methods also need to be revised continually in accordance with the advances in technologies.

There are challenges in developing effective fraud detection methods. Researchers face the difficulty in obtaining real data samples from credit card transactions, as financial institutions are reluctant to share their data owing to confidentiality issues (Dal Pozzolo et al., 2014). This leads to limited research studies on using real credit card data in this domain.

1.2 Problem Statement

According to the American Bankers Association (Forbes, 2011), it is estimated that 10,000 credit card transactions occur every second across the world. Owing to such a high transaction frequency, credit cards become the targets of fraud. Indeed, credit card companies have been fighting against fraud since Diners Club issued the first credit card in 1950 (Forbes, 2011). Each year, billions of dollars are lost due to credit card fraud. Fraud cases occur under different conditions, e.g., transactions at the Point of Sales (POS), transactions made online or over the telephone, i.e., Card Not Present (CNP) cases, or transactions with lost and stolen cards. Credit card fraud reached $21.84 billion in 2015, with issuers bearing the cost of $15.72 billion (Nilson Report, 2016). Based on European Central Bank, in 2012, the majority (60%) of fraud stemmed from CNP transactions, and another 23% at POS terminals.

The value of fraud is high globally, and also locally here in Malaysia. The volume of credit, debit, and charge cards was at 383.8 million, 107.6 million, and 4.1 million, respectively in 2016 and increased to 447.1 million, 245.7 million, and 5.2 million, respectively in 2018 (Payment and Settlement Systems, 2018). The overall payment (i.e. credit, debit, and charge cards) fraud volume was at 0.0186% in 2016 and increased by 37.6% to 0.0256% in 2018 (Payment and Settlement Systems, 2018). Potential of huge monetary gains combined with the ever-changing nature of financial services give opportunities to fraudsters. In Malaysia, 1,000 card transactions occur every minute. Fraud directly hits merchants and financial institution, who incur all the costs. Increase in fraud affects customers' confidence in using electronic payments.

There are three main issues faced by financial institutions. Firstly, human intervention is typically required to stop fraud cases upon detection. Secondly, there are missing data from transactions which could happen during transmission of data to the fraud detection systems. Thirdly, the current fraud detection systems are based on foreign technology customized for foreign transactions, which also creates a high cost of acquisition.

1.3 Objectives of Study

Based on the issues faced by financial institutions, the main aim of this research is to identify fraudulent credit card transactions using a hybrid machine learning approach. The key research objectives are three-fold:

- to develop a hybrid approach using machine learning with the capability of recognizing patterns and stopping fraud cases without human intervention;
- to classify fraudulent credit card transaction patterns with missing data using the developed hybrid approach;
- to monitor and identify locally-based fraudulent credit card cases from time-series transaction data in real-time.

1.4 Research Scope and Significance

The financial fraud scope is given in Figure 1.1. In this study, the scope is focused on detection of real fraudulent credit card transactions in Malaysia.

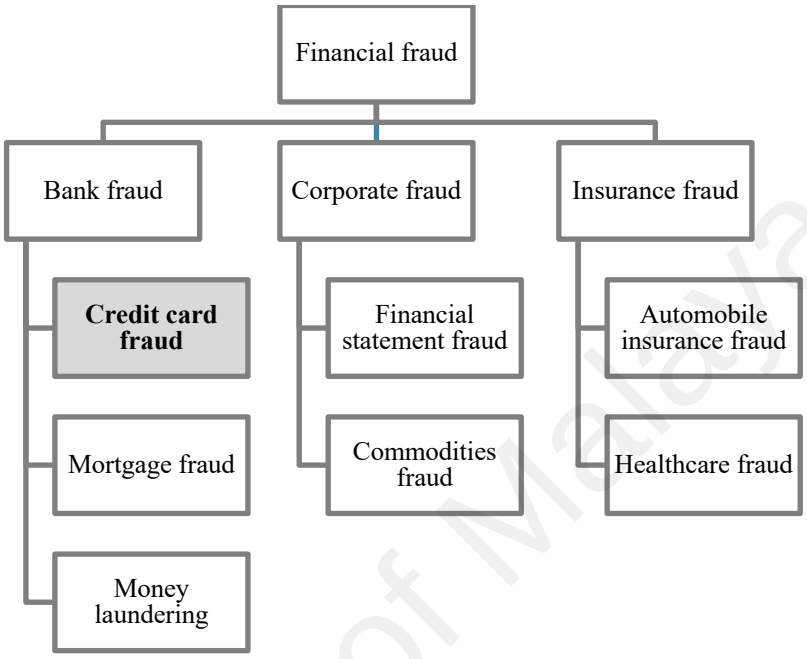

Figure 1.1: Scope of financial fraud (Popat & Chaudhary, 2018)

The research significance involves the design and development of a hybrid neural network for credit card fraud detection with the capabilities of addressing problems associated with class imbalance, missing data, and with real-time detection. The developed system offers a low-cost local technology customized to detecting fraudulent spending patterns of Malaysian cardholders.

1.5 Dissertation Organization

This dissertation is organized as follows. A literature review is first conducted in Chapter 2. The reviewed articles cover studies on credit and payment card fraud detection, with both benchmark and real-world data sets.

Various base models, from individual, AdaBoost, to Bagging are introduced in Chapter 3. Specifically, a total of twelve machine learning algorithms are used for detecting credit card fraud cases. The algorithms range from standard neural networks to deep learning models. Then, a hybrid machine learning approach is formulated and developed.

A series of systematic experiments using publicly available financial and credit card data sets is presented in Chapter 4. A total of four publicly available data sets are evaluated, with results compared to those in literature.

In Chapter 5, real-world credit card data from a financial institution is used to evaluate the developed hybrid model. The results are analysed and discussed. Finally, conclusions are drawn in Chapter 6. Contributions of this research are presented and a number of areas to be pursued as further work are suggested.

CHAPTER 2: LITERATURE REVIEW

A literature review encompassing credit and payment cards fraud detection is presented in this chapter. The literature review is structured into two main parts: individual and hybrid models. It is further divided into benchmark, synthetic, to real data from banks and the industry.

As the number of fraud cases are relatively small to the number of genuine transactions, an extreme class imbalance occurs in the data set. Most algorithms work well when the number of samples in each class are about equal, as the algorithms are designed to maximize accuracy and reduce error. Being a common problem in fraud detection, data imbalance can be resolved using sampling techniques.

Oversampling works by adding additional minority classes in the data. It can be used when there is not much data to work with. Undersampling works by removing some of the observations in the majority class. This can be a good choice when there is too much data, but one drawback is valuable data might be removed. This may lead to underfitting of the data set.

A number of metrics are available to evaluate the classifier performance. A common one is the confusion matrix. True Negative (TN) represents the number of normal transactions being flagged as normal while False Negative (FN) are the number of fraudulent transactions wrongly flagged as normal, i.e. missed fraud cases. True Positive (TP) are fraudulent transactions flagged as fraud, i.e. detected fraud cases while False Positive (FP) are the number of normal transactions flagged as fraud.

The Area Under the Curve (AUC) has been used in various domains. In the literature, there are two types of AUC. The Receiver Operating Characteristic (ROC)-AUC plots TP against FP. The Precision Recall (PR)-AUC plots precision against recall. In addition, f-measure (or F1 score) is the harmonic average of precision and recall, in which it reaches the highest score of 1 (perfect precision and recall) and the worst score of 0.

In the following sub-chapters, a review of the various models is done. The literature review encompasses the research objectives for developing a hybrid approach using machine learning, classifying fraudulent transactions, and identifying cases in real-time. A summary is given at the end of the chapter.

2.1 Individual Models

The individual models are reviewed in accordance with the types of data, i.e., benchmark, real data, and real data with feature aggregation.

2.1.1 Benchmark Data

Awoyemi et al. (2017), Manlangit et al. (2017), and Saia (2017) used the same data set of European cardholders that is available from Kaggle. It contained 284,807 transactions in a span of 2 days with 492 fraudulent transactions. A total of 30 attributes, consisting of Time, Amount, and 28 other features were transformed using the Principal Component Analysis (PCA). No details of the transformed attributes were given due to the sensitivity of the data.

A comparative analysis using Naïve Bayes (NB), *k*-nearest neighbor (*k*NN), and Logistic Regression (LOR) for credit card fraud detection was performed in Awoyemi et al. (2017). A hybrid technique with oversampling and under-sampling was used for analysing the skewed data. The results indicate the best accuracy rates for NB, *k*NN and LOR classifiers are 97.92%, 97.69%, and 54.86%, respectively (Awoyemi et al., 2017).

Analysis of credit card fraud was performed using Random Forest (RF), *k*NN, LOR, and NB in Manlangit et al. (2017). Data imbalance was addressed using a combination of undersampling and Synthetic Minority Oversampling Technique (SMOTE). The accuracy highest rate achieved by RF was 97.84%, followed by *k*NN (97.44%), LOR (94.49%), and NB (91.9%) (Manlangit et al., 2017).

A Discrete Wavelet Transform (DWT) approach was used in Saia (2017) for credit card fraud detection. No details of data sampling were provided. The f-scores and ROC-AUC for DWT were at 0.92 and 0.78, respectively, while for RF, it was at 0.95 and 0.98, respectively (Saia, 2017).

2.1.2 Real Data

A cost-sensitive decision tree approach that minimizes the sum of misclassification costs while using the splitting attribute for each non-terminal node was reported in Sahin et al. (2013). The data set included 22 million records, with 978 fraudulent cases. The data set was undersampled using stratified sampling. The Saved Loss Rate (SLR) was used as the performance indicator. It represents the saved percentage on the potential financial loss, i.e. the available usable limit of the cards which had fraudulent transactions. The highest SLR was at 95.8% using the Gini method (Sahin et al., 2013).

A Bayesian Network Classifier (BNC) algorithm was used in de Sá et al. (2018) for a real credit card fraud detection problem. The data set from a payment company in Brazil consisted of 887,162 genuine and 16,639 fraud transactions. Undersampling was conducted with the data. The data consisted of 24 attributes. BNC produced the highest F_1 score of 0.827 in the evaluation.

A data mining-based system was used in Carneiro et al. (2017) for credit card fraud detection. The data set was taken from an online luxury fashion retailer. The number of

features was 70, while the total transactions was not mentioned. Missing data were tackled using the imputation method. RF, SVM and LOR achieved ROC-AUC rates of 0.935, 0.906, and 0.907, respectively (Carneiro et al., 2017).

Artificial Immune Systems (AIS) was used by Brabazon et al. (2010), Wong et al. (2012), and Halvaiee and Akbari (2014) for credit card fraud detection. In Brabazon et al. (2010), the data set was provided by WebBiz, with 4 million transactions and 5417 fraudulent ones. Using a modified negative selection with AIS, an accuracy rate of 95.4% was achieved. In Wong et al. (2012), a data set from a major Australian bank was used. The data consisted of 640,361 transactions from 21,746 credit cards. The highest detection rate was 71.3%. In Halvaiee and Akbari (2014), the data set was from a Brazillian bank, with 3.74% of the transactions were fraudulent. The detection rate was at 0.518 with the FP rate at 0.017.

Association rules were applied to credit card fraud detection in Sánchez et al. (2009). The data set was taken from retail companies in Chile, which consisted of 13 features, including amount, age, and customer category. Using different confidence and support values, a certainty factor of 74.29% was presented for the rule typically used by risk experts (Sánchez et al., 2009).

The Modified Fisher Discriminant (MFD) method was used in Mahmoudi and Duman (2015) for credit card fraud detection. A data set from a bank in Turkey was examined, with 8,448 genuine and 939 fraudulent transactions. A total of 102 attributes were used. The developed model was skewed on correct classification of beneficial transactions, in order to maximize profit. MFD achieved a profit of 90.79%, which was higher as compared with that of the original Fisher method at 87.14% (Mahmoudi & Duman, 2015).

Credit card fraud detection was performed using the Long Short-Term Memory (LSTM) networks in Jurgovsky et al. (2018). Two data sets, ECOM and F2F, with 0.68 million and 0.97 million transactions, respectively, were used. Both data sets consisted of 9 features, and the data were undersampled. The PR-AUC for ECOM was 0.404 for RF, and 0.402 for LSTM, while for F2F, it was 0.242 for RF and 0.236 for LSTM (Jurgovsky et al., 2018), respectively.

Sequential fraud detection for prepaid cards using Hidden Markov Model (HMM) was investigated in Robinson and Aria (2018). The data set was taken from CardCom, consisting of 277,721 records with 9 features. The technique automatically created, updated, and compared HMM, with an average f-score of 0.7 (Robinson & Aria, 2018).

Detection of credit card fraud was reported in Minegishi and Niimi (2011) using a Very Fast Decision Tree learner. A data set consisting of 50,000 transactions with 84 attributes was used. Undersampling was performed, with a ratio of 1:9 for fraud to normal transactions. Accuracy rates from 71.188% to 92.325% were achieved (Minegishi & Niimi, 2011).

Hormozi et al. (2013) analysed credit card fraud detection by parallelizing a Negative Selection Algorithm using cloud computing. A total of 300,000 records from a Brazilian bank with 17 features from 2004 were utilized. Using a MapReduce framework, the detection rate hit as high as 93.08% was achieved (Hormozi et al., 2013).

Surrogate techniques in checking fraud detection technique for credit card operations were used in Salazar et al. (2014). The data set consisted of 8 million records, with 1,600 fraud cases. A total of 8 variables existed in the data. Using discriminant analysers, the ROC-AUC values from the experiments ranged from 0.8563 to 0.8708 (Salazar et al., 2014).

Credit card fraud detection based on Artificial Neural Networks (ANN) and Meta Cost was investigated in Ghobadi and Rohani (2016). A data set from a Brazilian credit card company with 3.75% of fraudulent transactions was used. A total of 18 attributes were available. In tackling imbalanced data, Meta Cost was used, where the model was named Cost Sensitive Neural Network (CSNN). The detection rates from the experiments were at 31.4% for ANN and 61.4% for CSNN (Ghobadi & Rohani, 2016).

Braun et al. (2017) aimed to improve credit card fraud detection through suspicious pattern discovery. A data set comprising of 517,569 transactions with 0.152% fraudulent transactions was used. The data set contained 21 features. Undersampling was done on the data, so that each set of data had 13,500 genuine and 1500 fraudulent transactions. The ROC-AUC and accuracy scores of RF and LOR were 0.971 and 99.9% as well as 0.944 and 99.6%, respectively (Braun et al., 2017).

A credit card fraud detection study for a bank in Turkey was reported in Duman and Elikucuk (2013) and Duman et al. (2013). A total of 22 million transactions with 978 fraudulent transactions and 28 different variables were analysed. Stratified sampling was carried out in both studies to balance the data. In Duman and Elikucuk (2013), the Migrating Birds Optimization (MBO) algorithm achieved the highest TP rate at 88.91%. In Duman et al. (2013), the highest TP rate was achieved by ANN at 91.74%.

Based on the review of individual models that use real data, it can be seen that most data sets originate from either payment, retail or banks. Most datasets have low amounts of fraud, creating an imbalanced data set. To resolve this issue, the authors used sampling techniques, with the undersampling most commonly used.

2.1.3 Real Data with Transaction Aggregation

In addition to standard features, some studies include aggregated features in addition to standard features. Transaction aggregation includes aggregated information related to the status of each account, which is continuously updated as new transactions occur. The use of additional features can directly influence the results of the models and has been found to be advantageous in many but not all circumstances (Whitrow et al., 2009).

An approach, named APATE, was proposed in Van Vlasselaer et al. (2015) for an automated credit card transaction fraud detection system using network-based extensions. A data set from a Belgian credit card issuer with 3.3 million transactions and 48,000 fraudulent data was used. A total of 60 new aggregated features, such as single merchant, country, currency, were created. The ROC-AUC and accuracy rate of LOR were 0.972 and 95.92%, ANN were 0.974 and 93.84%, and RF were 0.986 and 98.77%, respectively (Van Vlasselaer et al., 2015).

Feature engineering strategies using DT, LOR, and RF for credit card fraud detection was conducted in Bahnsen et al. (2016). A data set from a large European processing company with 120 million transactions was used. Based on the original 15 features, new aggregated features such as number of transactions and country from the past 24 hours were added. Using the proposed periodic features, the results showed an average accuracy increase of 13% (Bahnsen et al., 2016).

Credit card fraud detection using transaction aggregation was reported in Jha et al. (2012). A data set containing 49.8 million transactions over a period of 13 months was used. The original data had 14 primary features, and 16 new features such as average, amounts, and same merchants, were aggregated and added to the data. Using LOR, the model wrongly detected 377 transactions as fraud, while it wrongly detected 582 transactions as legitimate (Jha et al., 2012).

Transaction aggregation using multiple algorithms was examined in Whitrow et al. (2009) for credit card fraud detection. Two bank data sets were used, where Bank A data had 175 million transactions, with 5,946 fraudulent cases and 30 features. For Bank B, it had 1.1 million transactions, with 8,335 fraudulent cases and 91 features. Transaction aggregation depicted an advantage in many, but not all, circumstances. The loss function was calculated, where for Bank A and Bank B, the lowest loss was achieved by using RF with 7 days of aggregated data (Whitrow et al., 2009).

An evaluation of different methods for credit card fraud detection was done in Bhattacharyya et al. (2011). A total of 50 million real transactions from 1 million cardholders were used. In the experiments, a smaller dataset with 2,420 fraudulent transactions with 506 customers was analysed. On top of the original 14 features, additional 16 features such as average, amounts, same merchant, were added. The accuracy rates and AUC from the experiments were 94.7% and 0.942 for LOR, 93.8% and 0.908 for SVM, and 96.2% and 0.953 for RF, respectively (Bhattacharyya et al., 2011).

From the literature, it can be seen that the addition of aggregated features improved the algorithms ability to detect fraud. The use of transaction aggregation for various features are proposed for the experiments in Chapter 5.

2.2 Hybrid Models

The hybrid models are reviewed according to the types of data, namely benchmark, synthetic, and real data. Hybrid models are a combination of two or more classifiers that work together. It is typically designed for a particular task, where the combination of multiple models can greatly improve the final results.

2.2.1 Benchmark Data

A Deep Belief Network (DBN) based resampling SVM ensemble for classification was proposed in Yu et al. (2018). Two data sets from UCI, i.e., the German credit and Japanese credit data, were used. Both oversampling and undersampling were conducted on the data. The SVM model was used as a base classifier, creating ensemble input members to DBN. Using undersampling on the German credit data, the best results were achieved, with TP and TN of SVM at 72.7% and 63.93%, majority voting at 80.2% and 67.8%, and DBN at 87.9% and 61.6%, respectively. For the Japanese credit, the oversampling method performed the best, with TP and TN of SVM at 94.08% and 80.57%, majority voting at 94.5% and 81.14%, and DBN at 94.5% and 81.14%, respectively (Yu et al., 2018).

2.2.2 Synthetic Data

Synthetic data contain information created algorithmically and artificially manufactured rather than generated by real-world events. Kundu et al. (2009) and Panigrahi et al. (2009) applied synthetic credit card transaction records.

Two models, Basic Local Alignment Search Tool (BLAST) and Sequence Search and Alignment by Hashing Algorithm (SSAHA), named BLAST-SSAHA, were used in Kundu et al. (2009) for credit card fraud detection. A profile analyser determined the sequence similarity based on past spending sequences, while a deviation analyser

determined the possible past fraudulent behaviour. The TP rate varied from 65% to 100%, while the FP rate varied from 5% to 75% (Kundu et al., 2009).

A fusion approach using Dempster–Shafer theory and Bayesian learning was proposed in Panigrahi et al. (2009) for credit card fraud detection. The system consisted of a rule-based filter, Dempster–Shafer adder, transaction history database and Bayesian learner. Using the Dempster–Shafer theory, an initial belief was developed. The belief was further strengthened or weakened using Bayesian learning. The TP rate varied from 71% to 83% while the FP rate varied from 2% to 8% (Panigrahi et al., 2009).

2.2.3 Real Data

A framework for a hybrid model that consists of one-class classification and rule-based approaches for plastic card fraud detection systems was proposed in Krivko (2010). A total of 189 million transactions of real debit card data were used. Undersampling was performed on the data to divide them into smaller data sets. Hybrid and rule-based models were compared. The hybrid model identified only 27.6% of the compromised accounts while the rule-based method identified 29% (Krivko, 2010).

Duman and Ozcelik (2011) proposed the Genetic Algorithm and Scatter Search (GASS) method for detecting credit card fraud. Using data set from a bank in Turkey, undersampling ratios of 1:100 and 1:1000 were used on the data set. The method increased accuracy by up to 40% with the number of alerts being as many as four times from the suggested solution (Duman & Ozcelik, 2011).

A Scalable Real-time Fraud Finder (SCARFF), integrating Big Data tools (Kafka, Spark and Cassandra) with a machine learning approach was formulated in Carcillo et al. (2018). A total of 8 million transactions with 17 features were used. The experimental results indicated that on average, 24 out of 100 alerts were correct (Carcillo et al., 2018).

A credit card fraud prediction model based on cluster analysis and SVM was proposed in Wang and Han (2018). A data set from a bank in China was used. Undersampling ratio of 1:19 was used, with the data samples were clustered using *k*-means. A total of 3 models were then tested, i.e., the base SVM, KSVM (*k*-means with SVM), and KASVM (KSVM with AdaBoost). The AUC and f-measure results for SVM were 0.7755 and 0.975, KSVM were 0.7949 and 0.956, and KASVM were 0.9872 and 0.982, respectively (Wang & Han, 2018).

An ensemble consisting of six models was used in Kültür and Çağlayan (2017) for detection of credit card fraud. The six models consisted of DT, RF, Bayesian, NB, SVM, and *k*-models. A data set from a bank in Turkey which consisted of 152,706 transactions was used. Optimistic, pessimistic, and weighted voting were conducted in the experiments. Weighted voting yielded the highest accuracy at 97.55%, while optimistic voting showed the lowest FP rate at 0.1% (Kültür & Çağlayan, 2017).

A hybrid fuzzy expert system, FUZZGY, was proposed in HaratiNik et al. (2012) for credit card fraud detection. A real data set from a payment service provider was used. FUZZGY applied fuzzy rules, which identified logical contradiction between merchant current activities with trend of historical ones. The FP and TP rates from the experiments were 10% and 66% for a fuzzy expert system and 22.5% and 91.6% for FUZZGY, respectively (HaratiNik et al., 2012).

Heryadi et al. (2016) utilized Chi-Square Automatic Interaction Detection (CHAID) and *k*NN for detecting debit card fraud transaction. The data set used was taken from a bank in Indonesia, which consisted of 6,820 transactions with 1,939 fraudulent records. A total of 51 variables were used. The model achieved an accuracy rate of 72% (Heryadi et al., 2016).

2.3 Summary

It can be seen that researchers used various types of data from synthetic to benchmark, and real-world data, in financial fraud detection. Various models such as standard models of ANN to nature inspired metaheuristic approach, such as MBO have been used. The undersampling method is the most popular method used in order to tackle the imbalanced data problem, with various ratios used in different studies. Performance metrics of accuracy, TP, FP, and ROC-AUC have been used, but there is no standard metric in measuring the results. A summary of the performance across the various models is listed in Table 2.1.

Table 2.1: Performance comparison across models

Classifier		Reference	Acc.	TP	FP	ROC -AUC
Individual models						
Benchmark	NB	Awoyemi et al. (2017)	97.92%			
	*k*NN		97.69%			
	LOR		54.86%			
	RF	Manlangit et al. (2017)	97.84%			
	*k*NN		97.44%			
	LOR		94.49%			
	NB		91.90%			
	DWT	Saia (2017)				0.780
	RF					0.980
Real data	RF	Carneiro et al. (2017)				0.935
	SVM					0.906
	LOR					0.907
	AIS	Brabazon et al. (2010)	95.40%			
	AIS	Halvaiee and Akbari (2014)			0.017	
	AIS	Minegishi & Niimi (2011)	92.33%			
	MapReduce	Hormozi et al. (2013)	93.08%			
	Discriminant analysers	Salazar et al. (2014)				0.871
	RF	Braun et al. (2017)	99.9%			0.971
	LOR		99.6%			0.944

	MBO	Duman and Elikucuk (2013)	88.9%			
	ANN	Duman et al. (2013)		91.7%		
	LOR	Bhattacharyya et al. (2011)	94.70%			0.942
	SVM		93.80%			0.908
	RF		96.20%			0.953
Hybrid models						
Synthetic	BLAST-SSAHA	Kundu et al. (2009)		100%	75%	
	Dempster–Shafer + Bayesian	Panigrahi et al. (2009)		83%	8%	
Real data	KASVM	Wang & Han (2018)				0.987
	DT, RF, NB, SVM, Bayesian	Kültür & Çağlayan (2017)	97.50%			
	FUZZGY	HaratiNik et al. (2012)		91.6%	22.5%	
	CHAID + *k*NN	Heryadi et al. (2016)	72.00%			

It can be seen for individual models that the accuracy rates are generally above 90% for the benchmark experiments while RF acquired one of the highest accuracies and ROC-AUC rates. In hybrid models, there is no similar model across literatures, instead a combination of models, such as ensembles provide a boost to the individual models. In this work, both benchmark data and real data is used.

While the use of individual models can achieve good accuracy rates, the use of hybrid models in the literature have shown improved results as compared to individual models. Hybrid models are typically designed for a particular data set or task, and by combining two or more models, the overall results are greatly improved by each model adapting to the specific tasks. In addition, it can be seen that no model reported in the literature identifies fraudulent transactions in real-time.

CHAPTER 3: DEVELOPMENT OF HYBRID MODEL

In this chapter, various individual classifiers used in this study are described. Development of the hybrid machine learning approach is then presented. The novelty of this thesis is the proposition of a hybrid model (achieved via majority voting) to identify fraud in financial data.

3.1 Classifiers

In this study, a total of twelve classification algorithms are used. An overview of each algorithm with the settings used in RapidMiner is described, as follows.

3.1.1 Naïve Bayes

Naïve Bayes (NB) utilizes the Bayes' theorem with naïve or strong independence assumptions for classification. Some features of a class are assumed to be independent from others. It needs only a small training data set in estimating the mean and variance for classification. According to Bayes' theorem,

$$P(Y|\mathbf{X}) = \frac{P(Y) * P(\mathbf{X}|Y)}{P(\mathbf{X})}, \tag{3.1}$$

where input **X** comprises a set of *n* features/attributes of X_1, X_2, X_3, …, X_n, and Y is the label class; P(Y|**X**) is the posterior probability of class Y given **X**, P(**X**|Y) is conditional probability of input **X** given Y, while P(**X**) is the probability of evidence of **X**. A class label with the highest P(Y|**X**) is selected as the predicted output of input **X**. In an example with *n* attributes,

$$P(Y|\mathbf{X}) = \frac{P(Y) * \prod_{i=1}^{n} P(\mathbf{X_i}|Y)}{P(\mathbf{X})}, \tag{3.2}$$

The default settings used in RapidMiner was that of Laplace correction being used. Laplace correction is used in handling zero values, where it adds one to each count to avoid the occurrence of zero values.

3.1.2 Decision Tree

The DT model uses a set of nodes to connect the input features to certain classes. Each node denotes a splitting rule of a feature. The Gini impurity measure is used to determine how frequent a randomly chosen input sample is incorrectly labelled, which is computed using

$$\mathrm{G} = \sum (1 - {p_k}^2), \tag{3.3}$$

where p_k represents the proportion of samples in class k. New nodes are created till the stopping criterion is met. The class label is decided from the majority of samples that are from a particular leaf.

In RapidMiner, the default settings were used, with criterion of gain ratio, maximal depth of 20, pruning confidence of 0.25, prepruning minimal gain of 0.1, and minimal leaf size of 2. Gain ratio is a variant of information gain that adjusts the information gain for each attribute to allow the breadth and uniformity of the attribute values.

3.1.3 Random Tree

The Random Tree (RT) model functions as an operator of DT, with the difference on each split, only a random subset of input features is available. The learning process uses both numerical and nominal data samples. A subset is determined by a subset ratio parameter.

Similar to DT, the default settings were criterion of gain ratio, minimal size for split of 4, minimal leaf size of 2, minimal gain of 0.1, maximal depth of 20, and confidence of

0.25. Parameter confidence specifies the confidence level used for the pessimistic error calculation of pruning.

3.1.4 Random Forest

The model of Random Forest (RF) generates an ensemble of RTs. A user sets the number of trees, and the resulting RF model uses voting to determine the final classification outcome based on the predictions from all created trees. Structure of RF is shown in Figure 3.1. Classes are given as k while number of trees as T. The construction of RF is based on the bagging method, using random attribute selection.

Similar default settings to RT and DR, default settings were criterion of gain ratio and maximal depth of 20. The other default settings included number of trees of 10, pruning confidence of 0.25, minimal gain of 0.1, and minimal leaf size of 2.

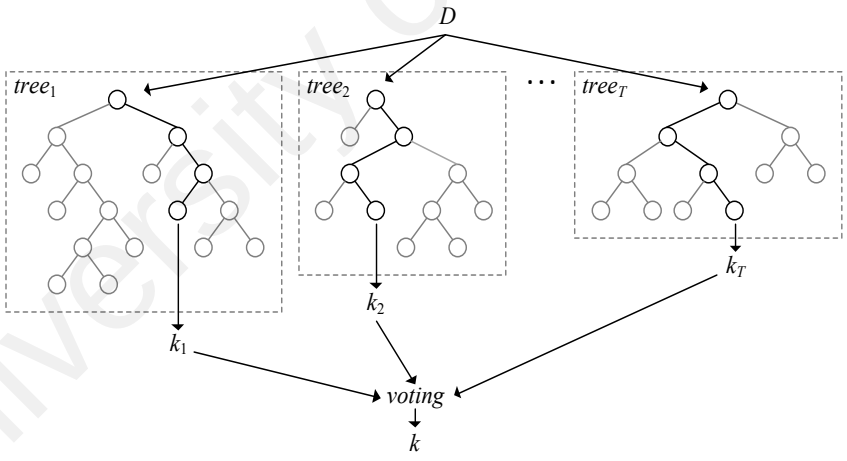

Figure 3.1: Structure of Random Forest

3.1.5 Gradient Boosted Tree

The Gradient Boosted Tree (GBT) is an ensemble model consisting of either regression or classification methods. It utilizes a forward-learning ensemble model to obtain predictive results using gradually improved estimations. Boosting assists to increase the tree accuracy.

In RapidMiner default settings, the number of trees of 20, maximal depth of 5, minimum rows of 10, number of bins of 20, and learning rate of 0.1 were used. While the default setting of learning rate of 0.1 was used, the range was from 0.0 to 1.0, which comes at the price of increasing computational time both during training and scoring, with lower learning rate requires more iterations.

3.1.6 Decision Stump

The Decision Stump (DS) model generates a DT with one split only. DS can be utilized to classify uneven data sets. It makes prediction from value of just one input feature, which is also called as 1-rules.

The default settings used in RapidMiner was criterion of gain ratio and a minimal leaf size of 1.

3.1.7 Neural Network with Back Propagation

The feed-forward Neural Network uses the supervised Back Propagation (NNBP) algorithm for training. The connections between the nodes do not form a directed cycle. Information only flows forward from the input nodes to the output nodes through the hidden nodes.

Default settings in RapidMiner included 2 hidden layers for the network, training cycles of 50, learning rate of 0.3, and momentum of 0.2. The momentum simply adds a fraction of the previous weight update to the current one, which prevents local maxima and smoothes optimization directions.

3.1.8 Linear Regression

Linear Regression (LIR) models the relationship of scalar variables by fitting a linear equation onto the observed data. The relationships are then modelled using linear predictor functions, where the unknown model parameters are estimated using the data

samples. When there are two or more predictors, the target output is a linear combination of the predictors, which can be expressed as

$$y = b_0 + b_1x_1 + b_2x_2 + \ldots + b_nx_n, \tag{3.4}$$

where y is the dependent variable, b_i's are the coefficients of x_i's, which are the explanatory variables. In a two-dimensional example, a straight line through the data samples is formed, whereby the predicted output, $\hat{y}$, for a scalar input x is given by

$$\hat{y} = b_0 + b_1x, \tag{3.5}$$

In RapidMiner, the default settings were used, which were minimum tolerance of 0.05 and ridge of 1E-8. The ridge parameter is used in ridge regression.

3.1.9 Logistic Regression

Another regression method, i.e., Logistic Regression (LOR), is able to handle both nominal and numerical features. It estimates the probability of a binary response based on one or more predictors. The linear function of predictor x is given by

$$\text{logit} = \frac{\log p}{1 - p} = b_0x + b_1, \tag{3.6}$$

where p is the probability of the event happening. Similar to Eq. (3.4), in the case involving independent variables, x_i's,

$$\text{logit} = b_0 + b_1x_1 + b_2x_2 + .. + b_nx_n, \tag{3.7}$$

The output probability is computed using

$$p = \frac{e^{\text{logit}}}{1 + e^{\text{logit}}}, \tag{3.8}$$

There is only a single default setting in RapidMiner, in which the solver was set to automatic.

3.1.10 Support Vector Machine

SVM handles both regression and classification problems. It creates a model by assigning new samples to a category or another, which creates a non-probabilistic binary linear classifier. The data in SVM are mapped in a way that samples from different categories can be segregated using a parallel margin, as wide as possible. A line (or a hyperplane in the general case) separating two attributes, x_1 and x_2, is established as

$$H = b + w \cdot x = 0, \tag{3.9}$$

where x is the input attribute vector, b is the bias, and w is the weight vector. In an optimal hyperplane, H_0, the margin, M, is given by

$$M = \frac{2}{\sqrt{w_0} \cdot w_0}, \tag{3.10}$$

where w_0 is formed with training samples, known as the support vectors, i.e.,

$$\left| w_0 = \sum y_{ii}\, x_i \right|, \tag{3.11}$$

The default settings in RapidMiner were used, which were kernel type of dot, convergence epsilon of 0.001, L positive of 1, and L negative of 1. Convergence epsilon is an optimizer parameter.

3.1.11 Rule Induction

The Rule Induction (RI) algorithm begins with less common classes and grows as well as prunes the rules until no positive instances are available, or the error rate is more than 50%. The rule accuracy, $A(r_i)$, is calculated using

$$A(r_i) = \frac{\text{Correct records covered by rule}}{\text{All records covered by rule}}, \tag{3.12}$$

During the growing phase, specific conditions are added to the rule until it is 100% accurate. During the pruning phase, the final sequence of each rule is removed using a pruning metric.

The default settings in RapidMiner were used, where criterion of information gain, sample ratio of 0.9, pureness of 0.9, and minimal prune benefit of 0.25 were selected. In information gain, the entropy of all attributes is calculated, and attribute with the minimum entropy is selected for split.

3.1.12 Deep Learning

Deep Learning (DL) is created from the base of a feedforward neural network trained using a stochastic gradient descent method with backpropagation. It has a large number of hidden layers, which consist of neurons with "tanh", "rectifier", and "maxout" activation functions. Each node takes a copy of the global model parameters from the local data, then periodically contributes towards the global model using model averaging.

In RapidMiner, the default settings were used, with activation of rectifier and epochs of 10. The activation function is used by neurons in the hidden layers while epochs detail the number of times a dataset should be iterated.

3.1.13 Classification Algorithm Strengths and Limitations

The strengths and limitations of each method discussed are given in Table 3.1.

Table 3.1: Strengths and limitations of machine learning methods

Models	Strengths	Limitations
Bayesian models	Good for classification problems; excellent use of computational resources; can be used in real-time operations.	Requires in depth understanding of normal and abnormal behaviours for various types of fraud cases.
Decision Trees	Simple to understand and implement; requires low computational power; good for real-time operations.	Potential of over-fitting if the training set does not represent the underlying domain information; re-training is needed for new fraud cases types.
Artificial Neural Networks	Good for classification problems; mainly used for fraud detection.	Need a high computational power, re-training is needed for new types of fraud cases.
Linear Regression	Provides optimal results when the relationship between independent and dependent variables are linear.	Sensitive to outliers.
Logistic Regression	Simple to implement, and historically used for fraud detection.	Poor classification performances when compared with other data mining methods.
Support Vector Machines	Can solve non-linear classification problems; requires little computational power; good for real-time operations.	Difficult to process the results due to the transformation of the input data.
Rule-based models	Easy to understand, and existing knowledge can be easily added.	Poor scaling with the training set size, and not suitable for noisy data.

3.2 Base Models

A total of three base models are used for the first set of experiments: Individual, AdaBoost, and Bagging. All models used are constructed graphically and simulations are conducted using RapidMiner.

3.2.1 Individual Models

In individual models, individual classifiers from a total of twelve algorithms detailed in Chapter 3.1 are used. The setup of individual model is shown in Figure 3.1. The process begins by retrieving data. Missing values are replaced. A sampling block is used, if the data set requires to be balanced.

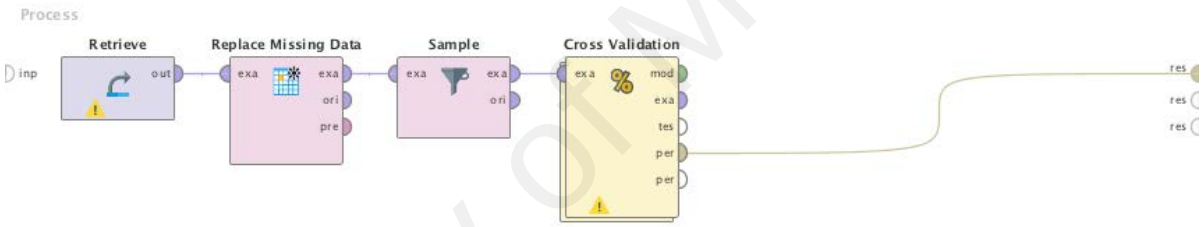

Figure 3.2: Setup of individual model

The cross-validation (CV) block, as shown in Figure 3.2, consists of the classifier, such as Naïve Bayes as given in the example. The performance, such as accuracy rate of the model, is then calculated.

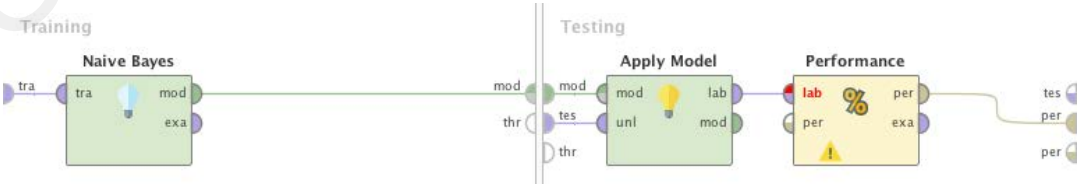

Figure 3.3: Expanded view of CV block for individual model

3.2.2 Adaptive Boosting (AdaBoost)

With a similar setup to the individual model, the AdaBoost model differs in which a block of classifier (as in Figure 3.2) is replaced with the AdaBoost block, as shown in Figure 3.3. AdaBoost adapts the subsequent weak learners in favour of instances wrongly classified by the classifier.

$$\mathrm{F}_T(x) = \sum_{t=1}^{T} f_t(x) \tag{3.13}$$

where f_t is a weak learner that takes object x as input and returns a value indicating the target class of an object. It is sensitive to outliers and noisy data, while being less vulnerable to overfitting problems. While individual learners can be weak, the final output model is proven to converge to a strong learner, provided that the performance of any weak learner is slightly better than random guessing.

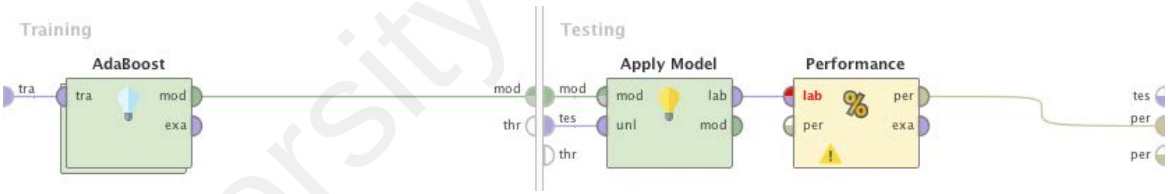

Figure 3.4: Expanded view of CV block for AdaBoost model

As an example, Naïve Bayes is used in Figure 3.4. The AdaBoost process completes in the Training section before moving to the Testing section. Default settings in RapidMiner was used, where the number of iterations was set to 5.

3.2.3 Bootstrap Aggregating (Bagging)

Similar to AdaBoost, the block is replaced with Bagging. Bagging is a machine learning ensemble meta-algorithm which increases accuracy and stability of classification algorithms. It helps reduce variance and avoids overfitting. Bagging is a special case of model averaging method. The setup is shown in Figure 3.5.

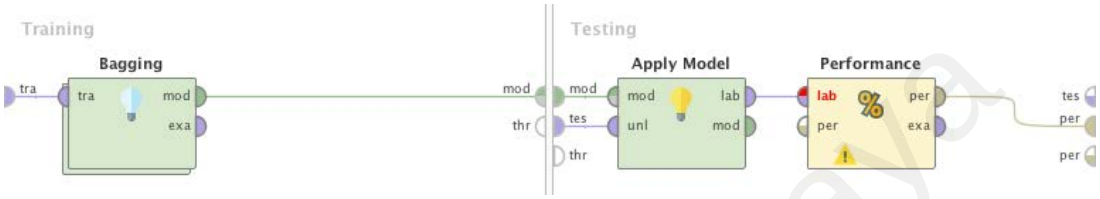

Figure 3.5: Expanded view of CV block for Bagging model

As an example, Naïve Bayes is used in Figure 3.5. Similar to Figure 3.4, once the Bagging process completes Training, it moves to the Testing section.

In total, twelve classification algorithms are used, where Naïve Bayes is one of them. Results from each of the algorithms are recorded in order to compare the performance. Similar to AdaBoost, the default settings in RapidMiner was used, where the number of iterations was set to 5.

3.3 Hybrid Machine Learning Approach

From the base models, a hybrid machine learning approach (hereinafter known as hybrid model) is developed. A hybrid model is a combination of two or more models. As discussed in Chapter 2.2, a variety of hybrid models have been used in the past by researchers.

Based on the individual models, the hybrid model is developed. The hybrid model is a complete system that can be used in the financial industry. A brief overview of the steps is as follows:

1) In this model, real-world data is first used.
2) If there are missing data, they will be imputed, where missing data are replaced with the mean value of that attribute.
3) If the data are unbalanced, the undersampling technique is used, where some of the majority class is removed.
4) The hybrid model uses a voting operator, with confidence and prediction of each model.

The pseudocode for the hybrid model is given in Table 3.2.

Table 3.2: Pseudocode of the hybrid model

Input: A set of data samples

Output: Prediction of transaction

while *each input sample* **do**

check if data is complete

if *missing values exists* **then**

replace missing values using imputation

end

check for number of samples in each class

if *data samples for each class differ >100 times* **then**

balance the data using undersampling

end

split data into training and prediction

train the data using hybrid model

predict new data using the hybrid model

compute output using majority voting operator in Eq. (3.14)

end

With the data samples checked for missing values and balanced, the data is then split as in Figure 3.6. The data set is first split into training and prediction sets, where both data sets are not overlapping each another.

The Vote block, as expanded in Figure 3.7, which has three classifiers, gives the output together with the prediction confidence. The performance measure provides the results, such as accuracy rates, sensitivity, and specificity.

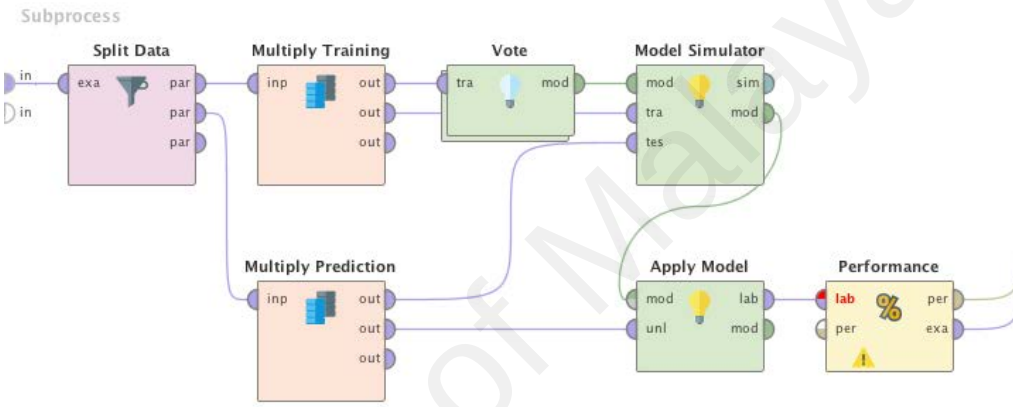

Figure 3.6: Expanded view of the Subprocess

A simple voting operator picks a winner based on the highest number of winning votes. Based on the literature review in Chapter 2.2, it can be seen that most hybrid models are made up from two or three classifiers. For instance, in the case of two classifiers voted against one classifier, the resulting winner will be from the two classifiers. To reduce the chance of bias, an odd number of classifiers is chosen, hence a total of three classifiers is chosen. Having more than three classifiers, such as five classifiers may slow down the identification of fraud when used in real-time.

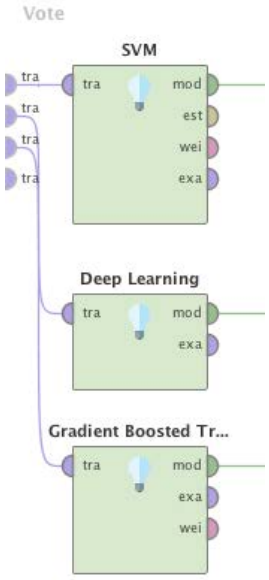

Figure 3.7: Expanded view of Vote block

A majority voting operator is developed for the experiments. The majority voting operator provides the final output based on the confidence and prediction of each model,

$$C(X) = \sum_{j=1}^{B} w_i p_{ij} \tag{3.14}$$

where w_j are the weights between 0 to 1 (1 being most confident), p_{ij} are the prediction, and B are the classifiers.

As compared with a simple voting operator, the majority voting operator takes into account of the confidence of each model output. A total of three classifiers, based on the best results from individual experiments, are used. SVM, DL, and GBT are the classifiers that perform best in most of the experiments (as in Chapters 4 and 5); therefore, they are used in the Vote model.

An example of how the majority voting operator works in given in Table 3.3. A total of three classifiers are used. The weight (confidence) for fraud and non-fraud is calculated for each classifier. Weight for both fraud and non-fraud is then averaged across the three classifiers.

Table 3.3: Sample of majority voting operator output

	Weight	
Model	**Fraud**	**Non-fraud**
SVM	0.99	0.01
DL	0.49	0.51
GBT	0.49	0.51
Average	0.66	0.18
Result	**Fraud**	**-**

As seen in the example above, the weight for fraud is 0.66 while non-fraud is 0.18. As the fraud has a much heavier weight, the data sample is said to be a fraudulent transaction. As compared to a simple voting operator, the result would have been non-fraud as both DL and GBT favour more towards non-fraud. In this case, the use of weights, or confidence from each model is helpful in predicting the fraud.

3.4 Summary

In this chapter, a total of twelve classifiers have been detailed. The strengths and limitations of the models are summarized. Development of the models are then done in stages. The individual models are first developed in RapidMiner. This is followed by AdaBoost and Bagging models.

From the results of the individual, AdaBoost and Bagging models, a hybrid machine learning approach is then developed. The hybrid model consists of three classifiers that performed the best in the experiments, i.e. SVM, DL, and GBT. A majority voting operator is used in summarizing the output prediction from the classifiers. In addition, the hybrid model includes the ability to handle missing information and imbalanced data.

CHAPTER 4: BENCHMARK EXPERIMENTS

In this chapter, a series benchmark experiments using publicly available data sets, from UCI Machine Learning Repository and Kaggle, is presented.

4.1 Experimental Setup

In this study, all experiments are conducted using RapidMiner Studio 7.6. All parameters are set based on the default settings in RapidMiner. The 10-fold cross-validation (CV) method is used in all experiments, as it reduces the bias associated with random sampling in the test stage. CV is known also as rotation estimation, where the data set is divided into train and test set, and each of the fold contains non-overlapping data for both training and testing.

The results from the 10-fold CV is then computed using the bootstrap method, where the averages were computed using a resampling rate of 5,000 to provide a good performance (Efron & Tibshirani 1993). Bootstrapping relies on random sampling with replacement.

Instead of describing the true and false positive rates and negative cases using one indicator, a good general measure is the Matthews Correlation Coefficient (MCC) (Powers, 2011). MCC measures the quality of a two-class problem, which considers the true and false positive and negative instances. It is a balanced measure, even with classes from various sizes. MCC can be calculated using

$$\mathrm{MCC} = \frac{TP \times TN - FP \times FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}, \tag{4.1}$$

where the result of +1 indicates a perfect prediction, and −1 a total disagreement.

4.2 UCI Data

Three data sets from the UCI Machine Learning Repository are used, namely Statlog (Australian Credit), Statlog (German Credit), and Default of Credit Card; hereinafter denoted as Australia, German, and Card, respectively.

4.2.1 Australia Data Set

In the first benchmark data set, there are a total of 690 instances with 14 variables and 2 classes. The data set is related to credit card applications. All attribute names and values have been changed to meaningless symbols, in order to protect confidentiality. Accuracy rates for the Australia data set are shown in Figure 4.1. In general, most of the classifiers acquire accuracy rates over 85%, with RT the lowest. DL acquires the highest accuracy rates. It can be seen that AdaBoost helps increase the accuracy rates for weak classifiers, such as NB and RT, but does the opposite for GBT and LIR. Bagging gives a higher accuracy rate for NB and RT, as compared with those of AdaBoost.

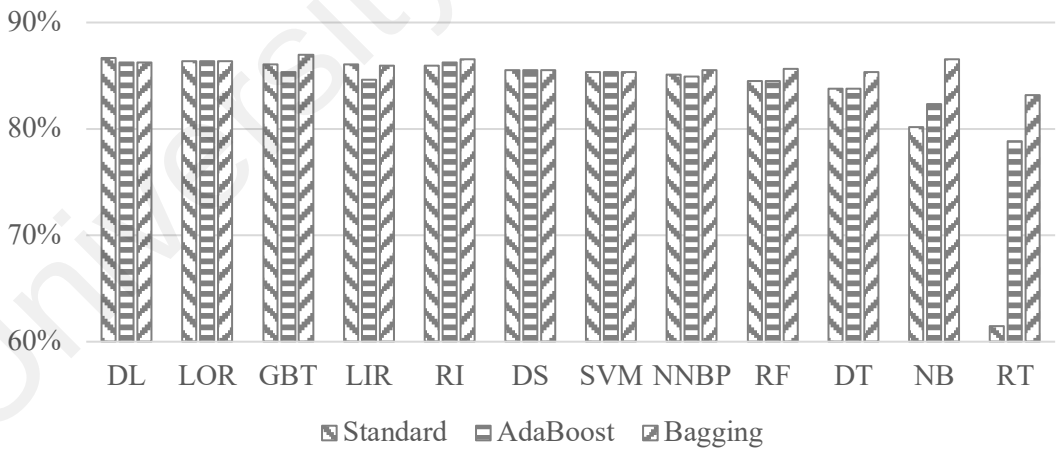

Figure 4.1: Accuracy rates for Australia data set

The MCC scores for the Australia data set are shown in Table 4.1. The highest rates are acquired using DL and LIR on the standard model at 0.730. There is a big boost of MCC for RT using AdaBoost and Bagging, while a moderate boost for NB. In some classifiers, AdaBoost and Bagging reduce the MCC rate, and in some cases, there is no difference. In the case of the classifier in AdaBoost and Bagging makes the wrong prediction multiple times, the MCC rate will fall. This is a downside of using a single type of classifier in AdaBoost and Bagging, hence selection of the right classifier is crucial.

Table 4.1: MCC rates for Australia data set

Model	Standard	AdaBoost	Bagging
DL	0.730	0.724	0.722
LIR	0.730	0.696	0.726
LOR	0.726	0.726	0.725
DS	0.720	0.720	0.720
GBT	0.719	0.705	0.738
SVM	0.716	0.716	0.716
RI	0.716	0.721	0.730
NNBP	0.702	0.698	0.707
RF	0.690	0.690	0.718
DT	0.681	0.681	0.716
NB	0.600	0.643	0.730
RT	0.231	0.572	0.659

For the Australian data set, the study in Ala'raj & Abbod (2016) is used for comparison. The best accuracy rate of 86.8%, as shown in Table 4.2, is yielded by RF w/GNG (Ala'raj & Abbod, 2016), DT w/GNG (Ala'raj & Abbod, 2016), and RF w/MARS (Ala'raj & Abbod, 2016). In comparison with the results, GBT (Bagging) produces comparable accuracy of 87.00%.

While the highest MCC scores were from DL and LIR, the accuracy rates for GBT (Bagging) were the highest. This is mainly due to the computation method for both

metrics. In MCC, a balanced approach which considers the true and false positive and negative instances is used. For accuracy, it is a measure of the correctly identified instances over the entire population, hence a highly imbalanced data set may have a high accuracy rate but a low MCC score if the minority class has a high number of wrongly identified instances.

Table 4.2: Comparison of accuracy using the Australia data set

Model	Accuracy
RF w/GNG (Ala'raj & Abbod, 2016)	86.80%
DT w/GNG (Ala'raj & Abbod, 2016)	86.80%
NB w/GNG (Ala'raj & Abbod, 2016)	86.50%
ANN w/GNG (Ala'raj & Abbod, 2016)	85.90%
SVM w/GNG (Ala'raj & Abbod, 2016)	86.30%
RF w/MARS (Ala'raj & Abbod, 2016)	86.80%
DT w/MARS (Ala'raj & Abbod, 2016)	82.80%
NB w/MARS (Ala'raj & Abbod, 2016)	78.50%
ANN w/MARS (Ala'raj & Abbod, 2016)	86.50%
SVM w/MARS (Ala'raj & Abbod, 2016)	85.30%
GBT (Bagging)	**87.00%**

4.2.2 German Data Set

The German data set consists of 1,000 instances with 20 variables and 2 classes. The data samples with numerical attributes are provided by Strathclyde University, where the data set has been edited with several variables added to make it suitable for algorithms which cannot handle categorical variables. The accuracy rates for the German data set are shown in Figure 4.2. It can be seen that LIR and LOR produce the best rates, while DS, DT, RF, and RT yield the same low rates at 70%. AdaBoost does not help much in most experiments, except for NB. Bagging helps increase the accuracy rates of LOR, RI, NNBP, GBT, and DL.

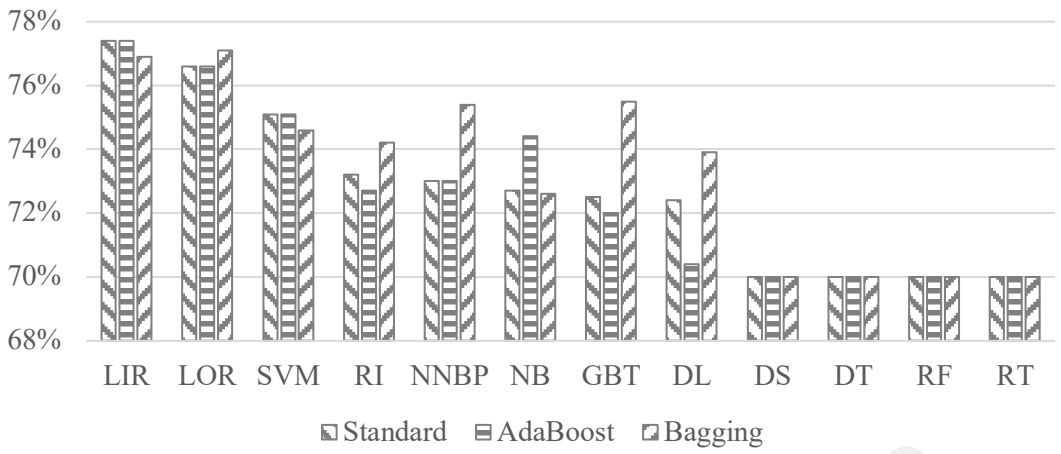

Figure 4.2: Accuracy rates for German data set

The MCC rates for the German data set are shown in Table 4.3. It can be seen that the rates are relatively low, between 0.3 and 0.4, with LIR achieves the highest. DS, RF, and RT rates are all at zero as MCC cannot be calculated. Similar to the accuracy rates, AdaBoost and Bagging are little to no help for most of the classifiers.

Table 4.3: MCC rates for German data set

Model	Standard	AdaBoost	Bagging
LIR	0.425	0.425	0.415
LOR	0.411	0.411	0.424
NB	0.379	0.386	0.374
GBT	0.368	0.342	0.359
RI	0.356	0.337	0.324
SVM	0.355	0.355	0.335
DL	0.344	0.324	0.384
NNBP	0.338	0.330	0.384
DT	0.039	0.039	0.039
DS	0.000	0.000	0.000
RF	0.000	0.000	0.000
RT	0.000	0.000	0.000

For the German data set, the results reported in Ala'raj & Abbod (2016), Cardoso et al. (2016), and Abellán et al. (2017) are used for comparison. A 5-fold CV was used in Ala'raj & Abbod (2016), while both Cardoso et al. (2016) and Abellán et al. (2017) used the 10-fold CV, i.e., the same as in this study. As shown in Table 4.4, the best accuracy achieved in the literature is from RF w/GNG (Ala'raj & Abbod, 2016) and SVM (Cardoso et al., 2016), while the best accuracy rate from this study is 77.40% using LIR.

Table 4.4: Comparison of accuracy using the German data set

Model	Accuracy
RF w/GNG (Ala'raj & Abbod, 2016)	77.00%
DT w/GNG (Ala'raj & Abbod, 2016)	74.50%
NB w/GNG (Ala'raj & Abbod, 2016)	75.90%
ANN w/GNG (Ala'raj & Abbod, 2016)	75.10%
SVM w/GNG (Ala'raj & Abbod, 2016)	76.80%
RF w/MARS (Ala'raj & Abbod, 2016)	76.70%
DT w/MARS (Ala'raj & Abbod, 2016)	72.10%
NB w/MARS (Ala'raj & Abbod, 2016)	74.40%
ANN w/MARS (Ala'raj & Abbod, 2016)	74.80%
SVM w/MARS (Ala'raj & Abbod, 2016)	76.60%
ClusWiSARD (Cardoso et al., 2016)	76.70%
SVM (Cardoso et al., 2016)	77.00%
BA-C4.5 (Abellán et al., 2017)	73.01%
BA-CDT (Abellán et al., 2017)	74.64%
RF (Abellán et al., 2017)	76.08%
CRF (Abellán et al., 2017)	76.38%
LIR	**77.40%**

4.2.3 Card Data Set

The Card data set consists of 30,000 instances with 24 variables and 2 classes. The data samples are from customers with default payments in Taiwan. The outcome is to evaluate whether the client is credible or not. The accuracy rates for the Card data set are given in Figure 4.3. It can be seen that most of the rates hover around 80%, with NB the lowest at 40%. Again, AdaBoost and Bagging are of little to no help in improving the results. NNBP produces the highest accuracy rate at 82.2%.

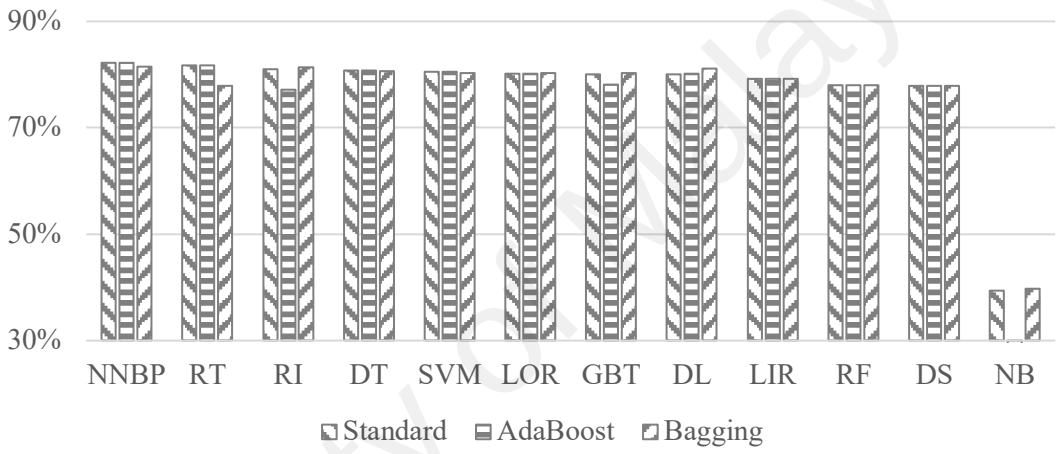

Figure 4.3: Accuracy rates for Card data set

The MCC rates for the Card data set are shown in Table 4.5. NNBP achieves the best MCC rate at 0.422 while DS the lowest at 0.035. Most classifiers produce MCC rates of around 0.3, while Bagging improves the performances of LOR, NB, DL, and RF. Bagging can improve the performance of certain classifiers when they constantly get the correct predictions.

Table 4.5: MCC rates for Card data set

Model	Standard	AdaBoost	Bagging
NNBP	0.422	0.422	0.383
RT	0.375	0.375	0.000
RI	0.347	0.205	0.358
GBT	0.343	0.285	0.329
SVM	0.335	0.335	0.300
DT	0.332	0.332	0.329
LOR	0.307	0.307	0.313
DL	0.306	0.325	0.363
LIR	0.231	0.231	0.231
NB	0.132	0.000	0.140
RF	0.058	0.058	0.068
DS	0.035	0.035	0.035

A performance comparison for the Card data set is conducted with those in Lu et al. (2017). The highest reported accuracy rate in Lu et al. (2017), as shown in Table 4.6, is 81.96% from RF. In this study, NNBP yields the best accuracy rate of 82.20%.

Table 4.6: Comparison of accuracy using the Card data set

Model	Accuracy
ELM (Lu et al., 2017)	76.86%
AdaBoost (Lu et al., 2017)	78.99%
KNN (Lu et al., 2017)	81.66%
SVM (Lu et al., 2017)	81.83%
RF (Lu et al., 2017)	81.96%
NB (Lu et al., 2017)	69.97%
NNBP	**82.20%**

4.3 Kaggle Data Set

A publicly available credit card data set is available from Kaggle. It has a total of 284,807 transactions made by European cardholders in September 2013. Only 492 fraudulent transactions are available, making it highly imbalanced.

For data protection reasons, a total of 28 principal components, namely V1 to V28 based on transformation are provided, except two features, i.e. Time and Amount. No metadata on the original features are given, hence pre-analysis or a study on the features is not possible in this case.

A correlation study is first conducted. A correlation matrix is a table that shows the correlation coefficients between the variables. Every cell in the table indicates the correlation in between two different variables. In other words, it can be used to summarize data that is present in the large data set.

Based on the correlation matrix in Figure 4.4, it can be seen that there is not much correlation between all the features, i.e. V1 to V28. Only Amount and the class label have some correlation between the features.

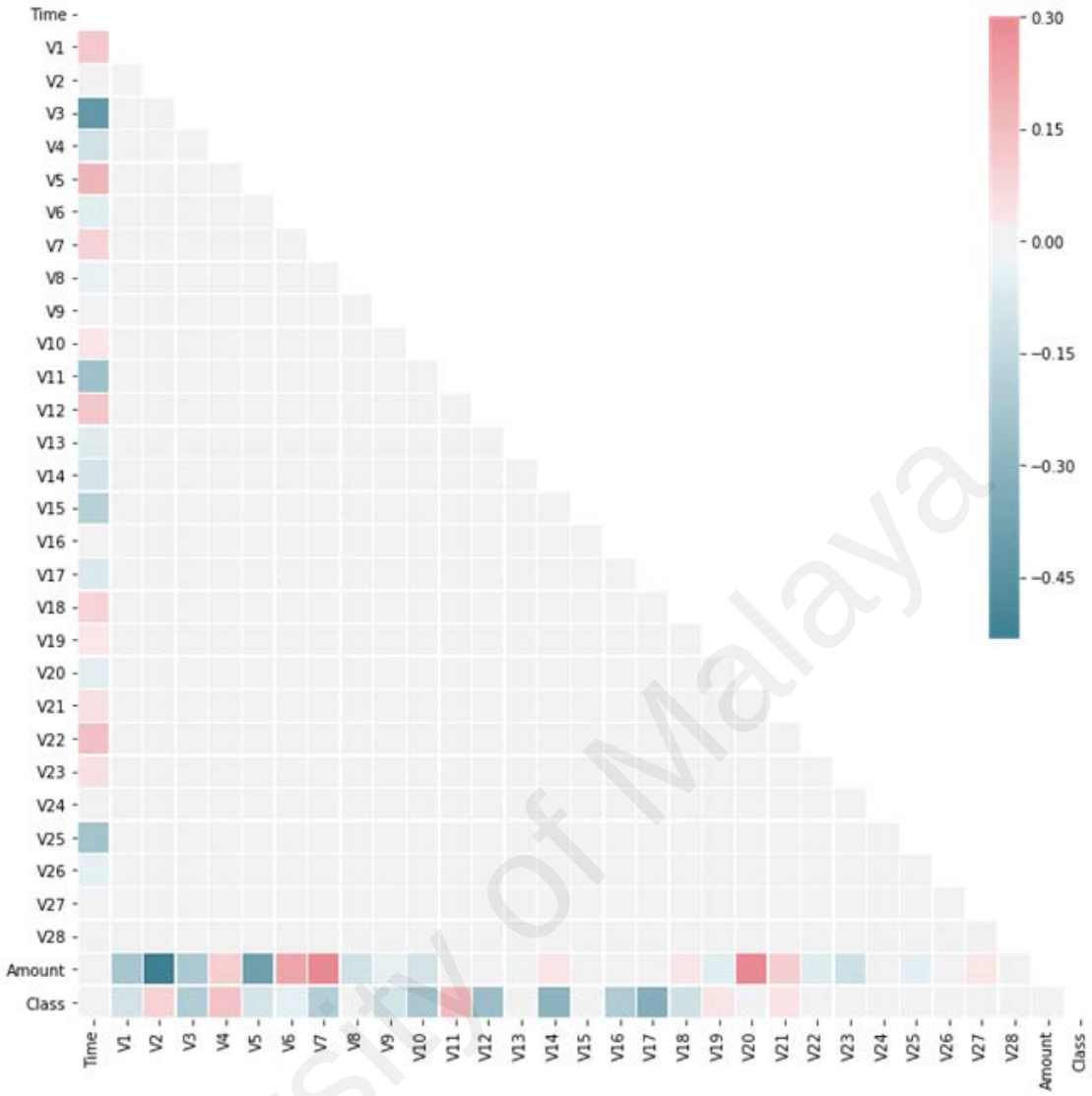

Figure 4.4: Correlation matrix for Kaggle data set

As the data set is highly imbalanced, undersampling is conducted. As reviewed in Chapter 2, the undersampling method is the most popular method used in order to tackle the imbalanced data problem various ratios were used, from 1:9 to 1:1000. Different ratios were used across different data sets, as the number of genuine to fraudulent transactions differed. In the experiments, two ratios are used, i.e. 1:50 and 1:100. These two ratios were selected based on the size of the data set and number of fraudulent transactions. The MCC, sensitivity, and specificity scores are presented and discussed in the following sub-chapters.

4.3.1 MCC

The MCC rates with ratio of 1:50 and 1:100 are shown in Figures 4.5 and 4.6, respectively. The results of both ratios are almost identical. While NNBP yields the highest MCC score from the ratio of 1:50, SVM produces the highest MCC score from the ratio of 1:100. Most MCC rates are above 0.8, close to 0.9, except that from RI (cannot be computed).

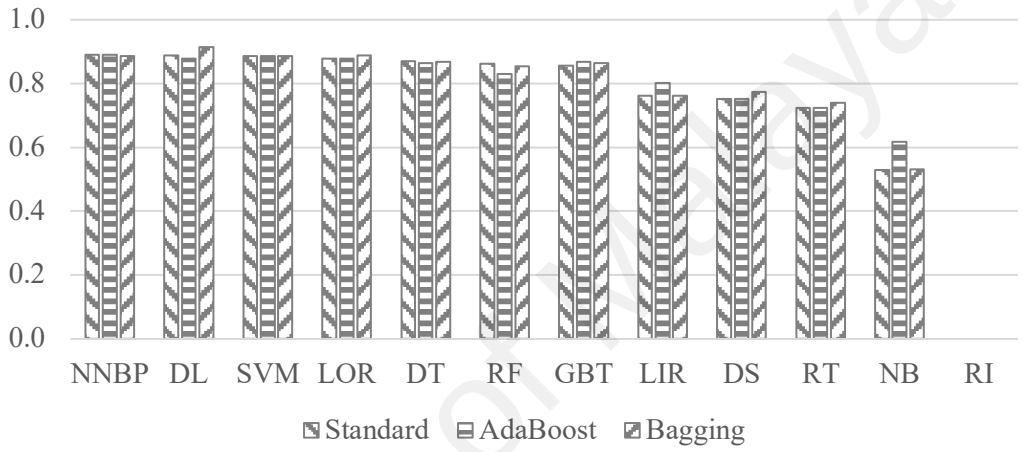

Figure 4.5: MCC rates for Kaggle data set, ratio 1:50

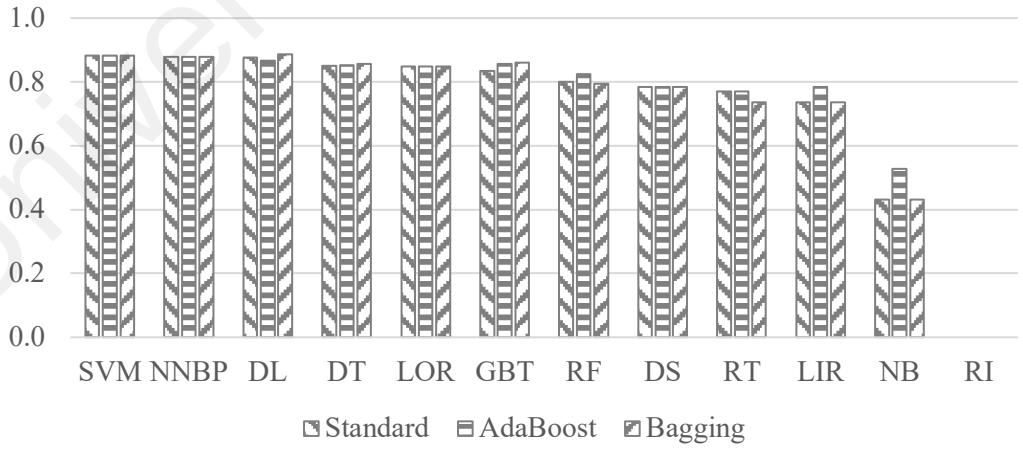

Figure 4.6: MCC rates for Kaggle data set, ratio 1:100

4.3.2 Sensitivity

The sensitivity rates with ratios of 1:50 and 1:100 are listed in Tables 4.7 and 4.8, respectively. Sensitivity in this case is the genuine transactions that are correctly classified. The results show good sensitivity rates, with most classifiers achieving 100% or close to 100%, for both ratios. Again, there is not much difference in sensitivity between both ratios.

Table 4.7: Sensitivity rates for Kaggle data set, ratio 1:50

Model	Standard	AdaBoost	Bagging
LIR	100.0%	99.6%	100.0%
RI	100.0%	100.0%	100.0%
SVM	100.0%	100.0%	100.0%
RF	100.0%	99.9%	100.0%
DS	100.0%	100.0%	100.0%
NNBP	100.0%	100.0%	100.0%
RT	100.0%	100.0%	100.0%
DL	99.9%	99.9%	100.0%
LOR	99.9%	99.9%	99.9%
DT	99.9%	99.9%	99.9%
GBT	99.9%	99.9%	99.9%
NB	97.4%	98.3%	97.4%

Table 4.8: Sensitivity rates for Kaggle data set, ratio 1:100

Model	Standard	AdaBoost	Bagging
LIR	100.0%	99.8%	100.0%
RI	100.0%	100.0%	100.0%
DS	100.0%	100.0%	100.0%
SVM	100.0%	100.0%	100.0%
NNBP	100.0%	100.0%	100.0%
RT	100.0%	100.0%	100.0%
RF	100.0%	100.0%	100.0%
DT	100.0%	99.9%	100.0%
LOR	99.9%	99.9%	99.9%
DL	99.9%	99.9%	100.0%
GBT	99.9%	100.0%	100.0%
NB	97.7%	98.6%	97.7%

4.3.3 Specificity

The specificity rates with ratios of 1:50 and 1:100 are listed in Tables 4.9 and 4.10, respectively. Specificity is the fraudulent cases that are correctly classifier. RI produces no results, indicating it cannot detect any fraudulent case. DL yields one of the best specificity rates in detecting fraudulent cases, which is followed by NB, GBT, and NNBP. Similar to the previous results, both ratios indicate similar performances.

Table 4.9: Specificity rates for Kaggle data set, ratio 1:50

Model	Standard	AdaBoost	Bagging
DL	83.7%	81.3%	80.5%
NB	82.1%	82.9%	82.1%
GBT	80.5%	78.9%	79.7%
NNBP	79.7%	79.7%	79.7%
SVM	78.9%	78.9%	78.9%
LOR	78.0%	78.0%	78.0%
DT	77.2%	78.9%	78.0%
RF	67.5%	68.3%	63.4%
DS	62.6%	62.6%	62.6%
RT	61.8%	61.8%	54.5%
LIR	54.5%	80.5%	54.5%
RI	0.0%	0.0%	0.0%

Table 4.10: Specificity rates for Kaggle data set, ratio 1:100

Model	Standard	AdaBoost	Bagging
DL	83.7%	82.1%	84.6%
LOR	82.1%	82.1%	82.9%
NB	82.1%	83.7%	82.1%
DT	81.3%	80.5%	80.5%
NNBP	81.3%	81.3%	80.5%
GBT	80.5%	79.7%	78.9%
SVM	78.9%	78.9%	78.9%
RF	75.6%	72.4%	73.2%
DS	58.5%	58.5%	61.8%
LIR	58.5%	83.7%	58.5%
RT	54.5%	54.5%	55.3%
RI	0.0%	0.0%	0.0%

4.3.4 Performance Comparison

For performance comparison, the results in Awoyemi et al. (2017) and Manlangit et al. (2017) are compared. Both studies use a train-test ratio of 70:30. A comparative analysis using NB, *k*NN, and LOR is included in Awoyemi et al. (2017), with a sampling rate of 10:90. A hybrid technique of under-sampling and oversampling is carried out on the skewed data. In Manlangit et al. (2017), an analysis using RF, *k*NN, LOR, and NB has been conducted. Data imbalance has been addressed using a combination of undersampling and Synthetic Minority Oversampling Technique (SMOTE).

The results are shown in Table 4.11. The sensitivity rates are all above 70%, with DL (Bagging) achieve a perfect score. The specificity rates are generally good except that of LOR (Awoyemi et al., 2017), while *k*NN (Awoyemi et al., 2017) yields a perfect specificity score.

Table 4.11: Comparison of accuracy and sensitivity using the Kaggle data set

Model	Sensitivity	Specificity
NB (Awoyemi et al., 2017)	82.1%	97.5%
*k*NN (Awoyemi et al., 2017)	82.9%	**100.0%**
LOR (Awoyemi et al., 2017)	71.6%	29.4%
RF (Manlangit et al., 2017)	96.1%	99.6%
*k*NN (Manlangit et al., 2017)	98.3%	96.6%
LOR (Manlangit et al., 2017)	91.6%	97.3%
NB (Manlangit et al., 2017)	86.0%	97.7%
DL (Bagging)	**100%**	84.6%

4.4 Summary

In this chapter, benchmark experiments from UCI (Australia, German, Card) and Kaggle have been conducted. Accuracy rates and MCC are presented and compared with those in the literature. As the Kaggle data set was highly imbalanced, two ratios (1:50 and 1:100) were used to balance the data. In addition, specificity and sensitivity rates were computed for the Kaggle data set. The standard, AdaBoost, and Bagging models that were developed in RapidMiner were able to perform as good as, or better with those reported in the literature. These models are then used in developing the hybrid model, which is used in the next chapter.

CHAPTER 5: REAL-WORLD EXPERIMENTS

A real credit card data set from a Malaysian financial institution is used in this experiment. The data set contains transactions from September to November 2016. A total of 68,597 records from 10,685 customers are available for evaluation. No identifying information about the customer such as name and address were taken, other than the masked account number.

The transactions cover activities in 124 countries, with various spending items ranging from online website purchases to grocery shopping. Among the transactions, 49% of them are made locally in Malaysia. This is followed by transactions made in Indonesia and Singapore.

Total of 28 transactions are labelled as fraud, with the remaining as genuine, or non-fraud cases. A general overview of the data set is given. The average transaction amount is RM 344.44, used over a total of five types of transactions (i.e. internet transaction, over the counter). There were 255 different types of Merchant Category Code (MECC) that were being transacted on.

The two top MECC were for airlines and advertising services. The trend for airlines is seen as more cardholders today book their flight and their related services bookings online. The highest number of accumulative transactions per cardholder over the data set period was 35, while the average transaction was 3. Most transactions take place in the evenings.

For the fraud cases, most of them had high number of total transactions, with an average of 20 per cardholder. The average transaction amount however did not have a specific amount, it ranged from a few ringgits to a few thousand. Most of the cases happen over the internet, with a few over the counter. The top two MECC codes that were part of the fraud cases were related to airlines and business services. This is in line with the total number of transactions seen in the data set.

A total of nine original features are acquired from the financial institution, consisting of two classes. The account number was masked for confidentiality purposes. A total of eight new aggregated features are added, which consist of the number and sum of the transaction amount, acquiring country, MECC, and transaction type. These eight features are added based on the literature review in Chapter 2.1.3. Feature aggregation adds information on each account status continuously, which is updated whenever a new transaction takes place. The list of features used is given in Table 5.1.

The experiments procedures were similar to those in Chapter 4. The 10-fold CV was used with results computed using bootstrap method with a resampling rate of 5,000.

Table 5.1: List of features

Features	Type
Masked Account Number	Original
Transaction Amount	Original
Transaction Date	Original
Transaction Time	Original
Device Type	Original
MECC	Original
Acquiring Country	Original
For Country	Original
Transaction Type	Original
Transaction Amount No.	Aggregated
Transaction Amount Sum	Aggregated
Acquiring Country No.	Aggregated
Acquiring Country Sum	Aggregated
MECC No.	Aggregated
MECC Sum	Aggregated
Device Type No.	Aggregated
Device Type Sum	Aggregated

5.1 Individual Models

This evaluation consists of the individual models, AdaBoost, and Bagging. The results of MCC, sensitivity, and specificity are discussed in the following sub-chapters. The same ratios of 1:50 and 1:100 are used.

5.1.1 MCC

MCC rates are shown in Figures 5.1 and 5.2. A similar trend can be seen, however the rates for the ratio of 1:100 are higher. GBT produces the highest MCC rates, which is followed by DL and NB. The remaining classifiers do not yield any MCC rates, due to the lack of specificity rates.

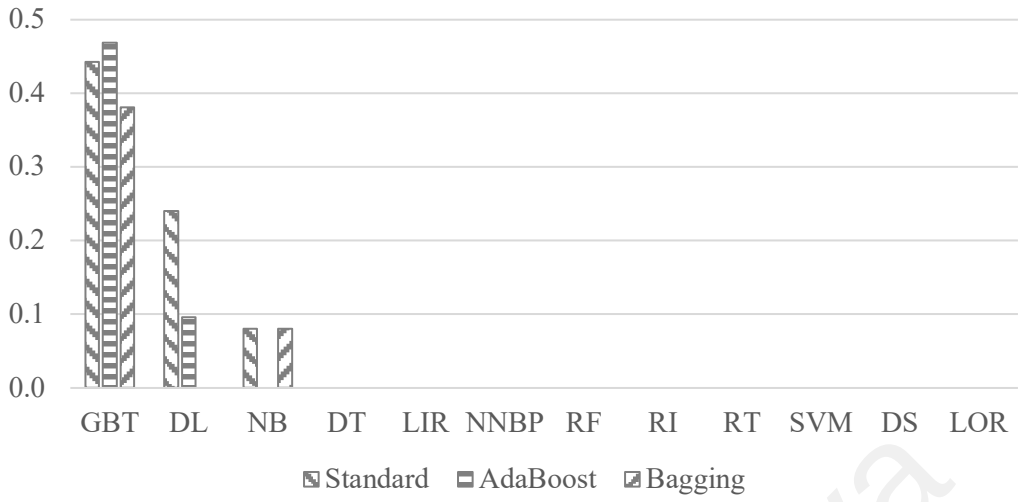

Figure 5.1: MCC rates for real-world data set, ratio 1:50

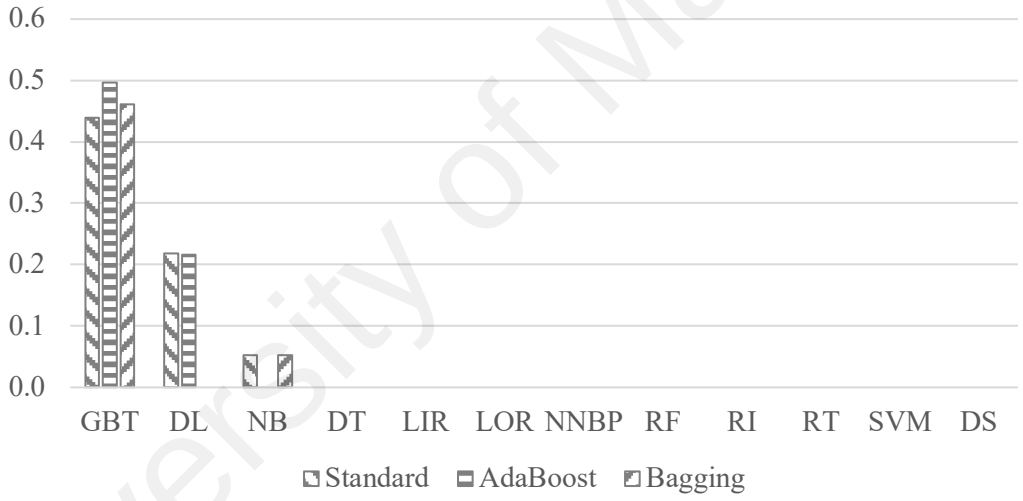

Figure 5.2: MCC rates for real-world data set, ratio 1:100

5.1.2 Sensitivity

The sensitivity rates with ratio of 1:50 are listed in Table 5.2. All classifiers, with the exception of NB, achieve perfect or near perfect scores. AdaBoost helps NB to achieve a perfect score, while Bagging does not help much.

Table 5.2: Sensitivity rates for real-world data set, ratio 1:50

Model	Standard	AdaBoost	Bagging
DT	100.0%	100.0%	100.0%
LIR	100.0%	100.0%	100.0%
NNBP	100.0%	100.0%	100.0%
RF	100.0%	100.0%	100.0%
RI	100.0%	100.0%	100.0%
RT	100.0%	99.9%	100.0%
SVM	100.0%	100.0%	100.0%
DS	99.9%	99.9%	99.9%
LOR	99.9%	99.9%	99.9%
GBT	99.1%	99.4%	99.9%
DL	96.2%	97.4%	100.0%
NB	29.7%	100.0%	29.6%

For ratio of 1:100, sensitivity rates are listed in Table 5.3. Similar to Table 5.2, NB acquired the lowest rates, with exception of being helped by AdaBoost.

Table 5.3: Sensitivity rates for real-world data set, ratio 1:100

Model	Standard	AdaBoost	Bagging
DT	100.0%	100.0%	100.0%
LIR	100.0%	100.0%	100.0%
LOR	100.0%	100.0%	100.0%
NNBP	100.0%	100.0%	100.0%
RF	100.0%	100.0%	100.0%
RI	100.0%	100.0%	100.0%
RT	100.0%	100.0%	100.0%
SVM	100.0%	100.0%	100.0%
DS	99.9%	99.9%	100.0%
GBT	99.5%	99.6%	100.0%
DL	97.4%	98.3%	100.0%
NB	31.3%	100.0%	31.4%

5.1.3 Specificity

The specificity rates are listed in Table 5.4 for ratio 1:50. Only NB, GBT, and DL produce the results, meaning that others cannot detect fraudulent transactions. NB produces a detection rate of 96.4% (ratio 1:50), which is followed by GBT and DL, at about 40%. AdaBoost and Bagging do not help improve the results.

Table 5.4: Specificity rates for real-world data set, ratio 1:50

Model	Standard	AdaBoost	Bagging
NB	96.4%	0.0%	96.4%
GBT	42.9%	39.3%	17.9%
DL	39.3%	14.3%	0.0%
DS	0.0%	0.0%	0.0%
DT	0.0%	0.0%	0.0%
LIR	0.0%	0.0%	0.0%
LOR	0.0%	0.0%	0.0%
NNBP	0.0%	0.0%	0.0%
RF	0.0%	0.0%	0.0%
RI	0.0%	0.0%	0.0%
RT	0.0%	0.0%	0.0%
SVM	0.0%	0.0%	0.0%

For ratio of 1:100, specificity rates are listed in Table 5.5. Similar results are recorded with ratio of 1:50, where both AdaBoost and Bagging do not help in improving the results.

Table 5.5: Specificity rates for real-world data set, ratio 1:100

Model	Standard	AdaBoost	Bagging
NB	92.9%	0.0%	92.9%
GBT	42.9%	46.4%	21.4%
DL	39.3%	32.1%	0.0%
DS	0.0%	0.0%	0.0%
DT	0.0%	0.0%	0.0%
LIR	0.0%	0.0%	0.0%
LOR	0.0%	0.0%	0.0%
NNBP	0.0%	0.0%	0.0%
RF	0.0%	0.0%	0.0%
RI	0.0%	0.0%	0.0%
RT	0.0%	0.0%	0.0%
SVM	0.0%	0.0%	0.0%

5.2 Hybrid Model

With the experiments using base models in Chapter 5.1 complete, the experiments are continued using the hybrid model developed in Chapter 3.3. The hybrid model consists of the classifiers that perform best in the experiments, which are SVM, DL, and GBT. Results from the experiments are listed in Table 5.6. Two ratios, 1:50 and 1:100 were used.

Table 5.6: Hybrid model results for real-world data set

Ratio	1:50	1:100
MCC	0.598	0.642
Sensitivity	100%	100%
Specificity	56.3%	62.5%

The MCC rates were at 0.598 and 0.642 for ratio of 1:50 and 1:100, respectively. The sensitivity rates were at 100%, similar to those in individual models. Specificity rates however was not as good. Using ratio of 1:50 and 1:100, the specificity rates were at 56.3% and 62.5%, respectively.

As compared with individual models, the results of the hybrid model is much higher. The hybrid model managed to acquire better rates than GBT, NB, SVM, and DL which scored well in the individual models.

5.3 Sliding Window Method

A sliding window method for prediction of transactions in real-time is designed for the experiments. The method is based on the developed hybrid model in Chapter 3.3. It consists of three classifiers, i.e. SVM, DL, GBT.

To simulate a real-time process in the financial institution, the evaluation conducted consists of multiple training and prediction days. A total of one and two training days with a combination of one, two, and three prediction days are evaluated. These evaluations are done separately, only the specific number of training and prediction days are taken.

A total of 17 features, which consists of the original and aggregated features as listed in Table 5.1 are used. For the experiments, a total of six sliding windows (i.e. one training day to one prediction day) are constructed based on scattered fraud are done. As an example, on two training and prediction days, for the first set, the training window covers Monday and Tuesday, while the prediction covers Wednesday and Thursday. For the second set, the training window covers Wednesday and Thursday, while the prediction covers Friday and Saturday. This process repeats a total of five times, with the results averaged and presented in Figures 5.3 and 5.4.

The sensitivity rates are shown in Figure 5.3. The sensitivity rates are all at 100%, which implies that regardless the training or prediction days, the genuine transactions can be classified correctly. The specificity rates are shown in Figure 5.4. The highest score comes from two days of training with one or two days of prediction, both at 82.4%, which is the fraud detection score. Training just one day is not sufficient, with a score lower than 10%. As such, it is important to have a longer training period, with a minimum of two days in order to get a good prediction outcome.

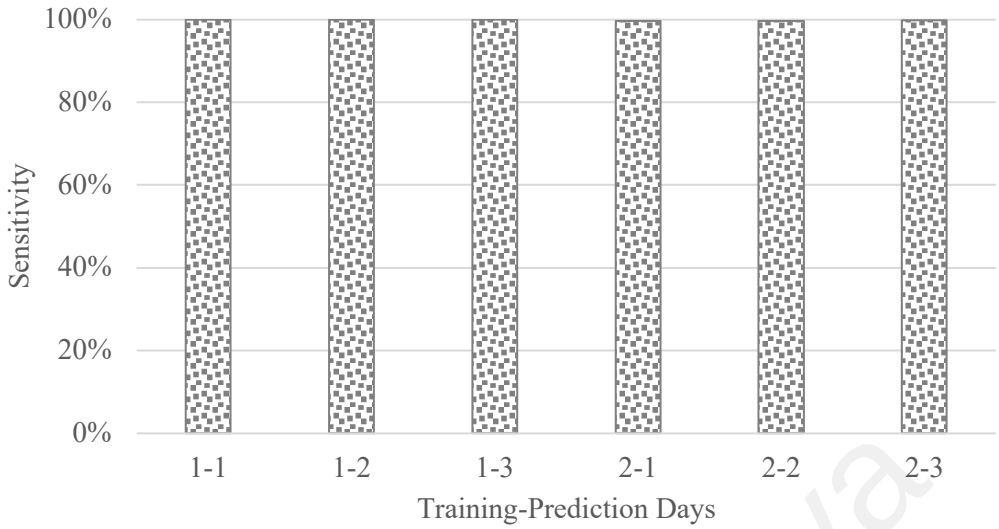

Figure 5.3: Sensitivity rates for sliding window model

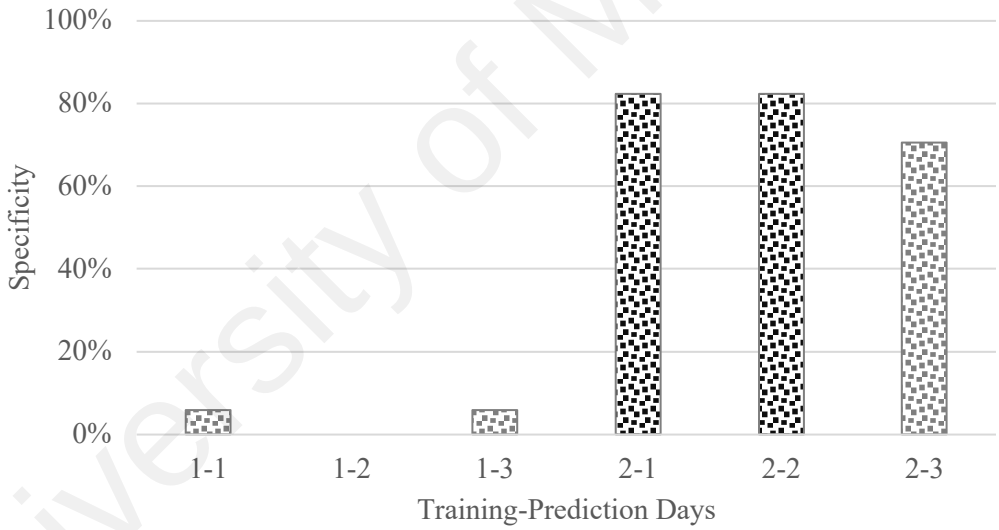

Figure 5.4: Specificity rates for sliding window model

Finally, the MCC rates from the sliding window model are shown in Figure 5.5. The results show the highest MCC rate, at 0.692, comes from two days of training with one day of prediction. This is followed by two days of training and two days of prediction, with an MCC rate of 0.557. It can be seen that the longer the training days, the better the prediction outcomes. Having a big enough training set is vital in order for the model to provide accurate predictions.

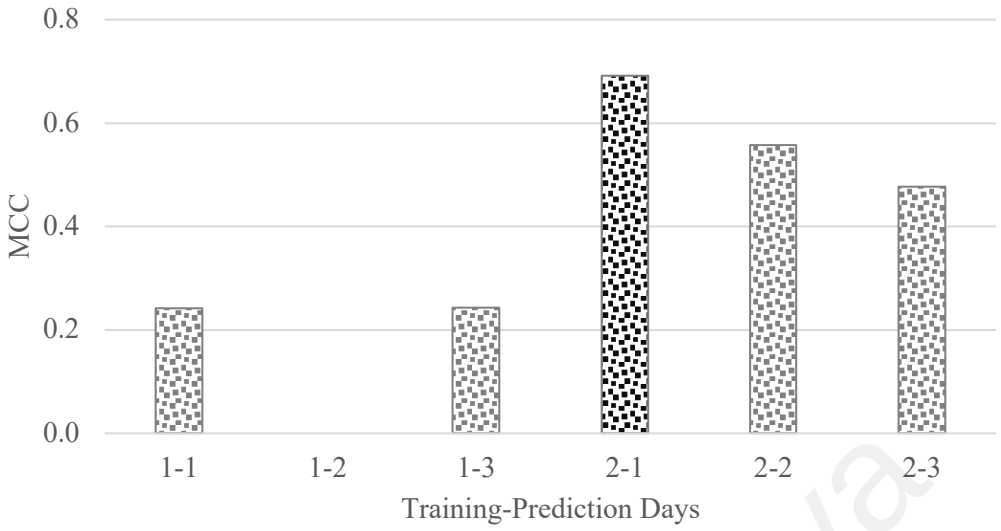

Figure 5.5: MCC rates for sliding window model

For financial institutions, the ability to detect genuine transactions correctly and fraud cases accurately is important. A genuine transaction that is labelled as fraud causes dissatisfaction from the affected customer. As such, having high sensitivity and specificity rates is vital for a prediction model to operate in the real-world.

5.4 Summary

In this chapter, an evaluation with real-world data from a financial institution in Malaysia has been conducted. Results from individual experiments indicate that DL and GBT acquire good results, as compared with the other classifiers. A hybrid model developed using classifiers that perform well in all experiments was then used in the experiments. The novelty of the hybrid model is the ability to deal with missing and imbalanced data and uses a voting operator that takes into account the confidence and prediction of each model. The MCC results from the hybrid model was 0.642, higher than individual models. In addition, another novelty of this thesis is a sliding window method that predicts days ahead of training is conducted, with the ability to accurately predict fraud cases up to two days in advance at 82.4%.

CHAPTER 6: CONCLUSIONS

In this chapter, concluding remarks are made and future research directions are discussed.

6.1 Conclusions

In this study, credit card fraud detection using machine learning models have been presented. A literature review was first conducted. Researchers used various types of data from synthetic to benchmark, and real-world data, in financial fraud detection. Various machine learning models, from individual to hybrid models were used. In detailing the performance, various metrics were used, with no standardization to quantify the results.

Development of models were then done. Individual models followed by AdaBoost and Bagging models were developed. Based on the results from these models, a hybrid model is then developed. The hybrid model uses a majority voting operator that combines three classifiers, i.e. SVM, DL, and GBT. Not only classifying data, the hybrid model handles missing information and also balances the data.

Publicly available data sets from UCI and Kaggle related to credit card fraud detection have been used for evaluation using standard and hybrid models. The metric of MCC has been used for measuring the performance, as it considers both true and false positive and negative predicted outcomes.

Initially, three benchmark UCI data sets were used for evaluations, namely Australian, German, and Card. GBT (Bagging), LIR, and NNBP achieved the best accuracy rates pertaining to the Australia, German, and Card data sets, respectively. Then, with the Kaggle credit card data set, two ratios were used in undersampling, i.e., 1:50 and 1:100.

The best MCC score was achieved by DL (Bagging) at 0.914, with 100% sensitivity and 84.6% specificity.

A real payment card data set from a financial institution in Malaysia was then used for evaluation. The data set contained three-month transactions from 10,685 customers. The individual and hybrid models used in the benchmark experiments were employed. Similar to the Kaggle data set, two ratios were used in undersampling, i.e., 1:50 and 1:100. Using standard models, the best MCC rates was acquired by GBT (AdaBoost) at 0.497. MCC results from the hybrid model was 0.642.

Based on the hybrid model, a sliding window model was then developed and evaluated. The test consisted of multiple training and prediction days, in order to simulate a real-time model used in the financial institution. While the best MCC rate was at 0.692, the hybrid model was able to predict fraud transactions up to two days ahead at an accuracy rate of 82.4%.

In summary, the main objective of this study, which is to detect fraudulent credit card transactions using a hybrid machine learning approach has been met. The contribution of this study includes the design of a hybrid approach that is able to recognize patterns from a large data set, classifying fraudulent credit card transaction patterns with missing data, and most importantly is the ability of predict fraudulent credit card cases in real-time.

6.2 Future Work

A number of enhancements can be made in future research. Firstly, newly developed machine learning models can be used to replace the standard models. The developed models can be customized specifically for the detection of fraudulent transactions.

The use of a parameter optimizer can be explored. Individual parameters for each setting in the classifier can be fine-tuned. New features based on existing features can also be extracted to enhance the fraud detection rate.

Next, supporting various types of payment methods, such as online banking can be explored. As the transactions happen in real-time on the web, a system that is able to detect fraudulent transactions in real-time will be able to save any potential money lost.

In addition, a model that covers spending patterns in various countries can be developed. As users spend differently in different parts of the world, it is important for a model to adapt so that fraudulent cases are not missed, while not wrongly classifying genuine transactions.

Finally, the hybrid model can be enhanced to support online learning. Not only that, other online learning models can be investigated. Online learning allows a rapid detection of fraud cases, potentially in real-time. This enables the detection and prevention of fraudulent transactions, which in turn reduces the number of incurred losses each day by the financial industry.

REFERENCES

Abellan, J., Mantas, C. J., & Castellano, J. G. (2017). A random forest approach using imprecise probabilities. *Knowledge-Based Systems, 134*, 72-84.

Adewumi, A. O., & Akinyelu, A. A. (2017). A survey of machine-learning and nature-inspired based credit card fraud detection techniques. *International Journal of System Assurance Engineering and Management*, *8*(2), 937-953.

Ala'raj, M., & Abbod, M. F. (2016). A new hybrid ensemble credit scoring model based on classifiers consensus system approach. *Expert Systems with Applications, 64*, 36-55.

Awoyemi, J. O., Adetunmbi, A. O., & Oluwadare, S. A. (2017, October). Credit card fraud detection using machine learning techniques: A comparative analysis. In *Computing Networking and Informatics (ICCNI), 2017 International Conference on* (pp. 1-9). IEEE.

Bahnsen, A. C., Aouada, D., Stojanovic, A., & Ottersten, B. (2016). Feature engineering strategies for credit card fraud detection. *Expert Systems with Applications*, *51*, 134-142.

Payment and Settlement Systems (2018, May) [Online]. Available: http://www.bnm.gov.my/files/publication/fsps/en/2018/cp05.pdf

Bhattacharyya, S., Jha, S., Tharakunnel, K., & Westland, J. C. (2011). Data mining for credit card fraud: A comparative study. *Decision Support Systems, 50*(3), 602-613.

Brabazon, A., Cahill, J., Keenan, P., & Walsh, D. (2010, July). Identifying online credit card fraud using artificial immune systems. In *Evolutionary Computation (CEC), 2010 IEEE Congress on* (pp. 1-7). IEEE.

Braun, F., Caelen, O., Smirnov, E. N., Kelk, S., & Lebichot, B. (2017, June). Improving card fraud detection through suspicious pattern discovery. In *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems* (pp. 181-190). Springer, Cham.

Bringing Trust Back to the Table - Part One: Adyen and Mobile Payments, Forbes, 2011.

Carcillo, F., Dal Pozzolo, A., Le Borgne, Y. A., Caelen, O., Mazzer, Y., & Bontempi, G. (2018). Scarff: a scalable framework for streaming credit card fraud detection with spark. *Information fusion*, *41*, 182-194.

Cardoso, D. O., Carvalho, D. S., Alves, D. S., Souza, D. F., Carneiro, H. C., Pedreira, C. E., & França, F. M. (2016). Financial credit analysis via a clustering weightless neural classifier. *Neurocomputing, 183*, 70-78.

Carneiro, N., Figueira, G., & Costa, M. (2017). A data mining based system for credit-card fraud detection in e-tail. *Decision Support Systems*, *95*, 91-101.

de Sá, A. G., Pereira, A. C., & Pappa, G. L. (2018). A customized classification algorithm for credit card fraud detection. *Engineering Applications of Artificial Intelligence, 72*, 21-29.

Duman, E., & Elikucuk, I. (2013, June). Solving credit card fraud detection problem by the new metaheuristics migrating birds optimization. In *International Work-Conference on Artificial Neural Networks* (pp. 62-71). Springer, Berlin, Heidelberg.

Duman, E., & Ozcelik, M. H. (2011). Detecting credit card fraud by genetic algorithm and scatter search. *Expert Systems with Applications, 38*(10), 13057-13063.

Duman, E., Buyukkaya, A., & Elikucuk, I. (2013, December). A novel and successful credit card fraud detection system implemented in a turkish bank. In *Data Mining Workshops (ICDMW), 2013 IEEE 13th International Conference on* (pp. 162-171). IEEE.

Edge, M. E., & Sampaio, P. R. F. (2012). The design of FFML: A rule-based policy modelling language for proactive fraud management in financial data streams. *Expert Systems with Applications*, *39*(11), 9966-9985.

Efron B. & Tibshirani R. (1993). An Introduction to the Bootstrap. Chapman and Hall, New York.

Everett, C. (2003). Credit card fraud funds terrorism. *Computer Fraud & Security*, *2003*(5), 1.

Ghobadi, F., & Rohani, M. (2016, December). Cost sensitive modeling of credit card fraud using neural network strategy. In *Signal Processing and Intelligent Systems (ICSPIS), International Conference of* (pp. 1-5). IEEE.

Halvaiee, N. S., & Akbari, M. K. (2014). A novel model for credit card fraud detection using Artificial Immune Systems. *Applied Soft Computing*, *24*, 40-49.

HaratiNik, M. R., Akrami, M., Khadivi, S., & Shajari, M. (2012, November). FUZZGY: A hybrid model for credit card fraud detection. In *Telecommunications (IST), 2012 Sixth International Symposium on* (pp. 1088-1093). IEEE.

Heryadi, Y., Wulandhari, L. A., & Abbas, B. S. (2016, November). Recognizing debit card fraud transaction using CHAID and K-nearest neighbor: Indonesian Bank case. In *Knowledge, Information and Creativity Support Systems (KICSS), 2016 11th International Conference on* (pp. 1-5). IEEE.

Hormozi, E., Akbari, M. K., Hormozi, H., & Javan, M. S. (2013, May). Accuracy evaluation of a credit card fraud detection system on Hadoop MapReduce. In *Information and Knowledge Technology (IKT), 2013 5th Conference on* (pp. 35-39). IEEE.

Jha, S., Guillen, M., & Westland, J. C. (2012). Employing transaction aggregation strategy to detect credit card fraud. *Expert systems with applications*, *39*(16), 12650-12657.

Jurgovsky, J., Granitzer, M., Ziegler, K., Calabretto, S., Portier, P. E., He-Guelton, L., & Caelen, O. (2018). Sequence classification for credit-card fraud detection. *Expert Systems with Applications, 100*, 234-245.

Kho, J. R. D., & Vea, L. A. (2017, November). Credit card fraud detection based on transaction behavior. In *Region 10 Conference, TENCON 2017-2017 IEEE* (pp. 1880-884). IEEE.

Krivko, M. (2010). A hybrid model for plastic card fraud detection systems. *Expert Systems with Applications, 37*(8), 6070-6076.

Kültür, Y., & Çağlayan, M. U. (2017). Hybrid approaches for detecting credit card fraud. *Expert Systems*, *34*(2).

Kundu, A., Panigrahi, S., Sural, S., & Majumdar, A. K. (2009). Blast-ssaha hybridization for credit card fraud detection. *IEEE transactions on dependable and Secure Computing*, *6*(4), 309-315.

Lu, H., Wang, H., & Yoon, S. W. (2017). Real Time Credit Card Default Classification Using Adaptive Boosting-Based Online Learning Algorithm. In *IIE Annual Conference. Proceedings* (pp. 422-427). Institute of Industrial and Systems Engineers (IISE).

Mahmoudi, N., & Duman, E. (2015). Detecting credit card fraud by modified Fisher discriminant analysis. *Expert Systems with Applications*, *42*(5), 2510-2516.

Manlangit, S., Azam, S., Shanmugam, B., Kannoorpatti, K., Jonkman, M., & Balasubramaniam, A. (2017, December). An Efficient Method for Detecting Fraudulent Transactions Using Classification Algorithms on an Anonymized Credit Card Data Set. In *International Conference on Intelligent Systems Design and Applications* (pp. 418-429). Springer, Cham.

Minegishi, T., & Niimi, A. (2011, February). Detection of fraud use of credit card by extended VFDT. In *Internet Security (WorldCIS), 2011 World Congress on* (pp. 152-159). IEEE.

Panigrahi, S., Kundu, A., Sural, S., & Majumdar, A. K. (2009). Credit card fraud detection: A fusion approach using Dempster–Shafer theory and Bayesian learning. *Information Fusion, 10*(4), 354-363.

Pavía, J. M., Veres-Ferrer, E. J., & Foix-Escura, G. (2012). Credit card incidents and control systems. *International Journal of Information Management*, *32*(6), 501-503.

Popat, R. R., & Chaudhary, J. (2018, May). A Survey on Credit Card Fraud Detection Using Machine Learning. In *2018 2nd International Conference on Trends in Electronics and Informatics (ICOEI)* (pp. 1120-1125). IEEE.

Powers, D. M. (2011). Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. *Journal of Machine Learning Technologies, 2*(1), 37-63.

Quah, J. T., & Sriganesh, M. (2008). Real-time credit card fraud detection using computational intelligence. *Expert Systems with Applications*, *35*(4), 1721-1732.

Robinson, W. N., & Aria, A. (2018). Sequential fraud detection for prepaid cards using hidden Markov model divergence. *Expert Systems with Applications*, *91*, 235-251.

Sahin, Y., Bulkan, S., & Duman, E. (2013). A cost-sensitive decision tree approach for fraud detection. *Expert Systems with Applications*, *40*(15), 5916-5923.

Saia, R. (2017, August). A Discrete Wavelet Transform Approach to Fraud Detection. In *International Conference on Network and System Security* (pp. 464-474). Springer, Cham.

Saia, R., & Carta, S. (2017). Evaluating Credit Card Transactions in the Frequency Domain for a Proactive Fraud Detection Approach. In *Proceedings of the 14th International Joint Conference on e-Business and Telecommunications* (Vol. 4, pp. 335-342).

Salazar, A., Safont, G., & Vergara, L. (2014, October). Surrogate techniques for testing fraud detection algorithms in credit card operations. In *Security Technology (ICCST), 2014 International Carnahan Conference on* (pp. 1-6). IEEE.

Sánchez, D., Vila, M. A., Cerda, L., & Serrano, J. M. (2009). Association rules applied to credit card fraud detection. *Expert systems with applications*, *36*(2), 3630-3640.

Srivastava, A., Kundu, A., Sural, S., & Majumdar, A. (2008). Credit card fraud detection using hidden Markov model. *IEEE Transactions on dependable and secure computing*, *5*(1), 37-48.

The Nilson Report (2016, October) [Online]. Available: https://www.nilsonreport.com/upload/content_promo/The_Nilson_Report_10-17-2016.pdf

Van Vlasselaer, V., Bravo, C., Caelen, O., Eliassi-Rad, T., Akoglu, L., Snoeck, M., & Baesens, B. (2015). APATE: A novel approach for automated credit card transaction fraud detection using network-based extensions. *Decision Support Systems, 75*, 38-48.

Wang, C., & Han, D. (2018). Credit card fraud forecasting model based on clustering analysis and integrated support vector machine. *Cluster Computing*, 1-6.

Whitrow, C., Hand, D. J., Juszczak, P., Weston, D., & Adams, N. M. (2009). Transaction aggregation as a strategy for credit card fraud detection. *Data Mining and Knowledge Discovery*, *18*(1), 30-55.

Wong, N., Ray, P., Stephens, G., & Lewis, L. (2012). Artificial immune systems for the detection of credit card fraud: an architecture, prototype and preliminary results. *Information Systems Journal*, *22*(1), 53-76.

Yu, L., Zhou, R., Tang, L., & Chen, R. (2018). A DBN-based resampling SVM ensemble learning paradigm for credit classification with imbalanced data. *Applied Soft Computing*, *69*, 192-202.

LIST OF PUBLICATIONS AND PAPERS PRESENTED

Randhawa, K., Loo, C. K., Seera, M., Lim, C. P., & Nandi, A. K. (2018). Credit card fraud detection using AdaBoost and majority voting. *IEEE Access, 6*, 14277-14284.