An Efficient Classification Model for Analyzing Skewed Data to Detect Frauds in the Financial Sector

Sara Makki

École Doctorale
Sciences et Technologies

N°d'ordre NNT : **2019LYSE1339**

THESE de DOCTORAT DE L'UNIVERSITE DE LYON
opérée au sein de
L'Université Claude Bernard Lyon 1

Ecole Doctorale N° 512
Informatique et Mathématiques de Lyon (InfoMaths)

Spécialité de doctorat :
Discipline : Informatique

Soutenue publiquement le 20/12/2019, par :
Sara MAKKI

An Efficient Classification Model for Analyzing Skewed Data to Detect Frauds in the Financial Sector

---

Devant le jury composé de :

Mme. MURISASCO Elisabeth, Professeure, Université de Toulon	Rapporteure
Mme. SOULE-DUPUY Chantal, Professeure, Université Toulouse	Rapporteure
M. BOUCELMA Omar, Professeur, Aix-Marseille Université	Examinateur
Mme. ASSAGHIR Zainab, Prof. Associée, Université Libanaise	Examinatrice
M. TAHER Yehia, McF, Université de Versailles	Examinateur
Mme. SEBA Hamida, McF-HDR, Université Lyon 1	Examinatrice
M. HACID Mohand-Saïd, Professeur, Université Lyon 1	Directeur de thèse
M. ZEINEDDINE Hassan, Professeur, Université Libanaise	Directeur de thèse
M. HAQUE AKM Rafiqul, Directeur de recherche, Intelligencia	Invité

To the soul of my grandfather ..

Declaration of Authorship

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements.

Sara A. Makki
December 2019

Acknowledgements

I take this opportunity to express my sincere gratitude to my teachers, colleagues, friends, and family. It is my pleasure to acknowledge the roles of everyone who was helpful and supportive for the completion of my PhD research.

I would like to thank my supervisors Pr. Hassan Zeineddine in Lebanon and Pr. Mohand-Saïd Hacid in France, for their support, their trust in my abilities and their caring. I also want to thank Dr. Zainab Assaghir for the great help and guidance she provided since she started teaching me in my masters years. My PhD was also partly supervised by Dr. Yehia Taher from the University of Versailles and Dr. Rafiqul Haque from intelligencia to whom I will always be grateful.

Besides my advisors, I would like to thank the rest of my thesis committee: Pr. Elisabeth Murisasco, Pr. Chantal Soule-Dupuy, Pr. Omar Boucelma and Dr. Hamida Seba for their insightful comments and constructive feedback; and also for the great discussions that encouraged me to widen my research to various perspectives.

I am incredibly grateful to my friends, for the unforgettable and joyful moments we shared that made those three years much more easier. I specifically want to thank the friends that I literally couldn't have made it without their continuous daily support, Ali Janbain, Mouhammad Ghader, Joseph Hadchiti, and my friend for almost eight years, Rayane Hashem. I want to thank the friends that I shared the best, the worst and the funniest moments of my life with, Ghina Nassredine, Lara Daw, Inès Abdallah, Nizar Obeid, Cihan Tunç, Mohammad Akil and Lania Hammoud. I would also like to thank Ali Masri and Khodor Hammoud for their help every time I needed it.

Finally, I want to thank my family for the unconditional love. I would not have made it this far without the huge support that my mother provided everyday. I want to thank my father and brother for being always by my side. I cannot thank enough my aunt Yamama Chreim, the inspiration to any strong independent woman; and my uncle Ali Makki for his love and caring. Finally, I want to thank my cousins Fayez-Rayan and David for always bringing joy and laughter to my heart.

Abstract

There are different types of risks in financial domain such as, terrorist financing, money laundering, credit card fraudulence and insurance fraudulence that may result in catastrophic consequences for entities such as banks or insurance companies. These financial risks are usually detected using classification algorithms.

In classification problems, the skewed distribution of classes also known as class imbalance, is a very common challenge in financial fraud detection, where special data mining approaches are used along with the traditional classification algorithms to tackle this issue.

Imbalance class problem occurs when one of the classes have more instances than another class. This problem is more vulnerable when we consider big data context. The data sets that are used to build and train the models contain an extremely small portion of minority group also known as positives in comparison to the majority class known as negatives. In most of the cases, it's more delicate and crucial to correctly classify the minority group rather than the other group, like fraud detection, disease diagnosis, etc. In these examples, the fraud and the disease are the minority groups and it's more delicate to detect a fraud record because of its dangerous consequences, than a normal one. These class data proportions make it very difficult to the machine learning classifier to learn the characteristics and patterns of the minority group. These classifiers will be biased towards the majority group because of their many examples in the data set and will learn to classify them much faster than the other group.

After conducting a thorough study to investigate the challenges faced in the class imbalance cases, we found that we still can't reach an acceptable sensitivity (i.e. good classification of minority group) without a significant decrease of accuracy. This leads to another challenge which is the choice of performance measures used to evaluate models. In these cases, this choice is not straightforward, the accuracy or sensitivity alone are misleading. We use other measures like precision-recall curve or F_1 - score to evaluate this trade-off between accuracy and sensitivity. Our objective is to build an imbalanced

classification model that considers the extreme class imbalance and the false alarms, in a big data framework.

We developed two approaches: A Cost-Sensitive Cosine Similarity K-Nearest Neighbor (CoSKNN) as a single classifier, and a K-modes Imbalance Classification Hybrid Approach (K-MICHA) as an ensemble learning methodology. In CoSKNN, our aim was to tackle the imbalance problem by using cosine similarity as a distance metric and by introducing a cost sensitive score for the classification using the KNN algorithm. We conducted a comparative validation experiment where we prove the effectiveness of CoSKNN in terms of accuracy and fraud detection. On the other hand, the aim of K-MICHA is to cluster similar data points in terms of the classifiers outputs. Then, calculating the fraud probabilities in the obtained clusters in order to use them for detecting frauds of new transactions. This approach can be used to the detection of any type of financial fraud, where labelled data are available.

At the end, we applied K-MICHA to a credit card, mobile payment and auto insurance fraud data sets. In all three case studies, we compare K-MICHA with stacking using voting, weighted voting, logistic regression and CART. We also compared with Adaboost and random forest. We prove the efficiency of K-MICHA based on these experiments. We also implemented K-MICHA in a big data framework using H2O and R. We were able to process and analyse bigger data sets in a very short period of time.

Keywords: Financial fraud, Class imbalance, F_1 – score, Cost Sensitive Classification, Cosine similarity, K-Nearest Neighbors, Ensemble learning, K-modes.

Résumé

Différents types de risques existent dans le domaine financier, tels que le financement du terrorisme, le blanchiment d'argent, la fraude de cartes de crédit, la fraude d'assurance, les risques de crédit, *etc.* Tout type de fraude peut entraîner des conséquences catastrophiques pour des entités telles que les banques ou les compagnies d'assurances. Ces risques financiers sont généralement détectés à l'aide des algorithmes de classification.

Dans les problèmes de classification, la distribution asymétrique des classes, également connue sous le nom de déséquilibre de classe (*class imbalance*), est un défi très commun pour la détection des fraudes. Des approches spéciales d'exploration de données sont utilisées avec les algorithmes de classification traditionnels pour résoudre ce problème.

Le problème de classes déséquilibrées se produit lorsque l'une des classes dans les données a beaucoup plus d'observations que l'autre classe. Ce problème est plus vulnérable lorsque l'on considère dans le contexte des données massives (Big Data). Les données qui sont utilisées pour construire les modèles contiennent une très petite partie de groupe minoritaire qu'on considère positifs par rapport à la classe majoritaire connue sous le nom de négatifs. Dans la plupart des cas, il est plus délicat et crucial de classer correctement le groupe minoritaire plutôt que l'autre groupe, comme la détection de la fraude, le diagnostic d'une maladie, etc. Dans ces exemples, la fraude et la maladie sont les groupes minoritaires et il est plus délicat de détecter un cas de fraude en raison de ses conséquences dangereuses qu'une situation normale. Ces proportions de classes dans les données rendent très difficile à l'algorithme d'apprentissage automatique d'apprendre les caractéristiques et les modèles du groupe minoritaire. Ces algorithmes seront biaisés vers le groupe majoritaire en raison de leurs nombreux exemples dans l'ensemble de données et apprendront à les classer beaucoup plus rapidement que l'autre groupe.

Après avoir mené une étude approfondie pour examiner les défis rencontrés dans le cas de déséquilibre des classes, nous avons constaté que nous ne pouvons toujours pas atteindre une sensibilité acceptable (c.à.d. une bonne classification du groupe

minoritaire) sans une diminution significative du taux total de classification correcte. Cela conduit à un autre défi qui est le choix des mesures de performance utilisées pour valider les modèles. Le taux total de classification correcte ou la sensibilité seuls ne sont pas suffisants. Nous utilisons d'autres mesures comme la courbe de Precision-Recall ou le score F1 pour évaluer ce compromis entre précision et sensibilité.

Dans ce travail, nous avons développé deux approches : Une première approche ou classifieur unique basée sur les k plus proches voisins et utilise le cosinus comme mesure de similarité (*Cost Sensitive Cosine Similarity K-Nearest Neighbors :* CoSKNN) et une deuxième approche ou approche hybride qui combine plusieurs classifieurs uniques et fondu sur l'algorithme k-modes (*K-modes Imbalanced Classification Hybrid Approach :* K-MICHA). Dans l'algorithme CoSKNN, notre objectif était de résoudre le problème du déséquilibre en utilisant la mesure de cosinus et en introduisant un score sensible au coût pour la classification basée sur l'algorithme de KNN. Nous avons mené une expérience de validation comparative au cours de laquelle nous avons prouvé l'efficacité de CoSKNN en termes de taux de classification correcte et de détection des fraudes. D'autre part, K-MICHA a pour objectif de regrouper des points de données similaires en termes des résultats de classifieurs. Ensuite, calculez les probabilités de fraude dans les groupes obtenus afin de les utiliser pour détecter les fraudes de nouvelles observations. Cette approche peut être utilisée pour détecter tout type de fraude financière, lorsque des données étiquetées sont disponibles.

La méthode K-MICHA est appliquée dans 3 cas: données concernant la fraude par carte de crédit, paiement mobile et assurance automobile. Dans les trois études de cas, nous comparons K-MICHA au stacking en utilisant le vote, le vote pondéré, la régression logistique et l'algorithme CART. Nous avons également comparé avec Adaboost et la forêt aléatoire. Nous prouvons l'efficacité de K-MICHA sur la base de ces expériences. Nous avons également appliqué K-MICHA dans un cadre Big Data en utilisant H2O et R. Nous avons pu traiter et analyser des ensembles de données plus volumineux en très peu de temps.

Mots-clés: fraude financière, déséquilibre de classe, score F1, classification sensible aux coûts, mesure de cosinus, K plus proche voisins, Apprentissage ensembliste, k-modes.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

AANN	Auto-Associative Neural Network
Adaboost	Adaptive Boosting
AIS	Artificial Immune System
ANN	Artificial Neural Network
AUC	Area Under the Curve
AUPRC	Area Under the Precision-Recall Curve
BNN	Bayesian Belief Network
C4.5	Decision Tree Algorithm
C5.0	Advanced version of the C4.5 Decision Tree Algorithm
CART	Classification And Regression Tree
CHAID	Chi-square Automatic Interaction Detector
CKNN	Simple voting K-Nearest Neighbor using cosine similarity
CoS	Cosine Similarity
CoSKNN	Cost-Sensitive Cosine Similarity K-Nearest Neighbor
cp	Complexity Parameter
CS	Cost Sensitive
DCKNN	Distance weighted K-Nearest Neighbor using cosine similarity
DEucKNN	Distance weighted K-Nearest Neighbor using euclidean distance
ESN	Electronic Serial Number
EucKNN	Simple voting K-Nearest Neighbor using the euclidean distance
FN	False Negative
FP	False Positive
FPR	False Positive Rate
IT	Information Technology
K-MICHA	K-modes Imbalance Classification Hybrid Approach
KNN	K-Nearest Neighbor

LR	Logistic Regression
MAE	Mean Absolute Error
NB	Naïve Bayes
NSA	Negative Selection Algorithm
OCC	One Class Classification
OCCoS	Cosine Similarity based One Class Classification approach
PR	Precision-Recall
RF	Random Forest
RO	Random Oversampling
ROC	Receiver Operating Characteristic
SVM	Support Vector Machine
TN	True Negative
TP	True Positive
TPR	True Positive Rate
VIF	Variance Inflation Factor

Glossary of Terms

Accuracy	The percentage of correctly classified observations from both classes.
Class imbalance	Skewed distribution of the data where one class is dominant and present much more than the other class.
Cost Sensitive	An imbalanced classification approach where higher weights are assigned to the minority class.
Ensemble Learning	An approach developed by building one model using either several samples of the data or several algorithms.
Explanatory Variables	The variables that are used as predictors for the machine learning algorithms.
Fraud	A criminal deception deliberately practiced by a person in order to gain a financial profit illegally.
H2O	An open source distributed in-memory prediction platform for big data science.
In-memory computing	The storage and process of information in the main random access memory (RAM) rather than in relational databases operating on slow disk drives.
Mode	The most frequently occurring value of a variable.
Negative	The majority class (legitimate or non fraud cases).
False Negative	The number of actual positives wrongly classified as negatives.
True Negative	The number of correctly classified observations from the majority class.
Observation	Data set row.
One Class Classification	An approach that uses data from one class only (usually the minority class) and learns its characteristics.
Positive	The minority class (fraud observations).

False Positive	The number of actual negatives wrongly classified as positives.
True Positive	The number of correctly classified observations from the minority class.
PR Curve	Precision-Recall curve: A graph representing the precision over the recall.
Precision	The ratio of examples classified as positive that are truly positives.
Random Oversampling	A Technique used to balance the classes by simply replicating observations as needed until balance between classes is reached.
ROC	Receiver Operating Characteristic curve: A graph representing the TPR over the FPR.
Sensitivity - True Positive Rate - Recall	The proportion of positives correctly classified as positives.
Target Variable	The variable to be predicted representing the fraud.
Variance Inflation Factor	Multicolinearity test for explanatory variables. A VIF higher than 10 means that multicolinearity is present between the variables.

List of Publications

The following is a list of publications containing the work presented in this thesis:

1) Sara Makki, Rafiqul Haque, Yehia Taher, Zainab Assaghir, Gregory Ditzler, Mohand-Saïd Hacid and Hassan Zeineddine, ***"Fraud Analysis Approaches in the Age of Big Data – A review of the State of the Art"***, in proceedings of the 2nd IEEE International Workshops on Foundations and Applications of Self* Systems (FAS*): 243 - 250, 2017.

2) Sara Makki, Rafiqul Haque, Yehia Taher, Zainab Assaghir, Gregory Ditzler, Mohand-Saïd Hacid and Hassan Zeineddine, ***"Fraud Data Analytics Tools and Techniques in Big Data Era"***, in proceedings of the International Conference on Cloud and Autonomic Computing (ICCAC): 186-187, 2017.

3) Sara Makki, Rafiqul Haque, Yehia Taher, Zainab Assaghir, Mohand-Said Hacid, Hassan Zeineddine, ***"A Cost-Sensitive Cosine Similarity K-Nearest Neighbor for Credit Card Fraud Detection"***, in the 1^{st} International Conference on Big Data and Cybersecurity Intelligence (BDCSIntell): 42-47, 2018.

4) Sara Makki, Zainab Assaghir, Yehia Taher, Rafiqul Haque, Mohand-Said Hacid, Hassan Zeineddine, ***"An Experimental Study with Imbalanced Classification Approaches for Credit Card Fraud Detection"*** - IEEE Access 7, 93010-93022 2019.

5) Sara Makki, Zainab Assaghir, Yehia Taher, Rafiqul Haque, Mohand-Said Hacid, Hassan Zeineddine, ***"K-MICHA: K-modes Imbalance Classification Hybrid Approach for Financial Fraud Detection"*** - (In progress 2019).

Chapter 1

Introduction

1.1 Background

Fraud is a *criminal deception* deliberately practiced by a person to gain a financial profit illegally. It can also occur solely to deceive another person or entity like providing false statements. Fraud is not a recent phenomenon unique to modern society. The fraudsters have been practicing fraudulent activities for more than decades [3]. Many factors can facilitate fraudulent activities such as help from bank's employees especially to access the bank's Information Technology (IT) systems, client's database, and personal information. This is better known as *Internal Fraud* which usually occurs when internal personnel of

an organization such as employee, manager, or executive commits fraud against his or her employer. Even if no financial profit was gained, the act of violating the privacy of the bank's clients is considered a fraud itself. Online and mobile banking services also facilitate fraud activity, giving more opportunities to fraudsters to access critical systems. In contrast, *External Fraud* is committed by external third-parties. For instance, stealing proprietary information of an organization by the supplier or other stakeholders.

In [4], the authors characterized the multifaceted phenomenon fraud include the following: uncommon, imperceptibly concealed, time-evolving and often carefully organized crime which appears in many types of forms. These characteristics are summarized in [3]:

- *Uncommon*: Independent of the exact setting or application, only a minority of the involved population of cases typically concerns fraud, of which furthermore only a limited number will be known to concern fraud. Also, the fraudsters mask fraudulent activities with the non-fraudulent ones – which is another cause of fraud being uncommon.
- *Imperceptibly Concealed Behaviour*: The behavior of fraudsters is not distinguishable from the others. They keep the behavior normal to go unnoticed and to remain covered by non-fraudsters. This effectively makes fraud *imperceptibly concealed*, since fraudsters do succeed in hiding by well considering and planning how to precisely commit fraud.
- *Time Evolving*: The fraudsters continuously change their method because their objective is to remain undetected as much as possible. This means the techniques and tricks the fraudsters use evolve in time along with or better ahead of fraud detection mechanisms.
- *Organized and barely isolated*: The fraudsters organize fraudulent activities very carefully like other organized crimes. Very often they do not operate independently rather they associate with the others. Some forms of fraud involve complex structures that are set up to commit fraud in an organized manner. This makes fraud not to be an isolated event.

These characteristics foster an enormous challenge to the industries mainly the financial industry to detect fraudulent activities such as fraudulent transactions made through a credit card or a check. Furthermore, Fraudulent activities differ depending on sectors, methods, severity, complexity, and difficulty of detection or prevention. As previously mentioned, fraud is not limited to attacks on banks. In some cases, banks or financial institutions are the fraudsters themselves like in financial statements fraud cases,

or sometimes in money laundering cases. Insurance companies, telecommunication companies, or investors are sometimes victims of fraud. There is an exhaustive list of frauds described in a large body of literature such as [5] and [6]. Table 1.1 summarizes a non-exhaustive and refined list depicting different types of frauds.

Table 1.1 Different types of fraud

Fraud Type	**Description**
Credit Card Fraud	It is defined as unauthorized use of a credit card account. It occurs when the cardholder and the card issuer are not aware that the card is being used by a third party. Therefore, fraudsters can obtain goods without paying, or gain illegal access to funds from an account [7].
Insurance Fraud	It can be described as an attempt to misuse or take advantage of an insurance policy. Insurance is made to cover losses and to protect against risks. Fraud occurs when the insured use the insurance contract as a tool to gain illegal profit [8].
Money Laundering	Money laundering is the scheme in which criminals try to disguise the source, and destination of money gained through illegal activities, intending to make it seem legitimate [9].
Telecommunication Fraud	In the telecommunication area, fraud is characterized by the abuse of any carrier services without the intention of paying. There could be other motivations like political or personal motivations, *etc.*
Financial Statement Fraud	Financial statement fraud is also known as *accounting fraud* is defined as intentional misstatements of financial statements to mislead the reader especially investors and creditors to create a false impression of an organization's financial strength [10].
Securities fraud	It is also known as financial markets fraud or investment fraud refers to deceptive practices in connection with the offer and sale of securities [11]. *High yield investment fraud* is a very common type of securities fraud. Very known examples are Pyramid schemes, Ponzi schemes, affinity fraud, *etc.*

In addition to the above, there are many other types of fraud including *counterfeit*, *product warranty fraud*, *click fraud*, *identify theft*, *tax evasion*, etc. Each of these fraud types has distinctive characteristics and consequence which is usually catastrophic. Therefore, *fraud detection* and *fraud prevention* are the most critical components of an effective strategy of fighting against fraudsters. Fraud detection refers to the ability to recognize or discover fraudulent activities, whereas fraud prevention refers to measures that can be taken to avoid or reduce fraud [12].

Expert-Based Fraud Detection

The most widely used approach for detecting fraud is called *Expert-based Approach*. It is a classical approach built on the experience, intuition, and business or domain knowledge of the fraud analyst [3]. An expert-based approach typically involves a manual investigation of a suspicious case, which may have been signaled, for instance, by a customer complaining of being charged for transactions he did not do. Such a disputed transaction may indicate a new fraud mechanism to have been discovered or developed by fraudsters, and therefore requires a detailed investigation for the organization to understand and subsequently address the new mechanism [12].

Data-driven Fraud Detection

Over the last ten years, the world has experienced an unprecedented growth of data. More than 2.5 exabytes data are generated every day which will accumulate to 175 Zettabyte data by 2025 [13]. According to IBM, 90% of the data in the world today has been created in the last eight years [14]. These data stem from everywhere: sensors, social media sites, digital images and videos, and purchase transactions, cell phones, GPS signals to name a few. These data have been characterized into the following: massive-scale, fast-moving, and diverse which are better known as *volume*, *velocity*, and *variety* respectively. These characteristics have given the rise to the notion of *Big Data* and data-driven applications especially the analytics which leverages the profound power of data to extract intelligence and make a better decision with high precision.

Although classic expert-based fraud-detection approaches are still in widespread use and represent a good starting point and complementary tool for an organization to develop an effective fraud detection and prevention system, a shift is taking place toward *data-driven fraud analytics* [3]. Today, data-driven analytics is the *de facto* solutions for all data-intensive industries including banking. The banking industry is exploring the opportunities and challenges of different types of analytics including Fraud Analytics

that is empowered by extremely scalable and high-performance advanced Big Data technologies.

Fraud analytics has become the emerging tool of the twenty-first century for detecting anomalies, red flags, and patterns within voluminous amounts of data that is sometimes quite challenging to analyze [15]. The techniques of criminals and fraudsters and their shenanigans are savvier due to technology and the means they use to hide fraudulent activities [12]. While the extreme digital transformation has opened a multitude of interfaces that have increased the opportunities to commit fraud, at the same time it is also playing a key role in developing new methods to detect and prevent fraud [15]. In the past, the spreadsheet was the master of fraud analytics. However, data-driven fraud analytics has taken the industry by force—new strategies, machine learning and statistical techniques, and powerful Big Data tools.

The data-driven fraud analytics is a highly promising approach for three apparent reasons explained [3]. These reasons are summarized in the following:

- *Precision*: Most organizations only have a limited capacity to have fraud cases checked by an auditor to confirm whether or not a case effectively concerns fraud. This human-driven approach is risk-prone. In contrast, good quality data is vital to increase the precision and accuracy of an analysis. The goal of data-driven fraud analytics is to make the most optimal use of the limited available inspection capacity, or in other words to maximize the fraction of fraudulent cases among the inspected cases (and possibly, besides, the detected amount of fraud) [3]. A system with higher precision, that is potential to be delivered by data-based methodologies, directly translates in a higher fraction of fraudulent inspected cases.
- *Operational Efficiency*: Data-driven fraud analytics are built using advanced techniques from various domains including machine learning, statistics, mathematics, deep learning. Also, advanced tools such as in-memory analytics engine are used to implement the analytic techniques. This significantly increases operational efficiency which is critically important for many fraud scenarios. For instance, when evaluating a transaction with a credit card, an almost immediate decision is required concerning approve or block the transaction because of suspicion of fraud. This needs real-time analysis and decision making - which can be done with data-driven fraud analytics tools and technologies.
- *Cost Efficiency*: Developing and maintaining an effective and lean expert-based fraud-detection system is both challenging and labor-intensive. A more automated and, as

such, a more efficient approach to develop and maintain a fraud-detection system, as offered by data-driven methodologies, is a more cost-effective approach.

In light of the above discussion, it is evident that data-driven fraud analytics have several benefits over the traditional expert-based approach for fraud detection. However, various challenges in data-driven approaches must be tackled to strengthen the fraud detection system to fight against fraudsters. This research focuses on the extremely critical *class imbalance problem* data-driven fraud analytics.

1.2 Motivation

Different facets are the driving factors of this research. The *financial impact* of fraudulent activities is mostly *catastrophic*. A large number of organizations and individuals are victimized by fraudsters every day. Unfortunately, this trend is increasing every year. This, by extension, has a *socio-cultural impact*. Therefore, the financial impact of fraudulent activities is the key factor that drove this research initiative. Moreover, there are several *challenges* remained unsolved for building efficient fraud analytics. These challenges are critical barriers that must be solved to fight against the fraudsters. This is the other important factor that motivated this research.

This section describes the financial impact and challenges in detail to provide a comprehensive view of the significance of this research in the real world and the domain of science.

1.2.1 Financial Impact of Fraud

Fraudulent activities have disastrous financial consequences. They are costing the banking industry around 67 billion dollars per year, as estimated by the association of certified fraud examiners in 2014 [16]. In reality, it can also be higher because many cases were not referred to the external authorities and solved internally to avoid bad publicity [16]. According to the 2019 Identity Fraud Study from Javelin Strategy & Research, in the United States of America, the number of consumers who were victims of identity fraud fell to 14.4 million in 2018, down from a record high of 16.7 million in 2017. However, identity fraud victims in 2018 bore a heavier financial burden: 3.3 million people were responsible for some of the liability of the fraud committed against them, nearly three times as many as in 2016. Moreover, these victims' out-of-pocket fraud costs more than doubled from 2016 to 2018 to 1.7 billion Dollars [17]. In the United Kingdom, unauthorized financial fraud losses across payment cards, remote banking, and cheques

totaled £844.8 million in 2018, an increase of 16 percent compared to 2017; In addition to this, in 2018 UK Finance members reported 84,624 incidents of authorized push payment scams with gross losses of £354.3 million [18].

Furthermore, Fraudulent activities are increasing every day. A global survey was conducted by KPMG in 2018. The survey of KPMG found that 61 percent of respondents indicated that the total volume of external fraud had increased and 59 percent said the value had increased [1]. It was also found that the trend of most of the fraud types is increasing. Figure 1.1 shows the trend of different types of fraud in different geographical locations.

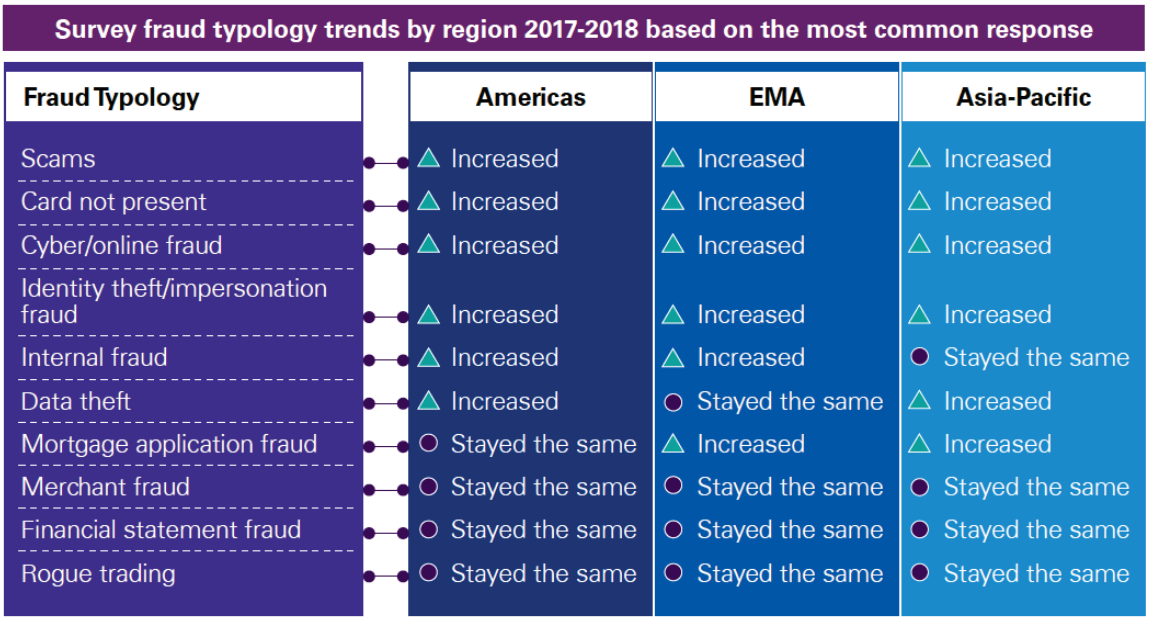


Fig. 1.1 The global trend of different types of frauds (Source [1])

Based on the above discussion, it is evident that current fraud detection systems have shortcomings which are allowing the fraudsters to continue their criminal acts. Also, these activities must be stopped or reduced to diminish the impact of fraud - which needs an efficient solution.

1.2.2 The Challenges of Fraud Detection

Although fraud-detection approaches have gained significant power over the past years by adopting potent statistically-based methodologies and by analyzing massive amounts of data to discover fraud patterns and mechanisms, still fraud remains enormously challenging to detect [3].

Researchers applied different methods and algorithms to detect fraud in financial or telecommunication sectors, or money laundering operations. However, we identified a few issues that make fraud detection more challenging and difficult. In the following, we briefly describe these issues.

- **Dynamic patterns of fraud** are highly challenging specifically for the systems that rely on supervised learning models. These systems can only detect fraud patterns based on the training data set that consists of patterns occurred in the past. However, the fraudsters never stop producing new fraudulent ways and strategies to overpower the systems. Such variation of fraudulent activities cannot be dealt with with the supervised learning-based system since the training data sets do not contain the new fraud patterns. This is frequently referred to learn new classes in machine learning [19].
- **Real-time fraud detection** is an ideal solution for financial institutes, because, real-time systems may save huge financial loss. However, detecting fraudulent activities in real-time fosters different challenges to the system. The three main challenges which we identified in our study include the followings: (i) the speed at which data flows today is difficult to process and analyze, (ii) the computational complexity of fraud analytics is huge, and (iii) building a high-performance algorithm for data-intensive applications is non-trivial.
- **Integrating large volume data with variety** introduces a huge challenge. One of the critical advantages of Big Data is that it enables the users to collect data from a wide variety of sources including financial sources and other resources such as hospital records, financial markets activities, messages between brokers, *etc.* These sources constitute a huge volume of data sets containing many attributes and records. Furthermore, these sources rely on different data models; consequently, data arriving from a variety of sources fosters integration challenges for the fraud analysis system.
- **Skewed class distribution** is one of the most crucial issues in fraud detection. Typically, the ratio of *fraud* : *legitimate* is very small in the data which are used to train the models [8]. These proportions are very challenging because they do not allow the learners to understand the dynamics of minority groups (fraud). This problem is more vulnerable to the classification algorithms that are implemented within a Big Data framework. Since Big Data technologies such as Hadoop breaks data into (*chunks*), the small portion of data in the samples becomes much smaller. Consequently, the characteristics of the minority class become harder to detect, the classification becomes more difficult, and in fact, the information is lost whereas the goal is to extract the maximum value from large-scale data.

- **Performance measure** is another challenge that is caused by skewed data distribution. The accuracy rate (correctly classified observations) is the one used for a typical classification problem. This measure may not be always appropriate for fraud detection [20]. For example, if the data contains 1% fraud observations, an accuracy rate below 99% is unacceptable. The reason is straightforward: the system can classify all records as legitimate and still give an accuracy rate of 99%. This leads to consider cost-sensitive performance measures that take into account the wrongly classified observations, without raising many false alarms, that would waste a huge amount of time and resources to be investigated. Such statistics, that are recommended to be considered in fraud detection analysis, including accuracy, F-score, sensitivity, Area Under the Curve (AUC), and Matthew's correlation coefficient. Moreover, Computational costs must also be considered.

The challenges explained in the above motivate to initiate this research project, as the problem lies in the heart of real-world scenarios, yet there are challenges that to be fixed.

1.3 Problem Description

Fraud is a longstanding problem for the industries. A broad spectrum of solutions are available however nothing could stop fraudulent activities and fraudsters. It is essentially a loop of actions between fraudsters and fraud detection experts. Fraudsters never stop developing new methods to commit fraud and the experts aim to detect every possible method for fraudulent activities. There are several challenging issues — that I identified through literature – discussed in the previous section. Based on my study, I believe that the issues previously mentioned must be solved to develop effective and efficient fraud detection systems.

Fraud analysis is of critical importance to the banking sector – as well as many others – since fraudulent activities are becoming a more frequent occurrence that leads to a disastrous impact on the organizations, society, and individuals. In our work, we focused on the skewed data distribution also known as class imbalance problem.

1.3.1 Class Imbalance Problem

The class imbalance problem is one of the most critical challenges of all. This problem is defined as having an extremely imbalanced and highly skewed distribution of the data [21]. In other words, the ratio of fraudulent or criminal activities is considerably smaller than the legitimate and genuine ones. Figure 1.2 illustrates a class imbalance problem

found in one of our experiments where the imbalance ratio is 5.96%. Figure 1.2 shows an imbalance situation in our data set according to two features. The red dots in the figure represent the fraud cases that have a much lower frequency than the other cases.

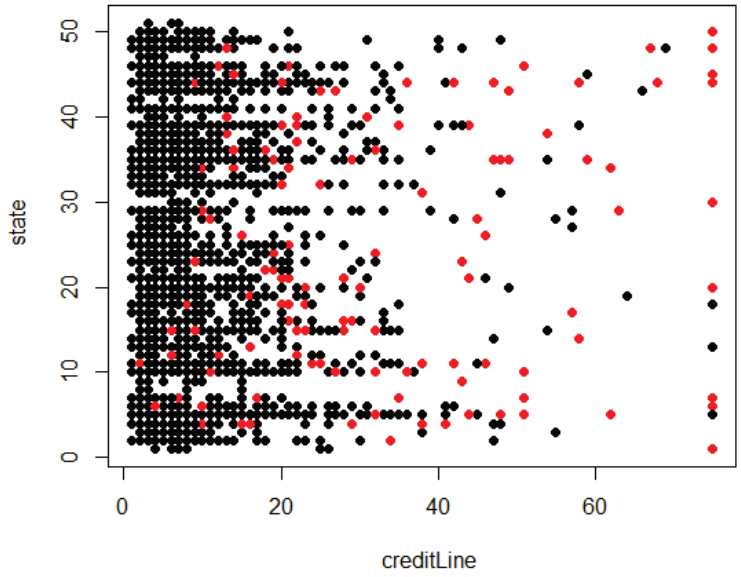


Fig. 1.2 An example of class imbalance situation

Class imbalance promotes a huge challenge in detecting the characteristics of fraudulent activities and extracting fraud patterns. Due to the dominance of one class, most of the optimization steps (concerning accuracy) performed by the classification algorithm, aim to correctly classify the dominant class while ignoring the others. These minority observations like the fraud observations are the most critical to be classified correctly. If the classification algorithm is unable to detect fraud patterns, the illegal transactions are considered as legal, thus causing severe financial damage to individuals and organizations.

Many research studies have been dedicated to the imbalanced classification problem. Several solutions have been proposed in the literature (*e.g.*, [22–24]) which, to the best of our knowledge, are built on machine learning and data mining algorithms. However, class imbalance remained an unsolved issue [25, 26]. An experimental study that we conducted [27], revealed that the approaches normally used to solve imbalance problems may have unpleasant consequences. These approaches improve sensitivity yet the improvement leads to an increase in the number of false alarm. The problem is summarized as follows: using imbalanced classification approaches, the number of false alarms generated is higher than the number of frauds that are more detected. The

results of this experimental study greatly motivated us to explore the other methods that focus on detecting the hidden patterns of fraud, with a minimum misclassification rate.

1.3.2 Selection of Performance Measures

Typically, **accuracy** is the most common performance measure in a classification problem. However, in our case accuracy is not adequate because we are tackling imbalanced classification and using it alone as a measure of performance is misleading.

The fraud percentage in our case studies range between 5% and 12% of total cases. This means that a classification's accuracy rate less than 95% or 88% respectively is not acceptable, simply because a random classifier is capable of achieving high accuracy in an imbalance classification case. Fraud detection may not be achievable by gaining a high accuracy rate. Therefore, we will consider other performance measures, specifically the **sensitivity**, **Area Under the Precision-Recall Curve** (AURPC) and the F_1 **score**. We provide details of our selection process.

To evaluate the methods presented in this thesis, confusion in the matrix of the form presented in Table 1.2 is calculated using test sets by comparing the prediction of the method with the actual value.

Table 1.2 Form of confusion matrix

Predicted	**Actual**	
	Legitimate (0)	**Fraud (1)**
Legitimate (0)	True Negative (TN)	False Negative (FN)
Fraud (1)	False Positive (FP)	True Positive (TP)

- TN: represents the number of correctly classified observations from the majority class.
- TP: represents the number of correctly classified observations from the minority class.
- FN: represents the number of actual positives wrongly classified as negatives.
- FP: represents the number of actual negatives wrongly classified as positives.

The accuracy rate represents the percentage of correctly classified observations from both classes:

$$\text{Accuracy in percentage} = \frac{TP+TN}{TN+FN+TP+FP} \times 100$$

According to this formula presented above, we remark that in the case of imbalance, the accuracy is biased towards the majority group, specifically the TN.

The sensitivity which is also known as True Positive Rate (TPR) and recall, represents the proportion of positives correctly classified as positives. This parameter is critically important and will be considered as a performance measure along with accuracy. This is defined as follows:

$$\text{Sensitivity} = \frac{TP}{TP+FN}$$

Note that, considering the sensitivity alone is also misleading, it allows to ignore a large number of false positives. We aim to find a balance between those two parameters. We need to obtain a high fraud detection rate (sensitivity), with the highest possible accuracy. To handle this issue, we consider "trade-off" measures like the AUPRC and the F_1 score.

In a typical classification problem, the Receiver Operating Characteristic (ROC) curve is commonly used as a performance measure. It is essentially a graph that presents the diagnostic ability of a binary classifier system as its discrimination threshold is varing[1]. It is created by plotting the TPR over the False Positive Rate (FPR). However, in imbalanced classification, this curve can mask poor performance. To the best of our understanding, the Precision-Recall (PR) curve is relatively a better measure because it is more sensitive to the class imbalance than the ROC curve [20]. This curve is defined by plotting precision rate over the recall rate. The use of precision instead of FPR which is used in ROC curve allows capturing the effect of large negative observations on the algorithm's performance. The precision represents the ratio of examples classified as positive that are true positives. These two measures are defined as:

$$\text{Precision} = \frac{TP}{TP+FP} \quad \text{and} \quad FPR = \frac{FP}{TN+FP}$$

In the PR curve, the quality of the model is determined by the proximity of the curve to the upper-right-hand corner. The closer the curve is to the upper-right-hand corner the better the model is. This can be measured with the AUPRC. Moreover, it is not always straightforward to find the class probabilities, so we will also use sometimes the F_1 score. The higher this score is better. F_1 score is defined as follows:

$$F_1 \text{ score } = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

[1] http://www.ashukumar27.io/roc-auc/

It is worth noting that each of these performance measures is interpreted differently. Also, each of these measures cannot be used alone to confirm the competitive quality of the methods.

1.4 Research Goal & Objectives

Practically speaking, eliminating fraud wholly might not possible by using technologies because there are issues (e.g., cross-border access to the financial system) that could not be dealt with technology; these issues need strategic and political directives. In this thesis, we aim to tackle the technological aspect.

In fraud analysis, efficiency remains an important issue that should be focused on to achieve a high fraud detection rate. Efficiency guarantees the performance of the fraud detection models, even an increase of 1% accuracy rate is crucial because it will have a huge impact in detecting fraudulent activities and fraudsters.

The goal of this thesis is to *design and develop a fraud analytics framework that deals with the most critical yet unsolved class imbalance problem to enhance the efficiency of the fraud analytics systems.*

This goal is decomposed into three objectives that are listed in the following:

- **Objective 1**: Design and develop a solution that enables to analyze fraud labeled data set collected from banks and financial institutions from different sectors.
- **Objective 2**: Design and develop a solution that enables to explore the hidden patterns of fraud by learning from fraud examples, using supervised cost sensitive machine learning algorithms.
- **Objective 3**: Design and develop a solution that enables to find a trade-off between the accuracy and sensitivity of the model. In other words, achieving the highest fraud detection rate with minimal false alarms.

1.5 Contributions

There are different contributions made in this thesis– are categorized into primary and secondary contribution. We present these contributions in this section.

Primary Contributions

In this thesis developed two approaches: a Cost-Sensitive Cosine Similarity K-Nearest Neighbor (CoSKNN) as a single classifier, and a K-modes Imbalance Classification Hybrid Approach (K-MICHA) as an ensemble learning methodology. Our contributions can be summarized as follows:

- **CoSKNN: Cost-Sensitive Cosine Similarity K-Nearest Neighbor**:
  We introduced a Cost-Sensitive KNN approach detailed in Chapter 3. We aimed to tackle the imbalance problem by using cosine similarity as a distance metric and by introducing a score for the classification. To improve the method's performance in terms of imbalance, we also studied the choice of the score's thresholds and the number of neighbors to consider. We conducted a validation experiment where we compare the performance of simple voting KNN using both euclidean distance and cosine similarity, a distance weighted KNN also using both euclidean distance and cosine similarity, a cost-sensitive KNN approach that was introduced by Qin et al. [28], a decision tree approach, a one-class classification Support Vector Machine (SVM) and CoSKNN. The comparison was done by applying these methods to an example of credit card fraud data set with imbalance, using multiple performance measures, mostly relying on AUPRC and F_1 score. This experiment shows that CoSKNN is outperforming all the other methods. (See Chapter 3 for more details.)

- **K-MICHA: K-modes Imbalance Classification Hybrid Approach**:
  We introduced a hybrid diversification approach using the k-modes clustering algorithm detailed in Chapter 4. We also applied it to (1) a credit card, (2) mobile payment and (3) auto insurance fraud data sets that we present in Chapter 5. K-MICHA aims to cluster similar data points in terms of the outputs of the classifiers. Then, calculating the fraud probabilities in the obtained clusters to use them for detecting frauds of new transactions. For case study 1, the credit card fraud detection, we combined Logistic Regression (LR), Artificial Neural Network (ANN), a cosine similarity cost-sensitive K-nearest neighbor approach (CosKNN) that we developed and a one-class classification approach also using cosine similarity (OCCoS). For case study 2, the mobile payment transactions fraud data set, we combined the two approaches based on cosine similarity previously mentioned OCCoS and CoSKNN with LR, decision tree (CART) and Naïve Bayes classifier (NB). In case study 3, the auto insurance fraud case, we combined five algorithms, CosKNN, LR, CART, NB, and ANN. In all three case studies, we compare K-MICHA with stacking using voting, weighted voting, logistic

regression, and CART. We also compared it with Adaboost and random forest. Even though the last two algorithms only use one classifier and they do not consist of combining multiple algorithms, we are interested in comparing their performance with our combination approach. In the end, based on the experiments, we prove the efficiency of K-MICHA.

Secondary Contributions

In addition to the main contributions discussed in the above, I conducted an experimental study and a survey of literature concerning fraud detecting solution. These are essentially strongly connected (or related) with the primary contribution. The studies are briefly explained below:

- **Survey**: A rigorous survey of the literature was conducted mainly to identify the strength and shortcomings of existing solutions. The survey covered all classical and advanced approaches such as data-driven approaches and produced a comprehensive report on fraud analytics approaches. A paper of this survey was published in 2017. Chapter 2 presents the results of the survey.
- **Experimental Study**: The literature review was not sufficient to verify the efficiency of the existing technologies. Hence, an experimental study was conducted with the key technologies including support vector machine (SVM), C5.0, Artificial Neural Network (ANN) that are widely used in today's advanced fraud detection system. This experimental study was very effective in understanding the genuine weaknesses of existing solutions concerning class imbalance issues. The results of this experimental study were included in an article published in 2019. This experimental study is presented in Chapter 2.

1.6 Research Scope

The primary scope of this research is three-fold: *data processing* and *predictive analytic*, and *data visualization*. These scopes are briefly presented in the following.

- *Data processing*: Data processing is a common task and the most critical tasks in any type of data analytics. Data processing techniques are applied to produce a high-quality dataset that is vital to extract meaningful intelligence from data. This research focuses on techniques for fraud data cleaning, pre-processing, outliers detection, and oversampling or undersampling. The objective was to explore the existing techniques and identify the best-suited ones for financial data processing.

- *Data Analysis*: This is the main focus of this thesis. The main target was to *discover*, *design*, and *develop* a solution for *prediction* and *classification* of observations. The machine learning algorithms were heavily investigated, and used to achieve the goal of this thesis.
- Data Visualization: Data visualization has two facets: (i) data are visualized during the descriptive or exploratory study to have a comprehensive visual representation and a better understanding of data. Also, the final results are visualized, specifically in the combination algorithm K-MICHA to relate to the original datasets used to interpret the results. In this research, I explored the visualization patterns and used the most suitable ones to visualize the data and results.

Anything outside of the above scope (yet concerns fraud analytics) has not been covered in this thesis. Some technologies, for instance, data storage technologies are used in this thesis; however, such technologies are considered outside of the scope of this research because there was no contribution concerning the storage. Neither scalability nor performance of the storage were investigated. Furthermore, anything related to data security is outside of the scope of this research.

1.7 Thesis Structure

This thesis consists of five chapters apart from the introductory Chapter 1, where we describe the general context of financial fraud and our research problem, the class imbalance.

In Chapter 2, we present studies that were conducted for financial fraud detection. We list them according to the different financial sectors (credit card, telecommunication, money laundering, *etc.*), and different technologies or strategies like real-time detection, big data analytics and ensemble learning. In this Chapter, we also present an experimental study [27], where we investigate the exciting solutions of class imbalance, and we investigate their limitations.

Chapter 3 presents our first contribution, the Cost Sensitive Cosine Similarity K-Nearest Neighbor approach (CoSKNN) as a single classifier. We provide the approach theory, along with the different steps of the model. Moreover, we discuss the results of the validation experiment we conducted to prove the model's efficiency.

In Chapter 4, we introduce K-MICHA, a hybrid approach that we develop using k-modes clustering algorithm with the aim to evaluate fraud probabilities. This approach is an ensemble learning technique based on clustering similar data points in terms of the

classifiers outputs, then calculating fraud probabilities in each cluster. Different examples of ensemble learning are presented in this chapter, along with the implementation of K-MICHA.

In Chapter 5, we present the results of the application of K-MICHA in the financial sector. Three case studies concerning credit card fraud, mobile payment fraud and insurance fraud are used. The purpose of these studies is to prove the efficiency of K-MICHA by comparing it to other single classifiers and to other ensemble learning methods. We also provide the results of the implementation of K-MICHA in H2O and R in this chapter.

Finally, we provide a conclusion for the work done in this thesis, and gives perspectives for further research and work.

Chapter 2

Literature Review

2.1 Introduction

The automated fraud detection systems have gained enormous popularity especially within financial institutions. These systems analyse complex events or actions over

historical facts (*i.e.,* data) and discover fraud patterns. The analysis of fraud is a process consisting of a sequence of functions to predict or discover potential or explicit threats of fraudulent activities. The process relies on techniques from a wide variety of areas including *data mining*, *statistics*, *machine learning* etc. The efficacy of a fraud detection system largely depends on the efficiency of the used techniques and relevant data.

Over the years, fraud analysis techniques underwent rigorous development. However, lately, the advent of Big data led to the vigorous advancement of these techniques since Big Data resulted in extensive opportunities to combat financial frauds. Given the massive amount of data that investigators need to examine, to find fraudulent patterns, Big Data analysis techniques are of critical importance. Unfortunately, conventional techniques or tools are not adequate to perform fraud analysis over Big Data. Thus, in the last few years, several advanced techniques have been proposed in the literature. We report in this chapter some of these techniques, to point out their strengths and weaknesses. To do that, we conducted a study to evaluate and understand the fraud detection mechanism using some of the available techniques.

2.2 Fraud Detection Technologies

Fraud detection is a very challenging and crucial topic that is the center of many researches and studies. In the following, we will present work that has been done so far, according to the different financial sectors, and different technologies or strategies like real-time or ensemble learning.

2.2.1 Credit Card Fraud Analysis Approaches

Credit card databases contain information about card transactions like account number, type of card, kind of purchase, location and time of the transaction, client's name, merchant code, size of the transaction, *etc.* This information was used by researchers as attributes to determine whether the transaction is fraudulent or legitimate, or to detect outliers that merit investigation. Bolton and Hand [29] proposed two clustering techniques: *peer group analysis* and *break-point analysis* to detect the behavioral fraud. These methods allowed us to catch suspicious behavior indicating a fraudulent transaction. Weston et al. [30] used peer group analysis on real credit card transaction data to find outliers and suspicious transactions.

Ramakalyani and Umadevi [31] also applied genetic programming to detect fraudulent card transactions. Bentley et al. [32] presented a fuzzy Darwinian detection system based on genetic programming to produce fuzzy logic rules.

Srivastava et al. [33] modeled the sequence of credit card transaction using a hidden Markov model initially trained with the normal behavior of cardholder, and showed how it can be used for the detection of frauds. Other studies were conducted by Esakkiraj and Chidambaram [34] and Mishra et al. [35] with the same purpose. Artificial immune systems were investigated by Brabazon et al. [36] and Wong et al. [37] for the detection of fraudulent transactions, imitating the immune system's ability to distinguish between *self* and *non-self*.

Sanchez et al. [38] used association rules to extract knowledge about fraudulent and unlawful card transactions. Sahin et al. [39] proposed a cost-sensitive decision tree approach for fraud detection, and this was also investigated by Bahnsen et al. [40]. Pasarica [41] suggested the use of support vector machine classification with the Gaussian kernel function and found it the best approach to detect the fraud patterns. Sahin and Duman [42] compared in their study decision trees (using three algorithms CART, C5.0, and CHAID) with support vector machines (with linear, sigmoid, polynomial, and radial kernel functions) and concluded that decision trees (especially CART algorithm) outperform support vector machines methods. Ganji and Mannem [43] proposed detecting credit card fraud by using a data stream outlier detection algorithm which is based on reverse k-nearest neighbors.

Several studies focused on neural network applications to credit card fraud detection, [44–46]. 29 Syeda et al. [47] proposed using fuzzy neural networks on parallel machines with the purpose to speed up rule production for customer-specific credit card fraud detection. Maes et al. [48] compared artificial neural network and Bayesian belief networks on real-world financial data, and they showed that bayesian belief networks detect 8% more of fraudulent transactions than an artificial neural network.

Whitrow et al. [49] considered transaction aggregation and proved that it is effective. They concluded that random forest performs better than other methods such as support vector machines, logistic regression, and K-nearest neighbors. Bhattacharyya et al. [7] conducted a comparative study between logistic regression, support vector machines and random forest, using different performance measures, they concluded that random forest outperformed both other methods. Subashini and Chitra [50] compared five models: decision trees (CART and C5.0), support vector machines with polynomial kernels,

logistic regression and Bayesian belief network, and concluded that CART performs better than other methods.

In a recent study, Mahmoudi and Duman [51] proposed a modified Fisher discriminant function, using simple linear discriminant analysis for credit card detection for the first time, and modified it to be more sensitive to false negatives.

2.2.2 Telecommunication Fraud Analysis Approaches

The first detection systems focused on the cloning problem and tried detecting this type of fraud, by *collision detection*, that consists of monitoring calls and catching temporarily overlapping calls, or *velocity checking* by monitoring calls made from very distant locations in a very short period. These two methods are useful for a certain level of usage by subscribers, fraudulent activities with low-usage legitimate subscribers will not be detected. Another method called *dialed digits analysis* consists of creating databases of telephone numbers dialed by bandits so that it can be used to match numbers dialed by customers and determine whether the call is fraudulent or legitimate [52].

Other methods were developed using the call data containing a great deal of information, like the date and time of call, duration, origin, destination, number dialed, *etc.* Fawcett and Provost [52] used a rule-learning algorithm and created user profiles to extract indicators of a fraudulent activity, then combined these profiles to generalize the rules. Moreau et al. [53] investigated three approaches: rule-based approach, and two neural networks considering both supervised and unsupervised learning. Taniguchi et al. [54] proposed using a supervised neural network and Bayesian belief networks on labeled data and concluded that they perform better than unsupervised methods. Murad and Pinkas [55] used a clustering algorithm to create profiles of normal behavior from each legitime account. Then, they set a threshold according to certain criteria like call duration or destination, and an alert is raised if this threshold is exceeded. Rosset et al. [56] presented a user profiling technique, first, rules are extracted using a C4.5 algorithm, and then these rules are sorted according to their ability to detect fraud. Cox et al. [57] used visual data mining techniques and graphical representation of data sets to detect fraud. Cahill et al. [58] used daily updated account signatures to distinguish fraud activities from the legitimate ones, using supervised algorithms.

Cortes et al. [59, 60] used link analysis and presented large dynamic graphs made up of subgraphs called *Communities of Interest*, linking fraudsters and their activities together; and new cases of fraud in the network were detected as *guilty by association.* Estevez et al. [61] proposed a system that detects subscription fraud, using fuzzy rules to classify

customers into four different categories: subscription fraudulent, otherwise fraudulent, insolvent and normal. Then a multilayer perceptron neural network was applied to the data to predict fraud. An application of decision trees to identify fraud from legal use was presented by Hilas and Sahalos [62].

2.2.3 Financial Statements Fraud Analysis Approaches

Several criteria can serve as indicators of accounting fraud like weak internal control system, rapid company growth, inconsistent relative profitability, company ownership status (private/public), aggressive management attitude towards financial reporting, financial ratios, *etc.* Persons [63] used the logistic model to detect financial reporting fraud. Beasley [64] proposed a logit regression analysis to predict financial statement fraud. Hansen et al. [65] used probit and logit techniques to predict fraud. Spathis [66] developed a model using logistic regression to identify factors associated with the fraudulent financial statement.

Feroz et al. [67] used a neural network to catch fraudsters and accounting manipulation. KrambiaKapardis et al. [68] tested the use of artificial neural networks as a tool for fraud detection.

Hoogs et al. [69] used genetic algorithms to detect financial statement fraud. Lin et al. [70] applied a fuzzy neural network for the detection of fraudulent financial reporting. Liou [71] applied three classification techniques: decision trees, neural network, and logistic regression and used several financial variables to study their effect at detecting fraudulent financial reporting and predicting business failures, the results showed that logistic regression outperforms the other algorithms.

Perols [72] compared six techniques: neural network, support vector machines, logistic regression, decision trees (C4.5), stacking and bagging. The results showed that logistic regression and support vector machines perform well relative to the other methods. The author also noticed from the classification algorithms that six indicators out of 42 were used. Zhou and Kapoor [73] used data mining techniques like regression, neural networks, decision tree, and Bayesian belief networks, and explored a self-adaptive framework to detect financial statement fraud.

Kirkos et al. [74] investigated and compared the performance of three data mining techniques: neural networks, decision tree, and Bayesian belief network. They concluded that the Bayesian belief network outperformed other methods and identified the important factors associated with fraudulent financial statements. Ravisankar et al.

[75] used a multilayer feed-forward neural network, support vector machines, genetic programming, logistic regression, and probabilistic neural network, finally, they showed that the latter gave the best results.

2.2.4 Securities Fraud Detection Approaches

The current financial markets demand sophisticated tools to detect securities and investment fraud. Researchers used several factors and indicators like historical trading data for each trader or security for pattern recognition. Sometimes data about the employment history of traders and brokers are also used to find relationships and prevent any type of collaboration between them to manipulate markets. Neville et al. [76] used statistical relational learning algorithms, to generate probabilities of brokers committing violations in the near future.

Diaz et al. [77] used decision trees to extract rules that serve to identify new fraud manipulation pattern characteristics. Ferdousi and Maeda [78] used peer group analysis as an outlier detection method to identify abnormal behavior of brokers. Ögüt [79] compared artificial neural network, support vector machines, logistic regression, and discriminant analysis, using the difference between average daily return, average daily change in trading volume and average daily volatility. They concluded that artificial neural network and support vector machines perform better than the other methods.

Golmohammadi and Zaiane [11] used supervised learning algorithms: decision tree (CART and C5.0), random forest, naïve Bayes, neural networks, support vector machine, and k-nearest neighbors to identify suspicious transactions concerning market manipulation in the stock market. Results show that naïve Bayes outperforms other learning methods.

2.2.5 Insurance Fraud Analysis Approaches

Numerous techniques were used to detect fraud in the insurance area. Data used here come from insurance policies regarding the insured item (car, property, *etc.*) and the personal information of the owner or policyholder. A logit model for the detection of automobile insurance fraud is proposed in [80, 81]. Derrig and Ostaszewski [82] used a fuzzy set to classify automobile claims. Brockett et al. [83] used a self-organizing map to reveal claims fraud.

Stefano and Gisella [84] presented a fuzzy logic control model, that uncover suspicious automobile claims to be investigated by experts.

Ormerod et al. [85] used dynamic Bayesian belief networks called Mass Detection tool to detect fraudulent claims. Musal [86] presented two models: clustering algorithms and regression analysis to identify fraud in medical insurance. Ortega et al. [87] proposed using a multilayer perceptron neural network to detect fraud or abuse in health insurance. Using hidden Markov models, Tang et al. [88] introduced a detection system that identifies non-compliant health insurance claimants. The system includes several tasks such as feature selection, clustering, pattern recognition, and outlier detection.

Rule-based algorithms were investigated for the detection of health insurance fraud by Yang and Hwang [89]. Bermudez et al. [90] used a skewed logit model for fraud detection in automobile insurance claims. Pérez et al. [91] compared consolidated trees and C4.5 trees for fraud detection in a car insurance company. Bhowmik [92] used decision trees and naïve Bayes classifier and evaluated these techniques to solve the fraud problem in automobile insurance. Brockett and Golden [93] investigated and compared two statistical methods (logistic regression and discriminant analysis) and two artificial neural network (back-propagation and learning vector quantization) to identify financially troubled life insurers. The results show that both artificial neural networks perform better than traditional statistical methods.

Xu et al. [94] combined neural network classifiers to detect automobile insurance fraud. Pathak et al. [95] developed a fuzzy logic expert system to help auditors in detecting elements of fraud in insurance claims. Tao et al. [96] constructed a dual membership fuzzy support vector machine for insurance fraud identification.

2.2.6 Anti-Money Laundering Systems

One of the highly structured money laundering detection systems is the one used by the U.S. Financial Crimes Enforcement Network, it is described by Senator et al. [97]. It consists of a rule-based system, that computes suspicion scores for transactions and operations and uses simple Bayesian updating to collect evidence resulting in a suspicion score. Chartier and Spillane [98] presented a money laundering detection model using neural networks.

Wang and Yang [99] used the decision tree as a method to evaluate risks in money laundering. Lv et al. [100] proposed a radial basis function neural network for the detection of money laundering and compared it with support vector machines and outlier detection methods and concluded that it generates better results.

Wang and Dong [101] proposed a money laundering detection algorithm based on improved minimum spanning tree clustering. Larik and Haider [102] presented a clustering system to detect suspicious behavior along with statistical techniques to determine the deviation of a particular transaction from the corresponding group behavior. Keyan and Tingting [103] used an optimized support vector machine classifier to detect money laundering operations. Perez and Lavalle [104] proposed a two stages method to detect suspicious transactions. The first stage is to model user behavior. The second stage is to monitor new transaction to identify it as normal, no standard, fraudulent or suspicious. Liu et al. [105] used core decision tree and clustering algorithms for money laundering detection.

Lopez-Rojas and Axelsson [106] investigated the advantages and disadvantages of using synthetic data for the detection of money laundering using decision trees and clustering techniques. Khan et al. [107] proposed a Bayesian belief network approach to detect suspicious operation using transaction history, this method generates a score which is an indicator of a user's behavior. When a certain transaction deviates from normal behavior, the system gives an alert and the operation is then investigated.

Krishnapriya et al. [108] used a time-variant approach using behavioral patterns to identify money laundering.

Heidarinia et al. [109] used a fuzzy neural network that determines the riskiness of a client's behavior to be related to money laundering activities.

Alexandre and Balsa [110] introduced client profiling for anti-money laundering using clustering and rule-based algorithms. Suresh et al. [111] proposed a hybrid approach for the detection of suspicious transactions in the first stage of money laundering, using a hash-based association approach and for the identification of agents and integrator in the second stage using graph-theoretic approach.

2.2.7 Computer Intrusion Detection Systems

Many types of research were done to detect systems intrusion using a variety of techniques. Lee and Stoflo [112] built classification models using the association rules algorithm and the frequent episodes algorithm, on a data set where activities were identified as *normal* or *abnormal*. Shieh and Gligor [113] suggested a pattern-based approach and concluded that it is more efficient in detecting known types of an intrusion than statistical methods, but their approach was unable to detect new types of intrusion.

Moreover, Ju and Vardi [114] considered a Markov chain model for profiling the command sequence of a computer user to identify a *signature behavior* for the user, and therefore distinguish between intrusion behavior and normal behavior.

On the other hand, Forrest et al. [115] present a method based on artificial immune systems and the distinction between *self* and *non-self* patterns. In a different study, Ryan et al. [116] suggested the use of a back-propagation neural network to construct user profiles, and thus to identify abnormal behavior when it occurs. Besides, Schonlau et al. [117] conducted a study of six statistical approaches to detect the impersonation of other users. Yoshida et al. [118] used a sample of labeled emails and found that support vector machines are the best supervised method for spam detection. Lane and Brodley [119] introduced a new clustering algorithm to compress the data and map it to space where instance-based learning can be conducted.

Rule learning algorithms (RIPPER) were used for intrusion detection by Fan et al. [120]. Sequeira and Zaki [121] applied the k-means clustering algorithm assuming that clusters are grouped using only *normal* data. Hawkins et al. [122] and Williams et al. [123] used replicator neural network for the outlier detection. Idris and Shanmugam [124] proposed using a modified version of the APRIORI algorithm for implementing fuzzy rules for anomaly detection. Peng and Zuo [125] introduced a real-time detection system using frequent-pattern tree structure and frequent-pattern growth mining methods. Game theory was used by Patcha and Park [126] to catch the interaction between attackers and nodes and therefore to detect intrusion from mobile networks. Dalvi et al. [127] created a detection system using game theory to relearn the changed strategies of the adversary and a naïve Bayes classifier.

2.2.8 Data Driven Fraud Detection Approaches

Traditionally, data was very structured and limited, but nowadays users and everyone with access to the internet can generate data, even machines are generating data like monitors and satellites. Consequently, the amount of data generated and captured became proportionately larger; which created the need to find new database software and techniques to store this large amount of data and to analyze it. Big data is best described as large volumes of data both structured and unstructured, gathered from different sources and/or received at very high speeds.

Using Big Data analytics to solve different modeling and decision making problems, is becoming remarkable, due to the ability to process this huge amount of data and to generate accurate results in close to real-time, and thus reducing costs. Recently,

Big Data-based analysis was used in the fraud detection domain due to the need to detect frauds as soon as possible to reduce losses and to analyze large amounts of data sometimes unstructured or combining data from different sources to catch more complex patterns or rings of fraudsters.

Veeramachaneni et al. [128] introduced a system to detect intrusions and attacks combining four components: a Big Data processing system that catches features, an outlier detection method to learn a descriptive model of these features, a feedback mechanism (continuous learning), that will help for the training of a supervised learning model to predict if a new incoming event is normal or malicious.

Xu et al. [129] suggested using data mining-based methods along with a Big Data approach to address the problem of a type of loan request fraud (Peer-to-Peer lending), they argue that the use of Big Data analytics is recommended due to the large volumes and variety of data on loan request sites.

Hormozi et al. [130] proposed a credit card fraud detection system by executing the negative selection algorithm (it is one of the artificial immune systems algorithms) using Hadoop and MapReduce paradigm. This algorithm requires generating a huge number of random samples to test them against the *self* set, and to create a *detector set* that will be used to detect fraud. The authors used data of 300000 records of authorized credit card transactions obtained from a Brazilian bank, divided into 70% as training and 30% to validate their method. The authors aimed to reduce the training time of this algorithm, since generating this large number of samples is time-consuming. Then they parallelized the training algorithm in the Hadoop framework. They conducted several experiments according to different parameters and the number of mappers, and the results were very satisfying regarding the reduction of time in all experiments, the most significant one from 80760 seconds to 3084 seconds.

Kamaruddin and Vadlamani [131] proposed a one-class classification approach to solving the imbalance problem. the data consists of 5.96% fraud and 94.04% legitimate transactions. They proposed a hybrid system of Particle Swarm Optimization and Auto-Associative Neural Network (PSOAANN) and implemented it in a Spark computational framework. The auto-associative neural network was trained using only the majority class (legitimate transactions), the weights were optimized using the particle swarm algorithm, the error function to be minimized was the mean squared error. The classification task was completed in the testing phase. In this phase, only the fraud transactions are fed back to the network, and according to a specific threshold specified by the users,

and the values of the relative error, the records are classified as fraudulent or legitimate. The proposed system caught 89% of the fraud cases.

Sadasivam et al. [132] introduced an approach for financial statements fraud detection using the annual reports of companies, and analyzing them on Hadoop due to heterogeneity of data in these reports, then MapReduce technique is applied to select features that identify fraudulent financial reports. The number of these features is then minimized using principal component analysis, and the obtained features are used to train a support vector machine classifier. The results show that the use of the MapReduce technique improves time efficiency by 85%.

Bologa et al.[133] presented the benefits of using Big Data technology and parallel processing power and described the data analytics that should be used to detect fraud in healthcare insurance claims (business rules, anomaly detection, text mining, database searches, and social network analysis). Dora and Sekharan [134] also addressed the problem of healthcare insurance fraud detection, they built a model that aims to identify the fraudulent claims rapidly; by relating data from insurance companies and hospital records from the same area. Hadoop was used to preprocess data from both sources, and factor analysis to extract the most important features with discriminative power and combined them in one coherent data used for modeling. Then they used the decision tree, Naïve Bayes classifier, and clustering to detect fraud. They concluded that decision trees are the best models to detect fraud in healthcare insurance.

2.2.9 Real-Time Fraud Detection Systems

Fraud detection models can be performed as offline or online approaches. An online approach implies monitoring activities and detecting the fraudulent ones in real-time, which is very important especially in some fraud cases that are highly damaging or when activities are executed on the spot. Other cases do not require building specific models for online fraud detection. For example, financial statements fraud should be detected once the financial reports are submitted for audit. There is no need to detect in real-time like the case of credit card fraud or financial markets fraud, when a single transaction or operation is costly and result in many implications.

Few researchers focused on this subject using particular techniques. Quah and Sriganesh [135] developed a real-time credit card fraud detection model using self-organizing maps to distinguish fraud activities from normal behavior patterns. A hybridization of Basic Local Alignment Search Tool (BLAST) and Sequence Search and Alignment by Hashing Algorithm (SSAHA) algorithms were used by Kundu et al. [136], as a profile analyzer

and a deviation analyzer for credit card fraud. These algorithms increase the processing speed which allows the online response of the model. Sherly and Nedunchezhian [137] developed an adaptive credit card fraud detection system, using a Bootstrapped Optimistic Algorithm for Tree construction (BOAT). The model consists of two stages: first detecting anomalies by comparison with transaction history, then reducing false alarms by comparison with fraud history. It supports incremental updates, and the BOAT algorithm reduces the training time with less cost. Minegishi and Niimi [138] proposed using an online type of decision tree called Very Fast Decision Tree (VFDT) for the detection of fraudulent use of credit cards.

In the telecommunication sector, real-time detection requires a model that updates quickly according to the fast changes in users' normal behavior and is effective enough to catch fraudulent activities rapidly. Hollmen and Tresp [139] proposed a real-time fraud detection system using a hierarchical regime-switching model. Sun et al. [140] used two online anomaly detection schemes, the Lempel-Ziv (LZ)-based and Markov-based detection schemes, the first one is derived from the LZ-based data compression techniques, and the exponentially weighted moving average is used to model the changes in the user's normal behavior. The second one used a Markov model to describe the normal behaviour. Results show that the LZ-based scheme performs better than the Markov-based one. Krenker et al. [141] developed a system using a bidirectional artificial neural network that can predict mobile user behavior in real-time.

A visual framework was proposed by Huang et al. [142] for real-time financial markets fraud detection. It consists of two phases. First monitoring the real-time stock market and identify suspicious trading patterns using 3D treemaps that represent the securities along with their volume and prices, then a social network analysis to analyze the brokers' behavior.

In insurance, some scientists tried to expedite the process of fraud detection. Francis et al. [143] used support vector machines to detect medical claims fraud, the advantage of the system introduced was to provide a real-world speed up for experts in their work. Tsai et al. [144] proposed a model using rule technologies for medical insurance fraud, reducing the time and labor cost spent on the traditional detection systems.

2.2.10 Ensemble Learning Approaches for Fraud Detection

Ensemble learning and models combination has been used in several financial fraud detection studies in different domains.

In computer intrusion detection, Cho [145] introduced a hybrid system that uses self-organizing maps to reduce the data and preprocess it to be used in a hidden Markov model with fuzzy logic. Genetic algorithms based on fuzzy logic were also used for intrusion detection by Prasad et al. [146] and Dhanalakshmi and Babu [147].

In credit card fraud detection, Duman and Ozcelik [148] used a genetic algorithm combined with a scatter search to minimize the number of wrongfully classified transactions. Other studies in this field combined neural network with other algorithms like Ogwueleka [149]. The latter proposed using an artificial neural network with a rule-based component. Patidar and Sharma [150] applied artificial neural network tuned by genetic algorithms for the same purpose.

In telecommunication fraud detection, Farvaresh and Mehdi [151] introduced a hybrid system to detect subscription fraud. Their system consists of three phases: preprocessing using principal component analysis to reduce data, clustering using k-means and self-organizing maps, finally the classification. Support vector machines, neural network and decision tree (C4.5) were used in the classification phase as single classifiers; bagging, boosting, stacking and voting as ensembles. They concluded that a support vector machine (as a single classifier) and boosted trees gave the best results in detecting fraud.

In financial auditing fraud detection, Chai et al. [152] and Lenard et al. [153] used fuzzy logic along with expert systems and rule-based reasoning to assess the risk of managerial fraud. Kotsiantis et al. [154] conducted a comparative study using multiple techniques: decision trees (C4.5), Bayesian belief network, artificial neural network, k-nearest neighbor, rule learning algorithm, logistic regression and support vector machines. They also created a hybrid model by combining these methods with a decision tree algorithm for learning at meta-level and compared it with other hybrid models (voting, grading, bestCV). The proposed stacking variant methodology achieved better results than any simple and ensemble method. Chen [155] built a fraudulent financial statement detection model that consists of two stages. The first stage was the selection of variables where tree algorithms are used (CART and CHAID). Then they construct the model combining CART, CHAID, Bayesian belief network, support vector machine, and artificial neural network.

In financial markets and securities fraud, Blume et al. [156] combined social network analysis and interactive visualization to identify malicious accounts as an attempt to catch trader accounts that collaborate in market manipulation.

In insurance fraud, Williams et al. [157] proposed a methodology to detect healthcare insurance fraud, of three steps: first a k-means clustering to find groups, then a tree algorithm (C4.5) to extract rules, finally the visualization of these rules along with statistical summaries. In [158], they improved the previous algorithm by generating and extracting the rules using genetic programming. A hybrid meta-classifier system was proposed by Phua et al. [21] for the detection of automobile insurance fraud. The authors used the back-propagation neural network, naïve Bayes, and decision tree (C4.5), then combined these algorithms using a stacking-bagging approach. Viaene et al. [159] used boosted naïve Bayes to detect insurance fraud.

For money laundering detection, Kharote and Kshirsagar [9] proposed a model that allows taking data from different sources and pre-process them to be clustered using the k-means algorithm. Then, the clusters are analyzed using frequent pattern mining algorithms, and user profiles are extracted to be used for the detection of anomalous and suspicious operation. Le Khac et al. [160] presented a model combining data mining techniques of clustering and neural network to detect suspicious money laundering activities in an investment bank.

2.3 A Comparative Study

As mentioned in Chapter 1 that the literature review may not be sufficient to discover the observable efficiency of the state of the art technologies. Thus, we conducted an experimental study. This Section reports the experimental study that was performed with machine learning algorithms and imbalance classification approaches, detailed in Appendix A. First, we provide a detailed description of the design of experiment followed by the results and discussion. Finally, we discuss some critical shortcoming we discovered based on our experiment.

In this experiment, we aim to identify weaknesses or disadvantages of existing class imbalance approaches; by comparing them with the original classifiers to evaluate their added value. Our goal is to detect the issues that must be solved to product a highly efficient solution for the class imbalance problem.

2.3.1 Design of Experiment

This section briefly presents the workflow of our experiments, the data set used, the selection of target variables and performance measure.

The Workflow of Experiments

Our experimental study is organized as follows. The experiment is presented and discussed in two phases. In the first phase, eight classification methods, described earlier in Section A.1 are compared. The comparison was carried out with respect to three measures: *accuracy*, *sensitivity*, and *the Area Under Precision-Recall Curve* (AUPRC) (see Section 1.3.2). This comparison results in selecting the most suitable algorithms including the following: C5.0, SVM, and ANN.

In the second phase, the selected algorithms are used in comparing selected imbalance classification approaches such as *Random Oversampling*, *One-Class Classification* and *Cost Sensitive*. The C5.0 decision tree algorithm is used and compared to C5.0 with Random Oversampling (RO) and C5.0 with Cost Sensitive tree (CS). Then, the SVM is used as a binary classification tool, and compared to the One-Class Classification OCC SVM and Cost Sensitive CS SVM. Also, the ANN is applied and compared to the Auto-Associative Neural Network (AANN).

Data set and Variable Selection

The data set used in our experiment is a credit card fraud labeled data[1]. It contains ten million credit card transactions described by 8 variables that include the customer ID, the gender, the state where the customer lives, the number of cards he or she holds, the balance in the account, the number of local and international transactions made to date and the credit limit. In the data set, 596,014 (5.96%) are fraud cases and 9,403,986 (94.04%) are legitimate. These data demonstrate the imbalance problem with 5.96% fraud cases. Data are divided into train set (70%) to create the models, and a test set (30%) to study their performance. Furthermore, we used a smaller data set that is 2% of the original data set. We maintained the same imbalance ratio.

2.3.2 Results and Discussion

In this section, we discuss the results of our experiments. The result of this experiment is summarized in Table 2.1. We found that for all algorithms the accuracies are higher than 90% with different sensitivity and AUPRC values. However, exploring the results, we concluded of all these algorithms C5.0 algorithm, SVM and ANN are the most eligible ones to be used in evaluating the performance of the imbalance classification approach described in Section A.2. The reason is: (i) comparing the results produced by the other

[1]Available at http://packages.revolutionanalytics.com/datasets/

algorithms, C5.0, SVM, and ANN produced balanced and the most reasonable outcomes in all three measures (Accuracy, Sensitivity, and AURPC); and (ii) these three algorithms are not constrained to mathematical or statistical assumptions.

Table 2.1 Performance of different methods

Method	Accuracy	Sensitivity	AUPRC	Performance
C5.0	96%	43%	0.6	High accuracy
SVM	96%	39%	0.63	High accuracy
ANN	96%	47%	0.62	High accuracy
LR	96%	49%	0.66	High AUPRC
NB	93%	56%	0.5	High sensitivity
BBN	94%	15%	0.64	High AUPRC
KNN	95%	45%	0.29	Low AUPRC
NSA	92%	51%	0.29	Low AUPRC

NB and NSA have the highest sensitivities, yet the sensitivity is obtained with increasing the number of false alarms (accuracy less than 94%).

The experiments with BBN produced the lowest sensitivity (15%) and the highest AUPRC (0.64) - which is promising. In this case, we concluded that different thresholds of class probabilities may produce better results.

Moreover, BBN needs normality assumption for the continuous variables, which is not satisfied all the time. The layout of BBN is shown in Figure 2.1. This structure of BBN was implemented based on expert knowledge. The algorithms commonly used to find BBN layout gave non-logical results, such as the node *fraud* being orphan, or a parent node, or being only dependent of variables such as *gender*, *state* or *cardholder*.

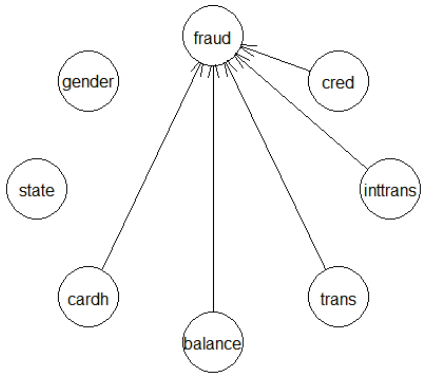


Fig. 2.1 BBN layout

In our experiment with KNN, data is first normalized using the min-max scale. The normalization is important especially when the variables are in diverse scales and ranges. Otherwise, the algorithm will be biased towards variables with larger scales [2]. For example in our case, the bias will be towards the variable “balance”, whereas it may not be the most important variable. The Euclidean distance is used and five nearest

neighbors are considered in the classification. The simplicity of this method does not allow for efficient results as shown in table 2.1. We obtained the lowest AUPRC using this method.

We found that the results of the LR model are the best of all. First, the multi-collinearity test was carried out using Variance Inflation Factors (VIF) to make sure that no variable can be written as a linear combination in terms of the others. The highest AUPRC was achieved using this model, indicating the best sensitivity with high accuracy.

Considering NSA, the selection of thresholds for the euclidean distance was challenging. After investigating several thresholds, it was set to 100. This method generated a considerably good sensitivity rate but with the lowest accuracy and AUPRC. It is worth noting that, unlike all other methods, a smaller data set was used to train this algorithm (1000 transactions) while keeping the same imbalance ratio. The reason behind using a small data set is that the computation cost for large data set is high; a high-performance system is required to conduct an experiment with NSA on the larger data set. The system we used for our experiment was rather simple and hence was not able to handle high computation cost. Besides, multiple parameters affect the accuracy of this model such as *the threshold* and *the size of the detector set*. We found a set of 2000 detectors, yet a bigger set is needed to achieve better results which would increase the training time and computation cost significantly.

Figure 2.2 presents different PR curves for the methods that singled out for our experiment with imbalance classification approaches. According to the results of AUPRC shown in Table 2.1, LR is the best method. Yet, LR was not chosen due to complexity issues. The other methods KNN and NSA resulted in either low sensitivity or low accuracy due to high sensitivity, which leads to low AUPRC.

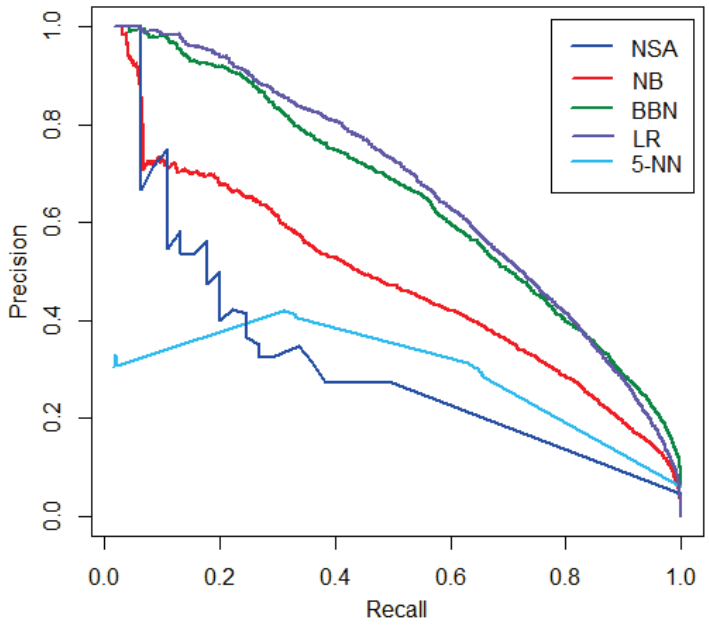


Fig. 2.2 Comparing PR curves

After selecting C5.0, SVM and ANN algorithms, we studied the performance of the imbalance approaches and their improvements to these methods. We begin with the C5.0 algorithm. The confusion matrices of the C5.0 methods are as follows:

C5.0

Predicted	Actual	
	0	1
0	55838	2053
1	546	1563

RO C5.0

Predicted	Actual	
	0	1
0	52988	1197
1	3396	2419

CS C5.0

Predicted	Actual	
	0	1
0	54200	1252
1	2184	2364

In the experiment with random oversampling, we replicated each observation on average 15 times due to the ratio of imbalance. For the cost-sensitive approach, the two new parameters that are added C^+ and C^- are equal to 3 and 1 respectively. In other words, the cost of wrongly classifying a fraud is considered three times the cost of wrongly classifying a legitimate transaction. The trees are too complex to be visualized (145 leaves for C5.0, 5020 leaves for RO C5.0 and 581 for CS C5.0). According to the tree's algorithm, the most important variables that are contributing to the discrimination are *balance*, *creditLine*, *numTrans* and *numIntTrans*.

The confusion matrices of SVM methods are presented below.

SVM

Predicted	Actual	
	0	1
0	56049	2190
1	335	1426

CS SVM

Predicted	Actual	
	0	1
0	54723	1268
1	1661	2348

OCC SVM

Predicted	Actual	
	0	1
0	51349	2522
1	5035	1094

In the case of SVM, similar to the CS C5.0 the costs are considered 1 for non-fraud class and 3 for the fraudulent ones. For OCC, in the training set only the fraud class is used. For the test set, the observations of both classes are used to evaluate this approach.

In the following, we present the ANN confusion matrices.

ANN

Predicted	Actual	
	0	1
0	55824	1893
1	560	1723

AANN

Predicted	Actual	
	0	1
0	44843	1924
1	11541	1692

In the experiment with ANN (shown in Figure 2.3) the network is composed of the input layer (7 nodes), one hidden layer (3 nodes) and an output layer. The activation function used is the sigmoid activation function. The algorithm used to adjust weights is the Resilient backpropagation algorithm (Rprop). Figure 2.4 shows the network resulted from AANN. It is composed of an input layer and output layer with 7 nodes, and 3 hidden layers with 2, 1 and 2 nodes respectively.

In the experiment with AANN, only the fraud cases are used to train the network. The Mean Absolute Error (MAE) is calculated as an average error and used as a threshold in the testing phase. For each observation having an MAE higher than the training's MAE is considered a fraud. Note that, the MAE represents the average absolute value of the error produced by the prediction, i.e. the error is the difference between the actual value and their forecasts.

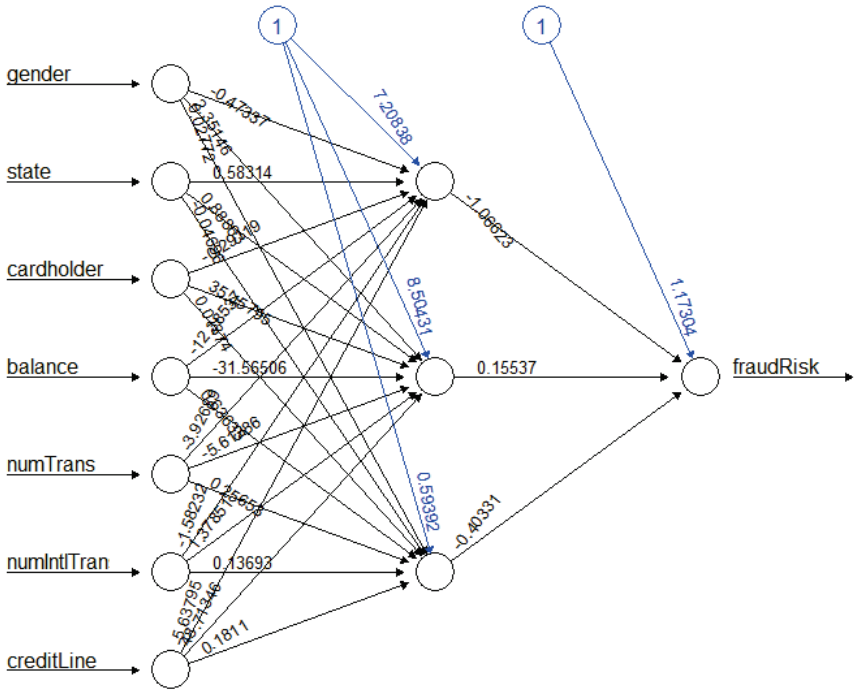


Fig. 2.3 ANN plot

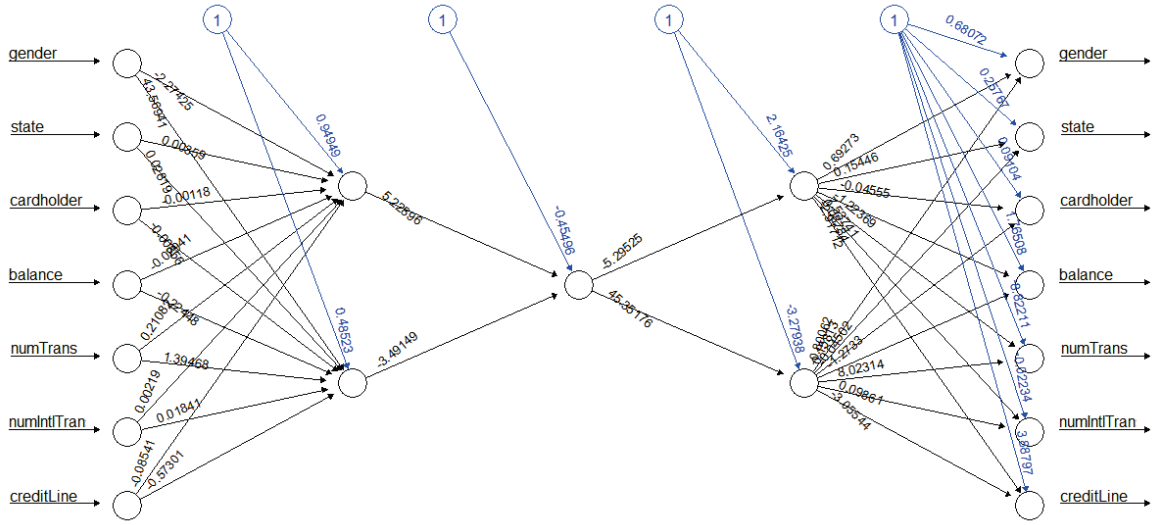


Fig. 2.4 AANN plot

Table 2.2 shows the accuracy, the sensitivity and the AUPRC of the methods employed for the imbalanced classification. These curves are shown in the following Figures 2.5, 2.6 and 2.7. Note that, the PR curves could not be plotted for CS C5.0, because the final

class prediction for a sample according to the equation presented in Section A.2.3, is a function of the class probability and the cost structure, not just the class probability [161]. Also, the class probabilities for one-class classification SVM are not supported yet.

Table 2.2 Table summarizing the performance measures of imbalance approaches

Method	Accuracy	Sensitivity	AUPRC
C5.0	96%	43%	0.6
RO C5.0	92%	66%	0.52
CS C5.0	94%	65%	Could not be plotted
SVM	96%	39%	0.63
OCC SVM	87%	30%	Could not be plotted
CS SVM	95%	65%	0.62
ANN	96%	47%	0.62
AANN	77%	46%	0.11

Table 2.2 shows that the accuracy for all methods is higher than 94% with different sensitivity levels, except for the OCC approaches (for SVM and AANN). According to AUPRC, the original classifiers (C5.0, SVM, and ANN) are performing better than the ones with imbalanced classification approaches. Even though, it is remarkable to all the models that the CS approach improve the sensitivity. Table 2.2 shows that these CS approaches to increase sensitivity, at the cost of slightly decreasing the accuracy. However, the OCC severely affects the algorithm in terms of accuracy and sensitivity. We believe that the reason behind this decrease in performance is the overfitting of data since the OCC approaches are built using the fraud observations only.

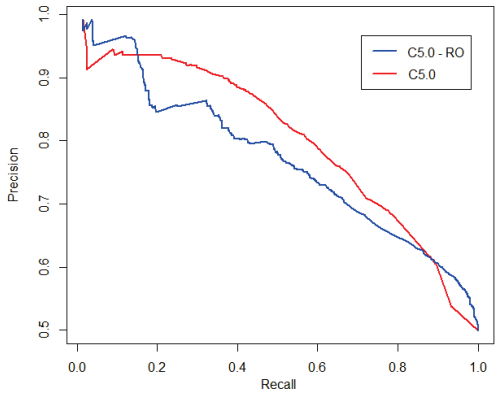


Fig. 2.5 PR curves for C5.0 methods

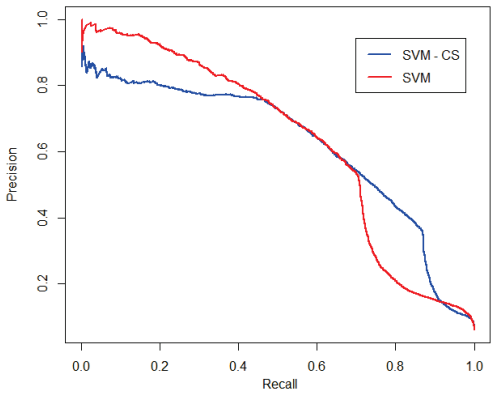


Fig. 2.6 PR curves for SVM methods

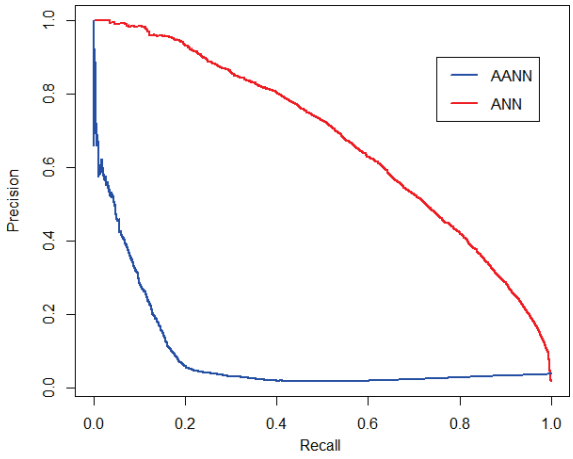


Fig. 2.7 PR curves for ANN methods

2.4 Shortcomings of Existing Methods

Fraud is a criminal practice for the illegitimate gain of wealth or tampering information. Over the last few years, various techniques from different areas such as data mining, machine learning, and statistics have been proposed to deal with fraudulent activities.

In the experimental study previously described, we discovered several shortcomings of existing methods. The study revealed that the approaches normally used to solve imbalance problems may have unpleasant consequences. We found that the approaches designed specifically to tackle the imbalance problem are not adequately effective. These approaches improve sensitivity, yet the improvement leads to an increase in the number of false alarms and hence the accuracy and AUPRC decrease. Practically speaking, this can be costly for the financial institution just the same way a fraud event costs. Even a minimal deterioration of accuracy say 1% hides a large misclassification rate of the majority class *i.e.* legitimate observations.

Our research problem is summarized as follows: using imbalanced classification approaches, the number of false alarms generated is higher than the number of frauds that are more detected. The results of this experimental study greatly motivated us to explore other methods that focus on detecting the hidden patterns of fraud, with a minimum misclassification rate.

Chapter 3

CoSKNN: Cost-Sensitive Cosine Similarity K-Nearest Neighbors Approach

The available imbalance classification approaches that are used, often increase the detection of the minority group at the cost of generating false predictions for the other class, which leads to an overall decrease of the model's accuracy. In this chapter, we develop the cost sensitive KNN based on cosine similarity.

First, we present the CoSKNN. Then, we compare it with other KNN algorithms and imbalanced classification methods, such as cost sensitive decision tree and one-class

classification SVM. Finally, we proved its efficiency using four performance measures (accuracy, sensitivity, PR curve and the F_1 score).

3.1 Introduction to K-Nearest Neighbors

In this section, we will describe in detail the K-Nearest Neighbors (KNN) classifier algorithm using both simple voting or distance weighted KNN. We will also describe a cost-sensitive KNN approach introduced by Qin et al. [28] for comparative purposes.

3.1.1 Classification Using KNN

As described in Section A.1, KNN is a data mining method widely used for classification and regression. It is a simple algorithm that consists of using the k nearest points to the one we aim to classify or predict [2]. KNN's performance depends on many parameters including the following:

1. The distance measure used to find the k nearest points. A specific norm is used to measure the distance between points. The norm commonly used to find the distance between two observations p and $q \in \mathbb{R}^n$ is the euclidean distance:

$$d(p,q) = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2}$$

2. The choice of the number of neighbors to consider k. It highly depends on the available data. A high number cannot be considered if we have a small data set. On the other hand, a very small k can lead to underfitting.
3. The decision rule for the classification, using the k nearest points of the new observation. It represents the criteria that the classification is done according to it. We will present in the following the two common decision rules used for classification, simple voting and distance weighted.

Simple Voting

The most straightforward rule used for the classification is the simple voting. The new observation is assigned to the majority class of the k nearest points. The classification is

done according to the following formula:

$$\hat{y} = \underset{\alpha \in \{0,1\}}{\operatorname{argmax}} \sum_{k \in K} \delta(\alpha, \mathrm{cl}_k)$$

Where:

- $\hat{y}$ represents the foretasted value of an observation y
- $\alpha \in \{0,1\}$ represents the possible classification categories of the target variable
- K is the subset of the chosen nearest neighbors of y
- cl_k represents the class of the neighbor $k \in K$

The function δ is defined as follows:

$$\delta(x,y) = \begin{cases} 0 & \text{if} \quad x = y \\ 1 & \text{if} \quad x \neq y \end{cases}$$

The function argmax returns the value of the category, whether 0 or 1 in our case, at which the maximum is reached, i.e. the category of the majority of the neighbors.

Distance Weighted Voting

Another decision rule considered is a distance weighted voting. In the simple voting, all neighbors had equal weights. The idea behind this approach is to take into account the distance of the neighbors and to assign a higher weight to the closest ones. The prediction rule is done as follows:

$$\hat{y} = \underset{\alpha \in \{0,1\}}{\operatorname{argmax}} \sum_{k \in K} w_k \cdot \delta(\alpha, \mathrm{cl}_k) \quad \text{with} \quad w_k = \frac{1}{d_k^2}$$

Where w_k is the assigned weight and d_k is the euclidean distance between y and the considered neighbor. According to the formula presented above, a close neighbor, will have a small d_k, and thus higher weight w_k.

3.1.2 Cost Sensitive KNN

In [28], Qin et al. introduced two cost-sensitive approaches for KNN. The first one called Direct Cost-Sensitive KNN (DirectCS-KNN) is based on calculating class probabilities

from the original KNN algorithm using the following formula:

$$P_i = \frac{k_i}{k}$$

Where P_i represents the probability that the class of a new observation y is i, and k_i is the number of neighbors of class i and k is the total number of neighbors chosen. This equation is an improvement to the majority rule used in simple voting since it allows to quantify the majority by a probability.

The other approach is distance weighted called Distance-CS-KNN. Considering two classes 1 and 0. The idea consists of calculating costs for both classes κ_1 and κ_0. The new sample is assigned to the class with lower κ_i. These costs are calculated as follows:

$$\kappa_1 = f_p \times P_0 \quad \text{and} \quad \kappa_0 = f_n \times P_1$$

f_p and f_n represent respectively the costs of false positives and false negatives.

P_i are fraud probabilities, calculated in a cost-sensitive manner different than the one presented in DirectCS-KNN, using cost sensitive weights W_i.

$$P_1 = \frac{W_1}{W_1 + W_0} \quad \text{and} \quad P_0 = \frac{W_0}{W_1 + W_0}$$

$$W_i = \sum_{k \in K_i} w_k$$

Where:

- K_i is the subset of neighbors of class i
- w_k is the distance weight of the neighbor as calculated in the distance weighted method. $w_k = \frac{1}{d_k^2}$, where d_k is the euclidean distance between the neighbor and the data point y.

This approach uses distance weights and imbalance costs to tackle the imbalance issue using KNN.

3.2 CoSKNN: Approach Theory and Implementation

We developed a Cost-Sensitive KNN method. We aimed to tackle the imbalance problem by using cosine similarity as a distance metric and by introducing a cost sensitive score

for the classification. To improve the method's performance in terms of imbalance, we also studied the choice of the score's thresholds and the number of neighbors to consider. These steps are described in the following.

3.2.1 The Use of Cosine Similarity

The first step of our method is the change of the distance metric used. We replaced the euclidean distance used in KNN with Cosine Similarity (CoS) when calculating the distance between observations in order to find the nearest neighbors. This metric consists of calculating the angle's cosine between two vectors p and q representing two observations. This is calculated as follows:

$$CoS(p,q) = \frac{\sqrt{\sum_{i=1}^{n} p_i q_i}}{\sqrt{\sum_{i=1}^{n} p_i^2}\sqrt{\sum_{i=1}^{n} q_i^2}}$$

Note that, this metric ranges between -1 and 1. If CoS is close to 1, it indicates that the angle between the two observations is close to zero and therefore they are similar, i.e. neighbors. Otherwise, these two observations are considered as dissimilar.

The advantage of using CoS instead of Euclidean distance is highlighted in the coming section when we compare KNN (whether simple voting or weighted distance) using both metrics and we prove that CoS is better in terms of sensitivity and accuracy.

3.2.2 The Introduction of the Score S_y

The second step of this method, after finding the neighbors, is to introduce an imbalanced classification score using CoS. The idea is to evaluate the similarity of an observation to its neighbors of the minority class, taking into account the other class as well. This is done by calculating the following score S_y for each observation y:

$$S_y = \frac{\sum_{i=1}^{k} \mathrm{Cl}_i \,.\, CoS_i}{\sum_{i=1}^{k} CoS_i}$$

Where:

- k is the number of neighbors considered for y.
- Cl is a vector of length k, $\text{Cl}_i \in \{0,1\}$ that represents the classes of the neighbors. 0 is used to denote the class of the majority group and 1 is used for the minority group.
- CoS is another vector representing the cosine similarity between y and its neighbors.

This score ranges from 0 to 1. It works similarly to a probability, that describes the likelihood of observation to be in the minority group. When it's close (or equal) to zero, it indicates that the neighbors are mostly (or all) of the majority group, which would lead to a majority group classification. However, when at least one of the neighbors of y is of the minority class, this score will be higher than zero; and closer to one, the more the neighbors are of the minority group.

3.2.3 The Classification Using CoSKNN

The classification is done according to a certain threshold $\theta \in [0,1]$. $\hat{y}$ represents the prediction of y.

$$\hat{y} = \begin{cases} 0 & \text{if} \quad S_y \leq \theta \\ 1 & \text{if} \quad S_y > \theta \end{cases}$$

the Score Thresholds

The choice of θ is not straightforward. A very low value will lead to a large number of false positives (observations of the majority group classed as a minority). However, a high threshold value will lead to a very low sensitivity rate. Therefore, θ should have a slightly low value. θ is chosen as the cut-off value that maximizes the F_1 score. (More details about the threshold's choice is provided later in Section 4.3.3)

The Choice of k

The choice of k affects many aspects of the KNN method. The number of neighbors should be large enough to be informative about the observation's neighborhood. On the other hand, due to the imbalance, a high number of neighbors will make the classification biased towards the majority group and time-consuming. For each experiment we conducted with KNN, we investigated several possible values of k to determine the best.

3.3 Validation Experiment

To prove the efficiency of the CoSKNN method we developed, we compare it with the original KNN classifier using both euclidean distance and cosine similarity, with simple voting and distance weighted KNN. We also compare it with cost sensitive C5.0, one class classification SVM, and the Distance-CS-KNN [28] previously described in Section 3.1.2.

We compare all these methods using the same data set used in the previous experimental study (Section 2.3). Recall that the credit card fraud labeled data, contains the following explanatory variables used previously, *gender*, *state*, *cardholder*, *balance*, *numTrans*, *numIntTrans*, and *creditLine*. The target variable is *fraudRisk* indicating if each observation is fraud (denoted 1) and legitimate (denoted 0) with an imbalance ratio of 5.96%.

Moreover, we extracted here a part of the original data due to the time consumption of the KNN method when calculating the distance to all observations in the training set. Extracted data consists of 6000 credit card transactions (observations) in which we have 5657 *Legitimate*(0), and 343 *fraud*(1) cases. This data set takes into account the same imbalance ratio as the original one.

The data is divided between train (3999 transactions) and test (2001 transactions) with a similar ratio of imbalance. For all KNN methods, 10 nearest neighbors are considered to classify the new observations. Due to the widely different scales of the explanatory variables, data is first normalized using the mean and standard deviation of the variables, to avoid bias towards variables with large ranges [2]. Table 3.1 lists the different methods we compared along with their descriptions.

Table 3.1 Methods used in the comparison

Method	**Description**
EucKNN	Simple voting KNN using the euclidean distance
CKNN	Simple voting KNN using cosine similarity
DEucKNN	Distance weighted KNN using euclidean distance
DCKNN	Distance weighted KNN using cosine similarity
Distance-CS-KNN	The distance based cost sensitive approach introduced by Qin et al. [28]
CS C5.0	Cost sensitive C5.0 decision tree
OCC SVM	One Class classification Support Vector Machine
CoSKNN	Cost sensitive cosine similarity based KNN

The challenge is that most imbalanced classification methods focus on increasing sensitivity, which will lead to a slight decrease in accuracy. This decrease that can sometimes be just 1% may seem insignificant, but in fact, it hides a high number of false alarms. Thus, we need to rely also on measures that find a trade-off between the high accuracy and the sensitivity; the Area Under Precision-Recall Curve (AUPRC), is used for that purpose. However, we will also use the F_1 score, since we may not always be able to plot this curve like the case of CS C5.0 and OCC SVM.

3.3.1 Methods Results

In the following, we present the results of the comparison of CoSKNN with the other methods. Table 3.2 shows the performance measures (the accuracy, the sensitivity, the AUPRC and the F_1 score) for all methods mentioned in Table 3.1. The PR curves are shown in Figure 3.1.

Table 3.2 Table summarizing the performance measures

Method	**Accuracy**	**Sensitivity**	**AUPRC**	**F_1 score**
EucKNN	95.8%	0.30	0.39	0.45
CKNN	95.8%	0.42	0.53	0.53
DEucKNN	95.6%	0.32	0.22	0.46
DCKNN	95.4%	0.34	0.54	0.46
Distance-CS-KNN	94.5%	0.53	-	0.52
CS C5.0	93.3%	0.65	-	0.53
OCC SVM	88.9%	0.22	-	0.18
CoSKNN	**95.5%**	**0.51**	**0.54**	**0.56**

Table3.2 shows that the accuracy is higher than 94.3% for all models except the CS C5.0 and OCC SVM. The slight differences in the accuracy between all the other methods show how much information this measure hides in a class imbalance case. We can conclude from these results when comparing the performance measures of EuCKNN with CKNN and DEuCKNN with DCKNN that the use of cosine similarity is improving the classification according to the sensitivity, AUPRC and F_1 score, with a reasonable decrease in accuracy.

The advantage of using cosine similarity is also observed in Fig. 3.1, where the three methods using cosine similarity have PR curves much closer to the upper right corner

than the curves of the methods using Euclidean distance. In the table, this is witnessed with the highest AUPRC for obtained with this cosine similarity based methods.

Our approach CoSKNN is outperforming all the methods according to the AUPRC and F_1 score. This method achieved the highest F_1 score of 0.56. It is considerably improving the sensitivity when compared to the simple KNN. The other cost sensitive models are performing better in terms of sensitivity but at the cost of raising false alarms and decreasing the accuracy sometimes to a less than acceptable value, like the case of CS C5.0 and OCC SVM.

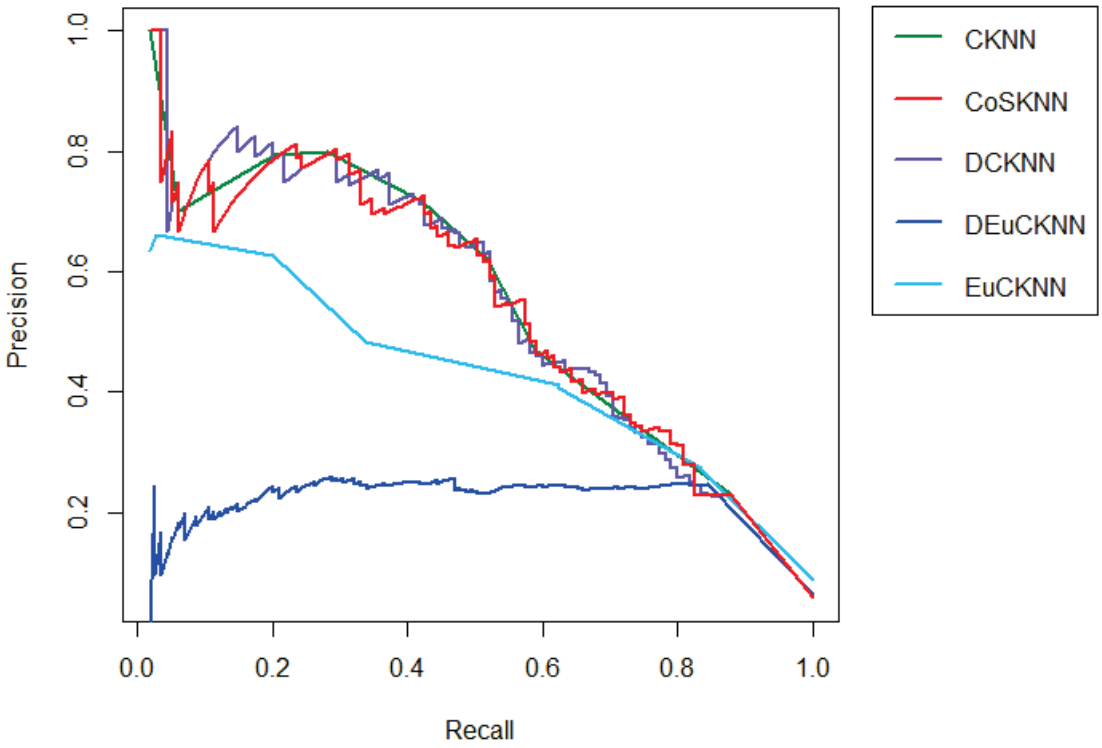


Fig. 3.1 PR curves for all methods

3.3.2 Discussion

In this chapter, we addressed the class imbalance problem. We investigated the use of KNN in fraud detection and we proposed a cost-sensitive KNN method. We provided comprehensive details of our method CoSKNN where we used cosine similarity instead of the euclidean distance to find the neighbors, and then we calculated a cost sensitive score to evaluate the probability of fraud risk.

We also compared the performance of simple voting KNN and distance weighted KNN using both euclidean distance and cosine similarity, with another cost sensitive KNN,

cost sensitive decision tree, one class classification SVM and CoSKNN. The comparison was done by applying these methods to a credit card fraud data set with imbalance, using multiple performance measures, mostly relying on AUPRC and F_1 score. This experiment shows that CoSKNN is outperforming all the other methods. Table 3.3 shows the different advantages and disadvantages of the CoSKNN method.

Table 3.3 Advantages and disadvantages of CoSKNN

Advantages	**Disadvantages**
– Higher sensitivity, i.e. higher fraud detection rate. – Acceptable accuracy. – Simple and easily interpreted algorithm.	– Time consumption specially when dealing with a big data. – Dependency on the ability of KNN in detecting the fraudulent patterns, and the use of cosine similarity.

It is proven that using our method, we obtained a high sensitivity rate with a slight and acceptable decrease in accuracy. CoSKNN showed a time consumption issue when the data set used is large. This method is an improvement of the KNN algorithm. However, it might not be efficient if in a certain data set the patterns of fraud are not detectable by the KNN algorithm. In other words, if the relationship between the fraudulent observations is not measurable with a simple distance measure like cosine similarity, the method might not reach a good detection rate, even though its results will still be better than simple KNN.

This was our motivation to investigate combination algorithms and ensemble learning. These approaches are not constrained by one method nor dependent on it, and thus provide better results in all types of data sets and for different types of fraud. In the following chapter, we provide more details about ensemble learning and how we used a combination approach to tackle the class imbalance problem.

Chapter 4

K-MICHA: K-Modes Imbalance Classification Hybrid Approach

4.1 Introduction to Ensemble Learning

In the past few years, several automated systems based on machine learning algorithms have been developed to detect fraud. Other algorithms were developed specifically to target the class imbalance problem. Ensemble learning or model combination is a common approach developed by building one model using either several samples of the data or several algorithms. This aims to obtain better performance and higher accuracy

rates. Every model combination requires two phases, diversification and integration as described in the following.

1. The diversification phase refers to the type of variation used in the combination. Sometimes different samples of the data set are used to create different models using the same machine learning algorithms, such as bagging [162] or boosting [2]. Sometimes, different sets of variables are chosen to create different algorithms alongside the different samples of the data set, like random forest [163]. Another diversification approach known as hybrid diversification is used, where different machine learning algorithms are considered to create multiple models instead of different data samples.
2. The integration phase refers to the strategy used to combine the results generated by the models. This can be done according to several approaches. The most straightforward way is simple or weighted voting, specifically for classification problems. For the regression, the mean or weighted average is used. Another more advanced integration technique is meta-learning using a second level machine learning. In this case, the method's results are combined in new metadata, where a new machine learning algorithm is applied and trained for prediction.

In this chapter, we will first present different examples of ensemble learning methods. Then, we will describe our approach K-MICHA, a hybrid approach using the k-modes clustering algorithm. Our approach aims to evaluate fraud probabilities and use them for fraud detection of new observations. This is done by clustering similar data points in terms of the outputs of the classifiers, then calculating fraud probabilities in each cluster. The theory and implementation of K-MICHA are explained in the following sections.

4.2 Examples of Ensemble Learning

This section summarizes examples of the basic ensemble learning techniques, following two approaches where different machine learning methods are used, and where different data sets samples are used.

4.2.1 Simple and Weighted Voting

Simple voting is a basic combination technique used with different machine learning methods. The new observation will be assigned to a certain class if the majority of the methods predict this class. The classification for a certain observation x is done

according to the formula using the statistical model:

$$\hat{x} = \text{mode}\{O_1, O_2, \ldots, O_N\}$$

Where $\hat{x}$ is the prediction of the voting of an observation x, N is the number of methods combined, O_i is the output of the i^{th} method.

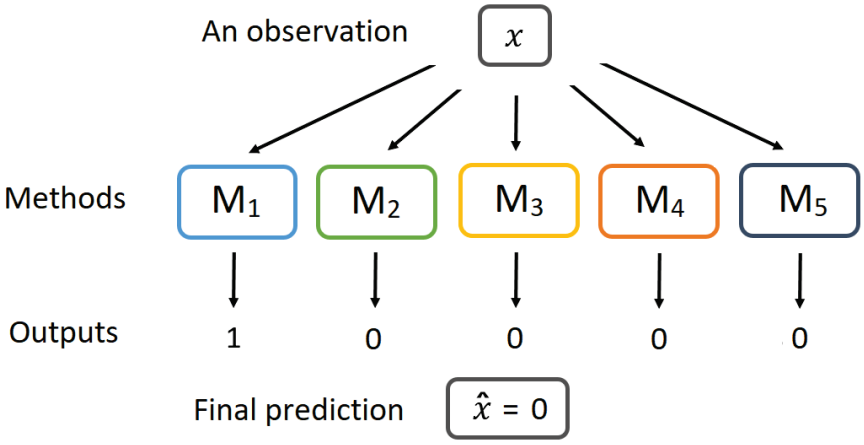


Fig. 4.1 Example of voting

Fig. 4.1 represents an example of the voting method, where $N = 5$ and M_i represents the different methods. The vector = (1, 0, 0, 0, 0) represent the 5 different outputs. The final prediction is the mode of the output vectors equal to 0 in this example.

Weighted voting is a more advanced voting technique where specific weights are assigned to each method to highlight its performance. A higher weight would indicate a more important prediction compared to other methods. The weights specified by the user and may vary from case to case. This technique is illustrated in Fig. 4.2 where weights are added to the same example presented in Fig. 4.1. In this example, even though just M_1 predicts 1, the final prediction will still be 1 since this method has a weight of 0.65. The classification is done according to the following formula:

$$\hat{x} = \underset{c}{\text{argmax}} \sum_{i=1}^{N} w_i \delta(\alpha, O_i)$$

Where $\hat{x}$ represents the foretasted value of a new observation x, $\alpha \in \{0,1\}$ represents the possible classification categories, O_i is the output of the i^{th} method and w_i is the

weight assigned to the i^{th} method. The function δ is defined as follows:

$$\delta(\alpha, y) = \begin{cases} 0 & \text{if} \quad \alpha = O_i \\ 1 & \text{if} \quad \alpha \neq O_i \end{cases}$$

The function argmax returns the value of the category, whether 0 or 1 in our case, at which the maximum is reached.

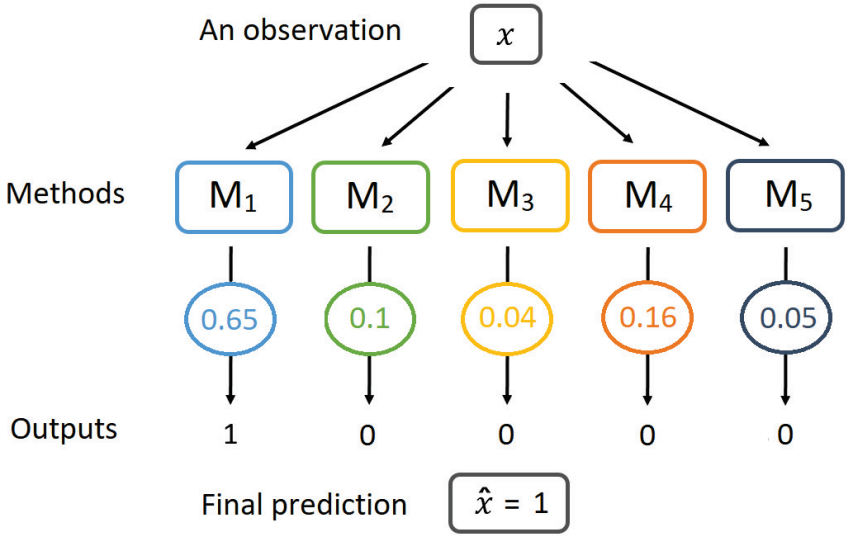


Fig. 4.2 Example of weighted voting

In the example, the prediction $\hat{x}$ is equal to:

$\underset{\alpha}{\text{argmax}}[0.65 \times \delta(\alpha,1) + 0.1 \times \delta(\alpha,0) + 0.04 \times \delta(\alpha,0) + 0.16 \times \delta(\alpha,0) + 0.05 \times \delta(\alpha,0)]$

For $\alpha = 1$: $0.65 \times 1 + 0.1 \times 0 + 0.04 \times 0 + 0.16 \times 0 + 0.05 \times 0 = 0.65$

For $\alpha = 0$: $0.65 \times 0 + 0.1 \times 1 + 0.04 \times 1 + 0.16 \times 1 + 0.05 \times 1 = 0.35$

Therefore, $\hat{x} = 1$.

4.2.2 Stacking

Stacking is an approach that uses different machine learning algorithms, trains them in parallel and combines them by training a meta-model. The final output is a prediction based on the different base model predictions. Stacking requires defining first the set of base machine learning models used, and the model that will be used to combine the chosen methods at the meta-level.

For example, for a classification problem as shown in Fig. 4.3, the base machine learning models chosen are decision tree, K-nearest Neighbor and Naïve Bayes. The meta-level model is a neural network. In this example, the neural network will take as inputs the outputs of our three models and will learn to return final predictions based on it.

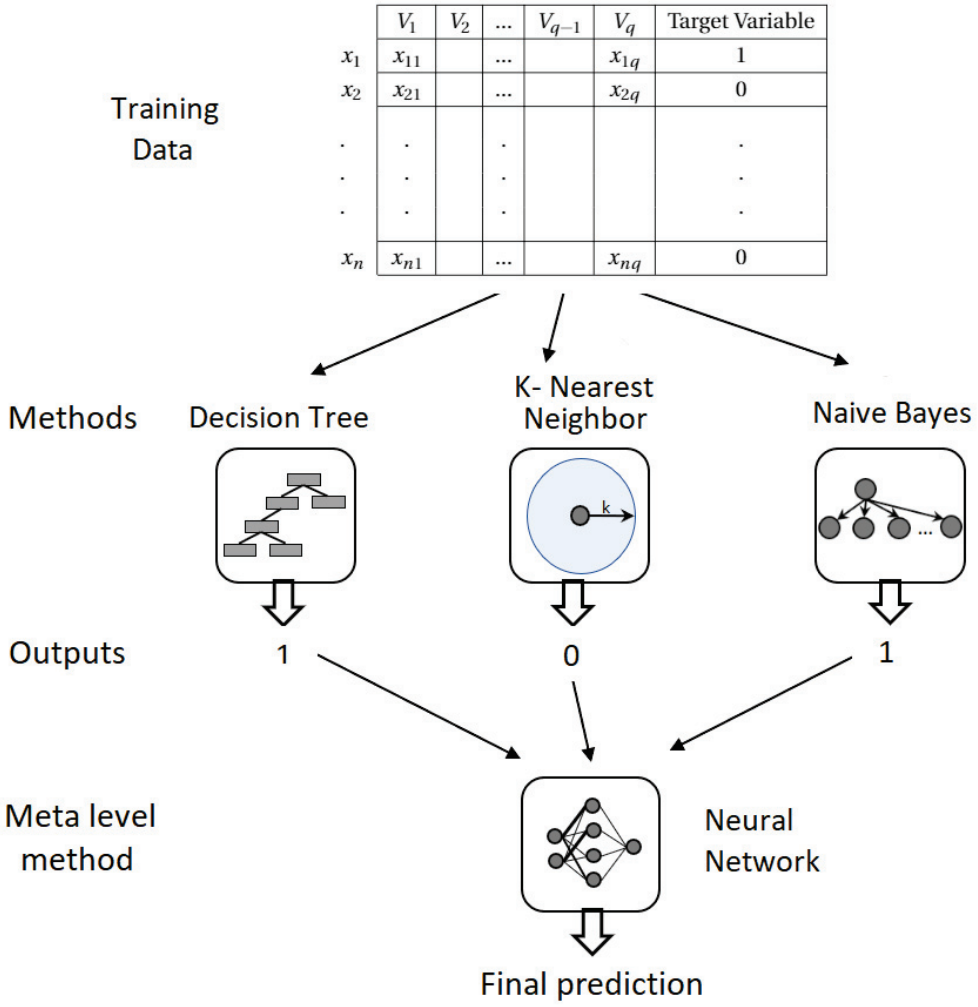


Fig. 4.3 Stacking diagram

4.2.3 Bagging and Boosting

Bagging and Boosting are two ensemble techniques different than stacking, by using one machine learning algorithm as a base learner instead of multiple different models. The diversification of bagging and boosting is done by using different data samples each time a new model is created.

Bagging stands for Bootstrap Aggregation. Bootstrap is a statistical technique that aims to generate L samples known as bootstrap samples, from the original data set using random selection with replacement. These samples will then be used to train L independent base models. Finally, the obtained models will be aggregated using simple voting for classification or average for regression, to form the final ensemble model.

Boosting method is a ensemble learning approach where multiple base models are aggregated to create a stronger model for classification or regression. Boosting fits the models iteratively or sequentially unlike Bagging were models are independent. In other words, the training of a model at a certain step in the Boosting process depends on the models fitted at the previous steps. Each new model focuses on the observations that were badly predicted in the previous model, by using a weighted sample of the data set.

Fig. 4.4 shows two diagrams that illustrate the difference between bagging (to the left) and boosting (to the right).

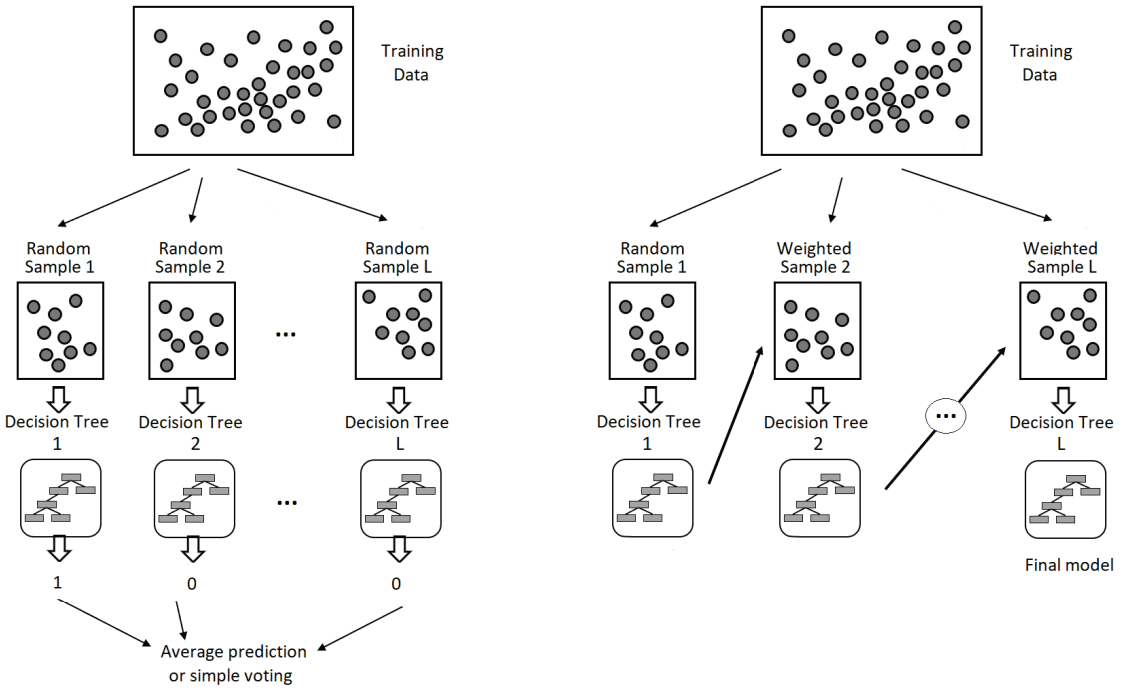


Fig. 4.4 Bagging vs. Boosting

4.3 K-MICHA: Theoretical Framework and Implementation

In this section, we will describe in detail our ensemble learning approach K-MICHA. The basic idea of K-MICHA is to combine the outputs of several methods in the fraud detection framework. K-MICHA handles the fraud detection problem in two phases. Phase I (diversification) consists of dividing the data set composed of n observations into three subsets (train and two test sets). Then, we apply the machine learning methods to the training set. In Phase II (integration), the obtained methods outputs and their corresponding actual target values are used to create the second training set. Thus, using this latter, a clustering based on the k-modes algorithm is applied to create clusters of observations where fraud probabilities are calculated in each cluster. Finally, the second test set will be used to validate the K-MICHA approach. The validation step is done by comparing K-MICHA with other methods. The approach is detailed hereafter.

Phase I - Diversification: The Training of the N Methods

In the first step, the data set is divided into 3 subsets. One train set, to build the N models that will be used in the combination, and two test sets. The first test will be used to validate the N models, and the second one to validate the combination approach.

In this phase, the first training set is used to train and build machine learning methods. This training set is from the original data set. It might differ from one method to another in terms of size and variables, according to the method's requirements. For example, a one-class classification method requires only one class of the target variable for the training.

The methods are trained independently from each other. When the training is done, the performance of the method (accuracy, sensitivity, and F_1 score) is evaluated using the first test set having p rows extracted from the original data. Formally, we consider N methods denoted by $M_1, M_2, \ldots, M_N$ for the combination approach.

Phase II - Integration: Combination of the N methods

This phase consists of the integration strategy that is used for the combination of the methods. K-MICHA uses unsupervised learning to tackle the imbalance problem in the second level classification. It allows fraud probabilities calculation based on imbalance ratios resulting from the unsupervised learning model.

First, we will create a second training set. This set is composed of p rows and $N+1$ columns resulting from the application of the N methods to the first test set.

Table 4.1 shows an example of the created training set. The N^{th} first columns represent the outputs of the N methods for the test set in the previous phase, taking values of 0 and 1. The $N+1^{th}$ column corresponds to the actual target values taking from the original data set.

Table 4.1 The created training set

	M_1	M_2	...	M_{N-1}	M_N	Target Variable
r_1	1	0	...	1	1	1
r_2	0	0	...	1	0	0
.	.		.			.
.	.		.			.
.	.		.			.
r_p	1	0	...	1	1	0

This obtained training set will be treated as a classification problem of N binary exploratory variables and a target variable. K-modes algorithm will be applied as an unsupervised learning method to find clusters of the N methods outputs.

Our purpose is to find groups of observations where the N methods give the same output. These observations are similar in terms of methods outputs. We aim to understand the way methods are generating predictions. For example, a weak method might still be a good predictor for a certain group of observations according to certain statistical or domain-related properties, where other stronger algorithms may not perform well. K-MICHA is built in a way to highlight these differences between the methods and combine their results. In the end, a weak method will only be considered in the clusters where its prediction is accurate, and vice versa. This is accomplished by calculating fraud probabilities based on the fraud distribution in the different clusters.

Generally, several clustering algorithms exist to group observations. In this work, we use the k-modes algorithm [164] since the N variables corresponding to the outputs of the method are categorical (0 and 1). This algorithm is summarized in the following section. The K-MICHA framework with both phases is illustrated in Fig. 4.5.

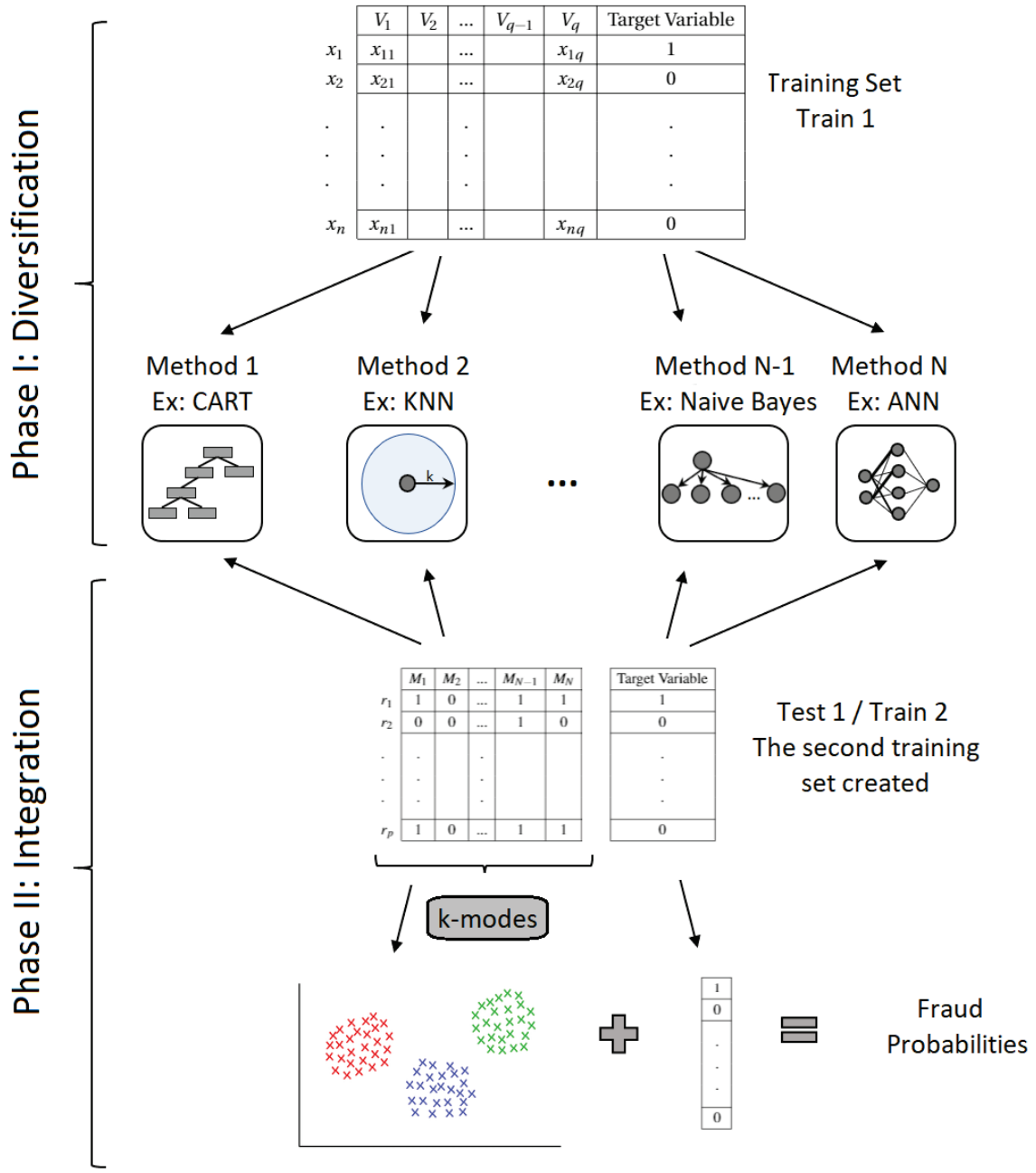


Fig. 4.5 K-MICHA framework diagram showing Phase I where the training of the N methods is done and Phase II where the integration is done using k-modes. An example is shown here with CART, KNN, ..., NB and ANN. The outputs of these methods form the second train set, where k-modes is applied to create clusters. Fraud probabilities are then calculated for each clusters using the target variable's actual values.

4.3.1 Clustering: The k-modes Algorithm

Cluster analysis or simply known as clustering is the process of partitioning a set of data observations into subsets called clusters [2]. These clusters group observations that are similar to one another, and dissimilar to the observations in other clusters. There are different clustering algorithms, which might result in a different set of clusters on the same data set. Thus the choice of the clustering algorithm is critical. A clustering example in $\mathbb{R}^2$ is provided in Fig. 4.6, where four clusters are generated based on two variables denoted Variable 1 and Variable 2, using a clustering algorithm. The figure shows how data observations are grouped and distinguished between 4 categories.

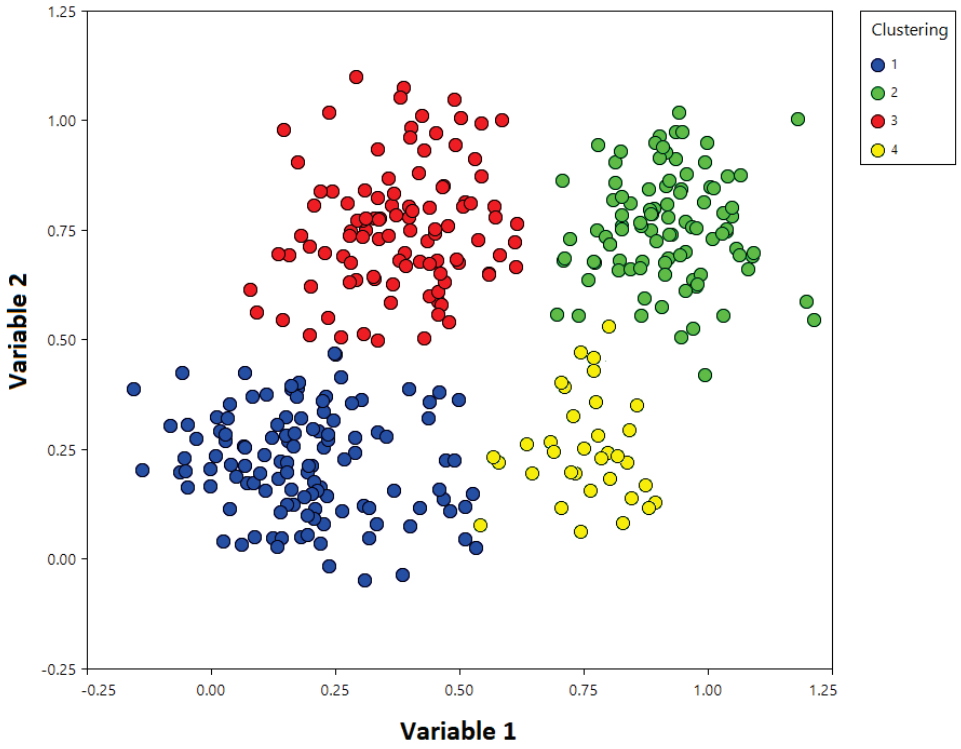


Fig. 4.6 Example of clustering results

Clustering has been used in many domains like business intelligence, image pattern recognition, web search, biology, and security.

K-modes is the clustering algorithm that K-MICHA is based on, to group the performance of the method accurately. K-modes is an extension of the k-means [164], which is the most common and straightforward clustering algorithm usually used to cluster the numerical variables. K-modes is a variant of this algorithm that is applied to categorical variables. K-modes use the modes of variables instead of the means, respectively the

simple matching dissimilarity measure to find the clusters instead of the euclidean distance.

Definition 1. *The **mode** of a variable M_i is the most frequently occurring value.*

Example 1. *Let $\mathcal{X}=(0,0,1,1,1,0,1)$ be a variable. $mode(\mathcal{X})=1$. Note that, in some cases, the mode is not unique.*

Definition 2. *The **mode of a set** $\mathcal{R}=\{r_1,\ldots,r_p\}$ described by categorical variables $\mathcal{M}=\{M_1,\ldots,M_N\}$ is a vector $(m_1,\ldots,m_N)\in\mathbb{R}^N$ where m_i is the mode of the variable M_i.*

Example 2. *Let $\mathcal{R}$ be a set of observations defined by 6 variables as follows.*

$\mathcal{R}$ =

M_1	M_2	M_3	M_4	M_5	M_6
0	1	1	1	0	1
1	1	1	0	0	0
0	1	0	0	0	0
0	0	0	0	0	0
1	0	0	0	1	0
1	1	1	1	0	1
0	1	1	0	1	0

Then, the mode of the set $\mathcal{R}$ is $mode(\mathcal{R})=(0,1,1,0,0,0)$.

Definition 3. *The **simple matching dissimilarity** between two rows r_i and $r_j\in\mathbb{R}^N$ represents the total mismatches of the corresponding variable categories of the two observations (rows). This is defined as follows:*

$$d(r_i,r_j)=\sum_{l=1}^{N}\delta(r_{il},r_{jl})$$

where

$$\delta(r_{il},r_{jl})=\begin{cases}0 & if \quad r_{il}=r_{jl}\\ 1 & if \quad r_{il}\neq r_{jl}\end{cases}$$

Example 3. *Let $r_1, r_2 and r_3$ be 3 rows of observations in a set $\mathcal{R}$ such that:*

$$r_1=(0,1,0,0,1)\quad r_2=(0,1,1,1,1)\quad r_3=(0,1,1,0,1)$$

$$\textit{Then} \quad d(r_1, r_2) = 2, \quad \textit{and} \quad d(r_1, r_3) = d(r_2, r_3) = 1$$

the pseudocode of k-modes is given in Algorithm 1 hereafter. In our case, we want to cluster "similar" observations together. This means that the distance between any two rows in the same cluster will be equal to zero. Thus, we need a higher number of clusters that we can have which is equal to 2^N.

Algorithm 1 k-modes

1: Input: Training set: $\mathcal{R} = \{r_1, \ldots, r_p\}$, Number of clusters: k

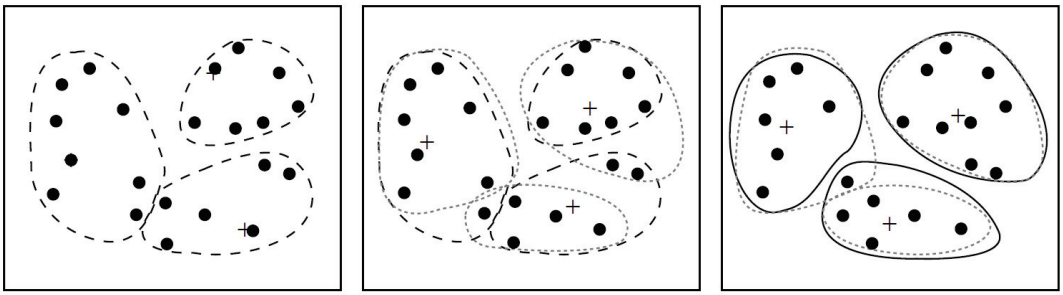
2: Initialization: Cluster centroids (modes): $(c_1, \ldots, c_k)$ initialized randomly, where $c_i \in \mathbb{R}^N$

3: For each $r_i \in \mathcal{R}$, assign r_i to the cluster C_j of center c_j, $j = 1, \ldots, k$, such that $d(r_i, c_j)$ is minimal.

4: For each Cluster C_j, calculate the new mode c'_j.

5: Repeat steps 3 and 4 until the centroids remain unchanged. (the optimal solution is found)

6: Output: Final clusters: $C_1, \ldots, C_k$



(a) Initial clustering **(b)** Iterate **(c)** Final clustering

Fig. 4.7 Example of k-means or k-modes iterations [2]

Fig. 4.7 shows a small example of clustering iterations using k-means or k-modes. In a first step, we initialize random centers (mean or mode) and find their corresponding clusters (a). In each following iteration, we update the centers and the clusters (b) until we reach a final optimal solution (c).

4.3.2 Fraud Probabilities and Approach Validation

In K-MICHA, after the clusters generation, we calculate fraud probabilities in each cluster. In other words, this probability represents the likelihood of the cluster's observations to be fraudulent.

For any final cluster C_j having q observations $r_1,\dots r_q$, the fraud probability of that cluster is calculated as follows:

$$P_j = \frac{\sum_{i=1}^{q} y_i^j}{q}$$

Where y_i^j denotes the actual value of the target variable for the i^{th} row in cluster C_j.

Example 4. *Let C_{10} be a cluster of 23 observations, we have*
$Y^{10} = (1,0,1,1,1,0,1,0,1,1,0,0,0,1,1,0,1,0,1,0,1,1,1)$*, Then* $P_{10} = \frac{\sum_{i=1}^{23} y_i^{10}}{23} = 0.609$

The final fraud detection can be done by defining a decision threshold for this probability. After assigning a fraud probability to each cluster, new observations can be predicted. To evaluate the performance of K-MICHA, we use the second test set and the three performance measures: accuracy, sensitivity and F_1 score. Firstly, we find the N outputs of the new observations. Then, we assign the obtained outputs row to the corresponding cluster and its fraud probability (respectively decision). A threshold might be used at the end, in case we need a binary output or a definitive prediction of fraud rather than just probabilities.

4.3.3 Threshold Choice

Some of the methods used for the combination, generate fraud probabilities instead of a class value 0 or 1, like LR or ANN. K-MICHA is built in a way to take inputs as 0 or 1 precise classes, not class probabilities. It became essential to find sometimes thresholds to cut off the fraud probabilities at, to define the decision rule.

Choosing this threshold is very crucial, especially since it affects all the performance measures that we are considering in our experiments like the accuracy, sensitivity and F_1 score. The threshold chosen should generate results that resemble a compromise between a high fraud detection rate and a low rate of false alarms. Since in all our experiments, we are considering the F_1 score to be the performance measure that reaches this compromise, the choice of the threshold should be based on this measure. In our

experiments, the threshold corresponds to the cut off the value that leads to the highest F_1 score obtained for the used method.

Fig. 4.8 provides an insight into the change of the performance measures according to different thresholds values. This graph also represents a case of class imbalance. When the threshold increases, the accuracy increases, and the sensitivity decreases. The F_1 score reaches a peak at the threshold that achieves the highest sensitivity possible at the highest accuracy, as shown in figure hereafter.

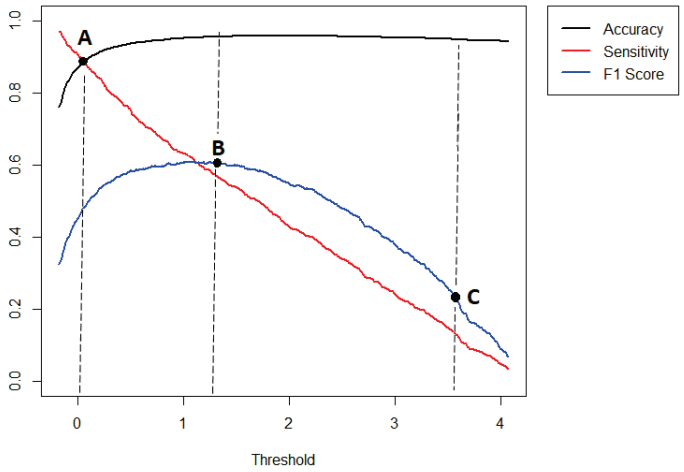


Fig. 4.8 The change of accuracy, sensitivity and F_1 score with different thresholds

Actually, for any classification case with no imbalance, a threshold can be easily chosen using only the accuracy. In that case, the accuracy curve reaches a noticeable peak, unlike the one in Fig. 4.8. This is a very clear proof that the accuracy is misleading in an imbalance case, where it already starts from a value 0.75 and keeps increasing. The sensitivity decreases when the threshold becomes higher. Without considering the F_1 score, it might look that the best choice for threshold is a very low one, **point A** in Fig. 4.8, where the sensitivity and accuracy are both very high. However, as discussed in Section 1.3, a tiny change in accuracy is crucial because it hides a large number of incorrectly classified observations from the majority classes. A better or suitable choice of threshold is **point B** where the highest accuracy is reached with the highest possible sensitivity rate. On the other hand, a very high threshold like **point C** would reach the highest accuracy possible, but with very low sensitivity and F_1 score. Thus, in the example in Fig. 4.8, the best threshold is chosen at **point B**.

Chapter 5

Evaluation of K-MICHA

5.1 Design of the Experiment

In this chapter, we present three case studies for the application of K-MICHA in the financial sector. The purpose of these applications is to prove the efficiency of K-MICHA by comparing it to other single classifiers and other ensemble learning methods existing in the literature review. We also present an implementation of K-MICHA in a big data framework. The purpose of this implementation is to compare the traditional data analytics and the Big Data analytics in terms of the performance of the algorithms, the processing speed and the volume of data analyzed. We will first present an overview of the plan of the experiment, then the results of each case study and the H2O experiment, and finally an overall discussion and comparison.

For K-MICHA's evaluation in terms of performance, we applied it on three financial fraud data sets detailed hereafter, where different machine learning methods are combined each time.

Table 5.1 Description of the three case studies

Case Study	Financial Sector	Combined Methods [1]
1	**Credit card fraud:** Synthetic data set of credit card transactions [2].	1. OCCoS 2. CoSKNN 3. LR 4. ANN
2	**Mobile payment fraud:** Simulation based on real data set extracted from a mobile money service financial logs [165].	1. OCCoS 2. CoSKNN 3. LR 4. CART 5. NB
3	**Auto insurance fraud:** Synthetic labeled data set of car accidents insurance claims [3].	1. CoSKNN 2. LR 3. CART 4. NB 5. ANN

[1] See the List of Abbreviations on page xxi for the methods mentioned in this table
[2] Available at http://packages.revolutionanalytics.com/datasets/
[3] Extracted from https://databricks-prod-cloudfront.cloud.databricks.com

Table 5.1 shows the combined methods for each case study. The first (1) is a credit card fraud detection with imbalance ratio of 5.96%. The second (2) is mobile payment fraud detection with an imbalance of 5% and the third (3) is auto insurance fraud detection with an imbalance of 12.2%.

For the comparison between traditional analytics and Big Data analytics, we apply K-MICHA to the credit card fraud data set, by combining LR, ANN, NB and CART.

All the methods used in this chapter are presented and detailed in Section A.1, except for CoSKNN which is presented and detailed in chapter 3. Moreover, OCCoS is a simple one-class classification approach based on cosine similarity, using only the fraud observations. In this approach, we evaluate the average "similarity" between the test observation and all fraud observations. Then the classification is done according to a certain threshold specified as mentioned in section 4.3.3 (Page 63).

In all three case studies, we compare K-MICHA with stacking using voting, weighted voting, logistic regression, and CART. We also compare K-MICHA with Adaboost and random forest, we are interested in comparing their performance with K-MICHA. Even though these two methods only use one model and they do not consist of combining multiple models like K-MICHA. Table 5.2 provides more details about the method we compared our approach with.

It is important to note that some of the methods applied whether as a single classifier or as combination might require certain mathematical or statistical assumptions that we should abide by. For example, the logistic regression requires to have little or no multicollinearity among the explanatory variables. In other words, these variables should not be too highly correlated with each other. This assumption is tested using the Variance Inflation Factor (VIF). This test is based on the linear correlation coefficient resulting from the multiple regression applied to each one of the predictive variables in terms of the others. If one of the variables has a strong correlation with at least one other variable, it will result in a high VIF. A VIF higher than 10 is a signal that we have a multicollinearity problem in the model. The assumptions and requirements are validated each time we use them.

Table 5.2 Ensemble learning methods compared with K-MICHA

Method	Description
Voting	The simple voting combination technique is described in Section 4.2.1. If the majority of the methods predict fraud, the voting prediction will be a fraud.
Weighted Voting	In the weighted voting technique, specific weight are assigned to each method to highlight its performance (Section 4.2.1). In our applications, we choose the F_1 score as weights for the methods since it is the most important performance measure.
Logistic Stacking	This method represents a stacking approach where the meta level method used is a logistic regression where the target variable is the actual value of the fraud. The linear coefficient obtained will be then used to estimate fraud, and the classification will be done using a specific threshold.
CART Stacking	This method represents a stacking approach where the meta level method used is the CART tree algorithm.
Adaboost	Adaptive Booting known as Adaboost is a boosting algorithm introduced by Freund and Schapire [166]. The ensemble model is defined as a weighted sum of the L weak classifiers. Finding the weights is done by an iterative optimisation process. In our experiment, the base classifier is the CART algorithm.
RF	Random Forest is a specific bagging algorithm using decision trees as base [2]. The key is to generate a large number of uncorrelated trees by using different data samples, and different set of variables. The final decision is made according to votes for classification and means for regression.

5.2 Case Study 1: Credit Card Fraud Detection

For the credit card data set, we choose four machine learning algorithms: OCCoS, CoSKNN, LR and ANN. We combine them using K-MICHA and compare it with other stacking approaches. In the following, we describe the data set, the training, and testing, and finally the results.

5.2.1 Data Description

Data used is a credit card fraud labeled data set [4], that was used before in [131]. It contains ten million credit card transactions representing the observations described by 9 variables. These variables are described in the following.

1. **custID** is an integer variable that represents the customer ID. It identifies a unique number for each customer. This variable was later removed for not having any relevance in fraud detection since in this data set, there is no record for the same customer making more than one transaction at this particular time. In other data sets, if the work of fraud detection is based on a profiling method, this variable would be crucial.
2. **gender** is the variable representing the customer's gender where 61.7% are male and 38.3% female. It is a categorical variable where 1 denotes male and 2 denotes female.
3. **state** is a discrete quantitative variable that ranges from 1 to 51 representing the state in which the customer lives in the United States.
4. **cardholder** is a discrete variable that represents the number of cards that the customer holds. In our data set, 2.95% of the customer hold 2 cards, where the rest have only one card.
5. **balance** is a continuous variable indicating the balance on the credit card in USD. Fig. 5.1 and 5.2 shows respectively the histogram and the boxplot of this variable. In both figures, we can see outliers, where the amount is much higher than normal values for other customers. Even though these are outliers, it is important to include them in the fraud detection analysis. First, these observations represent the most important and privileged clients in the bank, and their satisfaction is essential. Second, These customers are usually fraud targets and victims.

[4] Available at http://packages.revolutionanalytics.com/datasets/

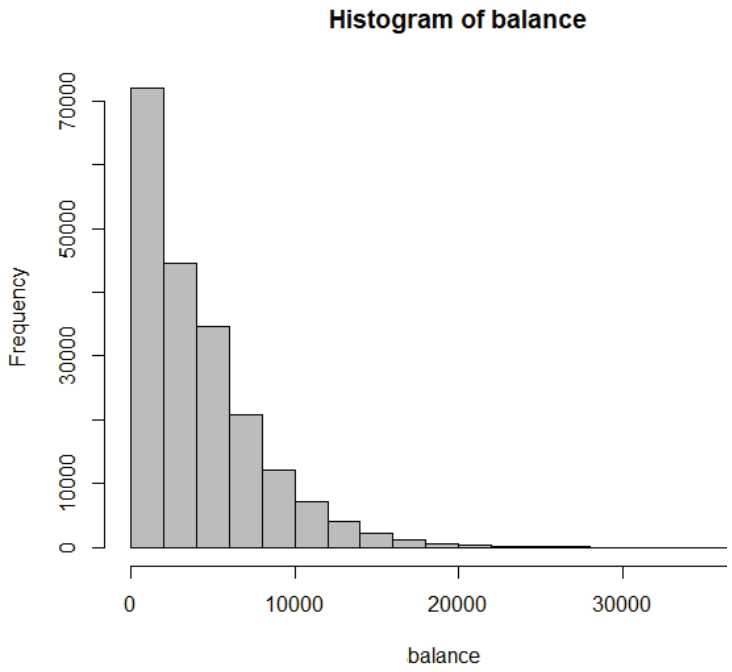


Fig. 5.1 Histogram of balance

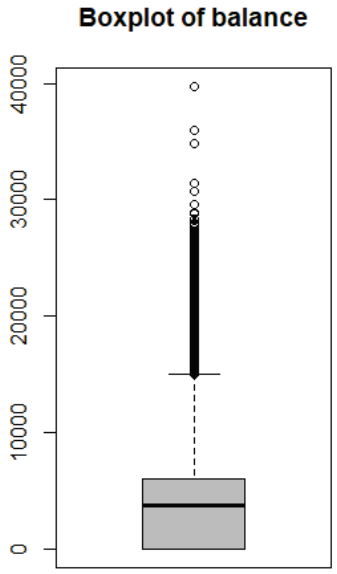


Fig. 5.2 Boxplot of balance

6. **numTrans** is a discrete variable that represents the number of transactions made to date by the customer.
7. **numIntlTrans** is a discrete variable that represents the number of international transactions made to date by the customer.
8. **creditLine**: is a discrete variable that denotes the customer's credit limit.

Table 5.3 Statistical summary of numTrans,numIntlTrans and creditLine

Variable	Minimum	1^{st} Qu.	Median	Mean	3^{rd} Qu.	Maximum
numTrans	0	10	19	28.9	39	100
numIntlTrans	0	0	0	4.0	4	60
creditLine	1	4	6	9.1	11	75

Table 5.3 shows a statistical summary of these three variables. The mean for local transactions is 28.9, whereas for the international ones it is 4. It is logical for international card transactions to be much less than the local ones. These variables are also important because they differentiate clients according to their business size, and therefore their importance to the banks.

9. **fraudRisk** is the categorical target variable. It is a binary variable indicating the labels: fraud denoted by 1 and legitimate denoted by 0. In the data set, 596,014 are fraudulent transactions and 9,403,986 are legitimate. This data exemplify the class imbalance problem with 5.96% fraud cases.

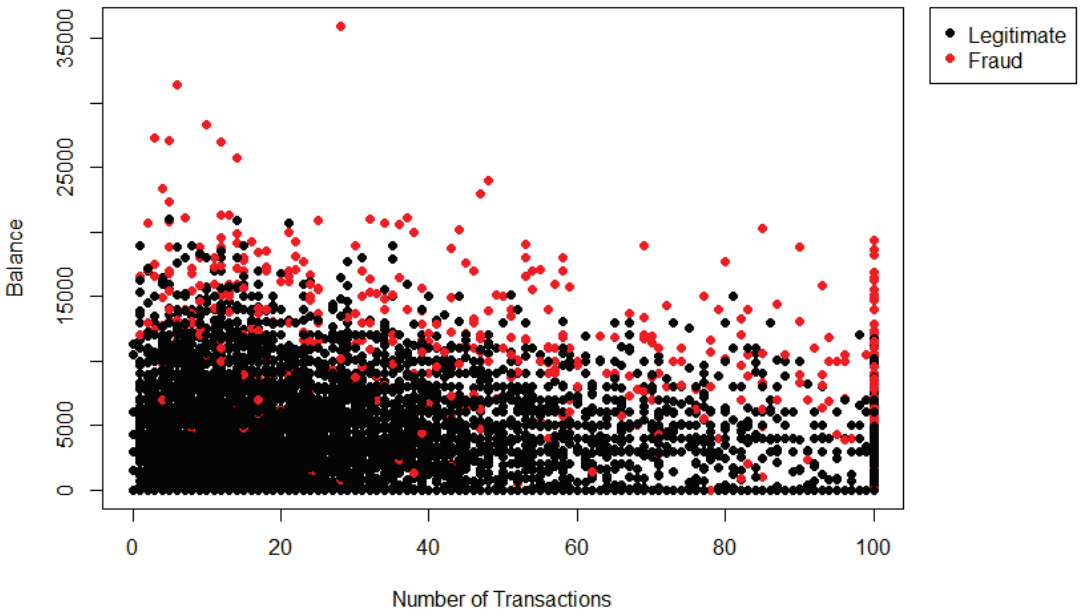


Fig. 5.3 Scatter plot of balance against number of transactions

The scatter plot in Fig. 5.3 shows the distribution of the number of transactions against the balance for a part of the data set, where fraud observations are colored in red and legitimate ones are colored in black. The plot shows the small ratio of fraud in the data set.

5.2.2 Implementation and Results

The first training set differs according to the methods. For example, OCCoS uses only fraud observations. For time consumption reduction purposes, we did not run the whole data set, only a part of the ten million transactions is used while keeping the same imbalance ratio. Table 5.4 shows the description of each data partition and the thresholds used for each method.

Table 5.4 Data partition and thresholds for the methods - case study 1

Method	Train 1	Test 1 / Train 2	Threshold
OCCoS	9536 fraud observations	20000 observations	0.35
CoSKNN	159999 observations		1.43
ANN			1.2
LR			0.34

Fig. 5.4 shows the change of accuracy, sensitivity and F_1 score according to different thresholds, for all the methods used in this case study. These graphs prove the threshold's choices for those methods.

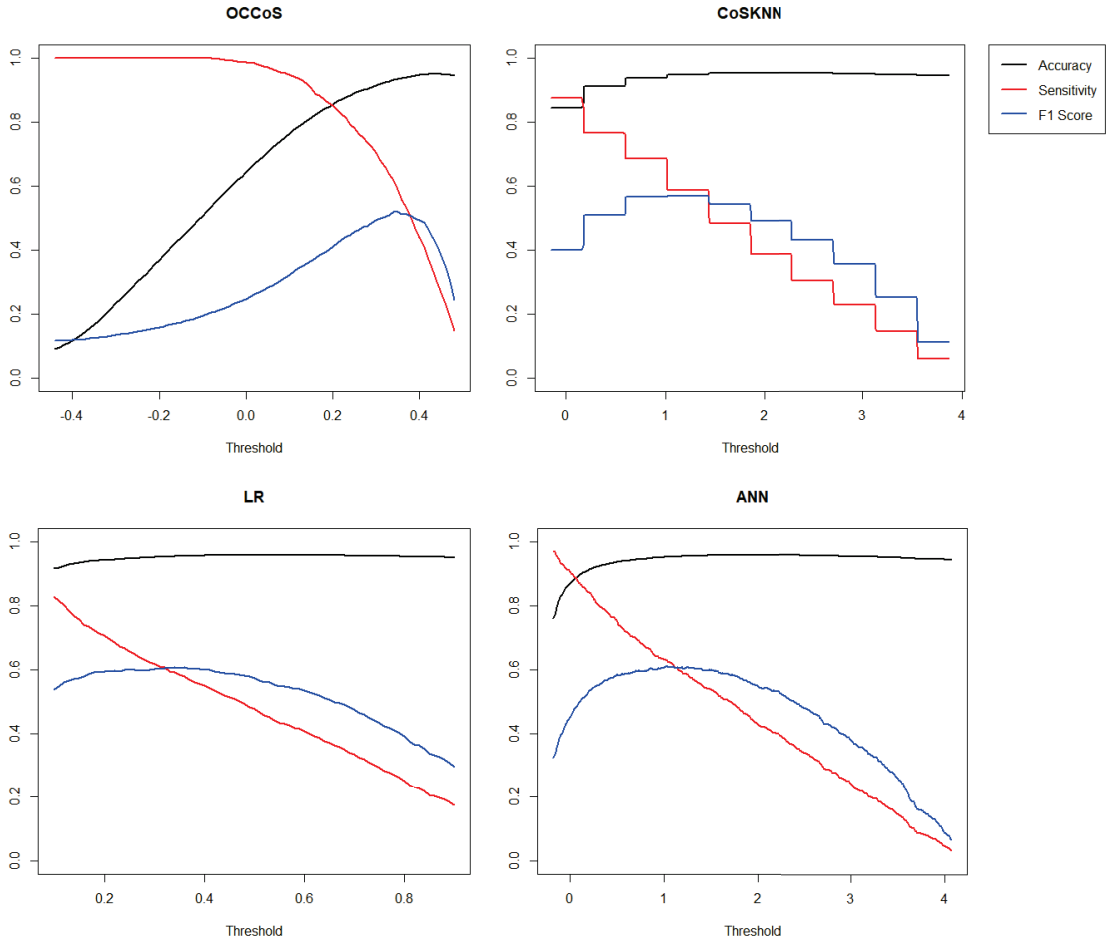


Fig. 5.4 Performance measures variations according to different thresholds - case study 1

For the OCCoS, only the fraud classes were used and the classification was done in the testing step, where the similarity between the new observation and the fraud samples is evaluated. For the CoSKNN, the number of neighbors chosen was 10. The fraud is then evaluated using their classes. For both these methods, data is first standardized, i.e. the variables are transformed to have a mean of zero and a standard deviation of 1 following the formula:

$$x_{new} = \frac{x - \mu_x}{\sigma_x}$$

where μ_x and σ_x are resp. the mean and standard deviation of x.

For the LR, The VIF test was applied to the data set's seven explanatory variables to test the multicollinearity. Table 5.5 shows the results for this test. All the variables have VIF less than 10, i.e. the collinearity problem is not present in this case study. Thus, all variables are all kept for the training of LR.

The LR model results are presented in Table 5.6 with the coefficients for each variable. According to Table 5.6, the fraud probability in LR is euqal to: $\frac{e^{X\alpha}}{1+e^{X\alpha}}$ where α is the vector of coefficients and X is the vector of variables.

Table 5.5 VIF test results for LR - case study 1

Variables	VIF
balance	1.070
cardholder	1.000
creditLine	1.070
gender	1.002
numIntlTrans	1.002
numTrans	1.002
state	1.002

Table 5.6 LR coefficients - case study 1

Variable	Coefficients
Constant	−9.915
balance	0.0004
cardholder	0.456
creditLine	0.095
gender	0.612
numIntlTrans	0.032
numTrans	0.047
state	−0.013

For the ANN, data is first normalized using the minmax normalization, meaning that variables are scaled to a range of [0,1], as follows:

$$x_{new} = \frac{x - \min(x)}{\max(x) - \min(x)}$$

where $\min(x)$ and $\max(x)$ are resp. the minimum and maximum of x. A multilayer feedforward network architecture is applied and trained with one input layer of 7 nodes representing the explanatory variables, a hidden layer of 3 nodes, and an output layer providing the fraud probability as shown in Fig. 5.5. The algorithm used to adjust the weights is the resilient backpropagation algorithm.

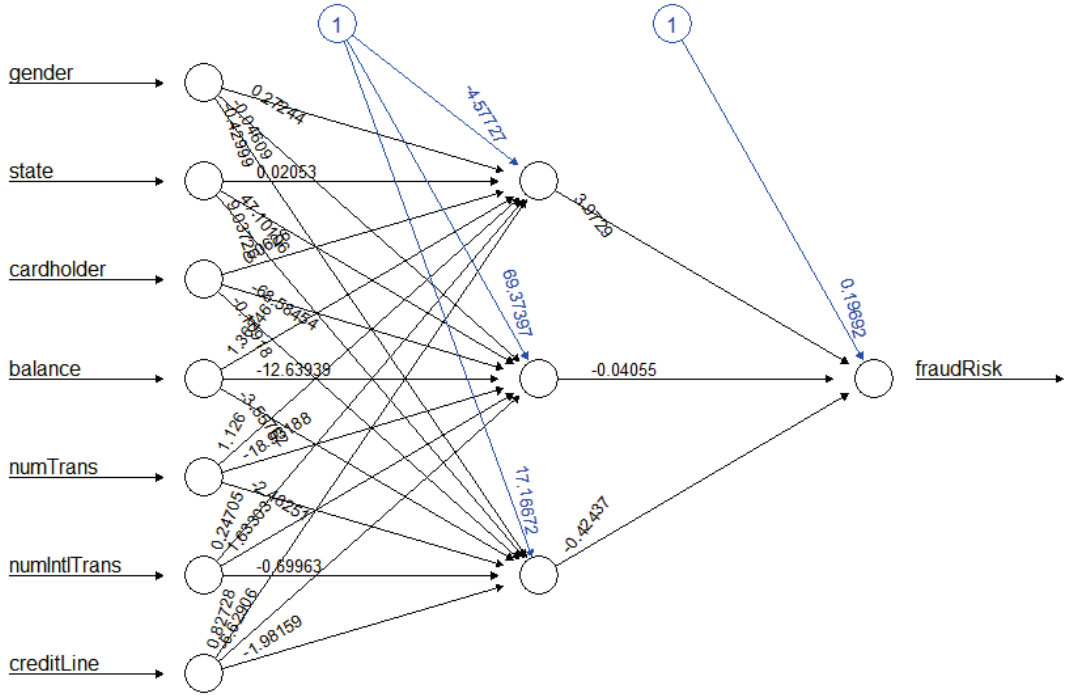


Fig. 5.5 ANN architecture - case study 1

These methods are built using the first train set (Train 1) and evaluated using the first test set (Test 1), which will allow the training to the combination system. In Phase II the k-modes algorithm is applied to find clusters of the data points and to calculate class probabilities as shown in Table 5.7.

Table 5.7 The clusters characteristics - case study 1

Cluster	Cluster Center (modes)	Size	Fraud Probability
1	(0 , 0 , 1 , 1)	134	0.254
2	(1 , 1 , 1 , 0)	179	0.587
3	(1 , 1 , 1 , 1)	775	0.675
4	(0 , 0 , 0 , 0)	18084	0.018
5	(0 , 0 , 1 , 0)	133	0.158
6	(1 , 0 , 1 , 0)	6	0.333
7	(1 , 1 , 0 , 1)	77	0.429
8	(1 , 0 , 0 , 0)	11	0.273
9	(1 , 1 , 0 , 0)	66	0.470
10	(0 , 1 , 0 , 1)	6	0.667
11	(1 , 0 , 0 , 1)	9	0.444
12	(1 , 0 , 1 , 1)	15	0.400
13	(0 , 0 , 0 , 1)	482	0.193
14	(0 , 1 , 1 , 1)	9	0.556
15	(0 , 1 , 0 , 0)	2	0.500
16	(0 , 1 , 1 , 0)	12	0.250

Table 5.7 shows the characteristics of the clusters that were obtained for case study 1. The most discriminating clusters are clusters number 1, 2, 3, 4, 5, 10 and 13, these clusters either contain most data points or extremely low or high fraud probability. The centers represent the method's predictions that are identical in each cluster respectively ANN, LR, CoSKNN, and OCCoS. The total number of clusters obtained is $2^4 = 16$. According to Table 5.7, the biggest cluster is cluster number 4 which contains 18084 observations. In this cluster, all the methods predictions were 0, which is logical due to the imbalance. The second big cluster is the one having 775 observations where all methods predictions were equal to 1, meaning a fraudulent transaction. These two clusters have fraud probabilities respectively equal to 0.018 and 0.675. The different clusters show the difference one method can contribute to the fraud probability. For example, the OCCoS method increases the fraud probability by 8.8% when all other methods predict fraud according to clusters 2 and 3. Similarly, CoSKNN increases the fraud probability by 24.6% according to clusters 3 and 7.

In the analysis above, we were trying to understand the strengths and weaknesses of each method inside each cluster. We illustrate in Fig. 5.6 the scatter plots of the variables

colored by the clusters to relate to the data set's original variables. It is not straightforward to analyze 16 clusters in these plots. However, two plots are remarkable, where certain groups of observations can be differentiated by two or three colors representing different clusters. These two plots are the ones that show the variable balance against numTrans and balance against creditLine. These plots are presented in Fig. 5.7 and 5.8, where three clusters are easily distinguished. These clusters are numbers 3, 4 and 13, that correspond to the clusters with a very high and very low probability according to Table 5.7. These plots also show the importance of the variables balance, creditLine, and numTrans, since these are the variables that highlight the different clusters.

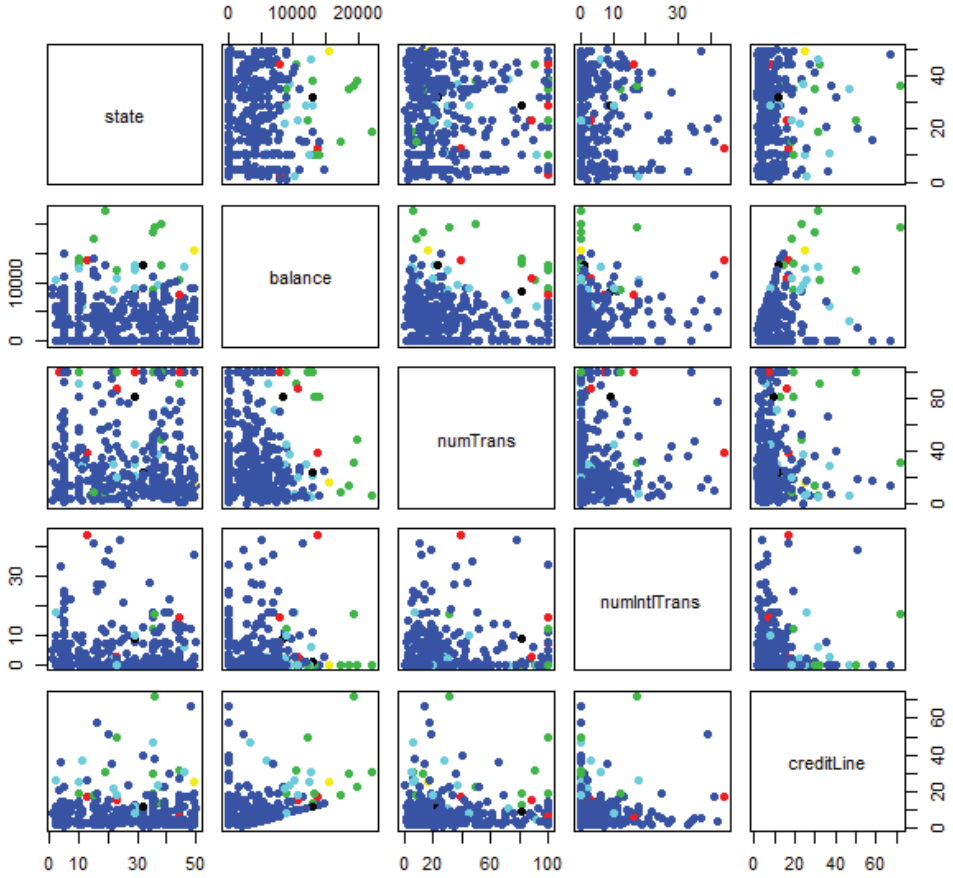


Fig. 5.6 Pairs of the explanatory variables scatter plots colored by the clusters - case study 1

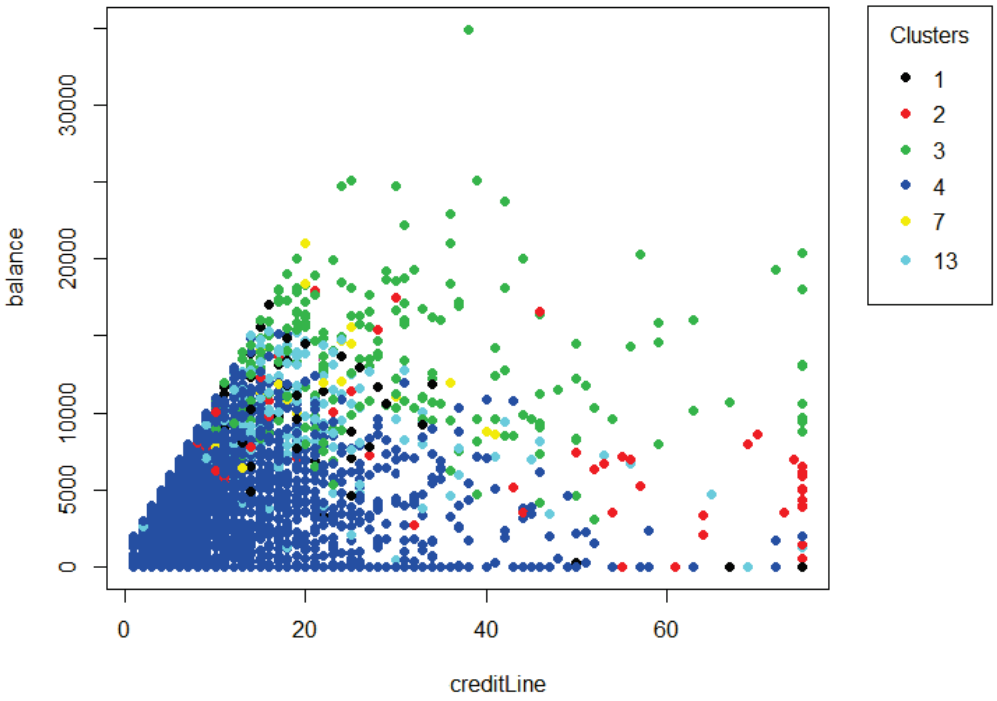


Fig. 5.7 Scatter plot of creditLine against balance colored by the clusters

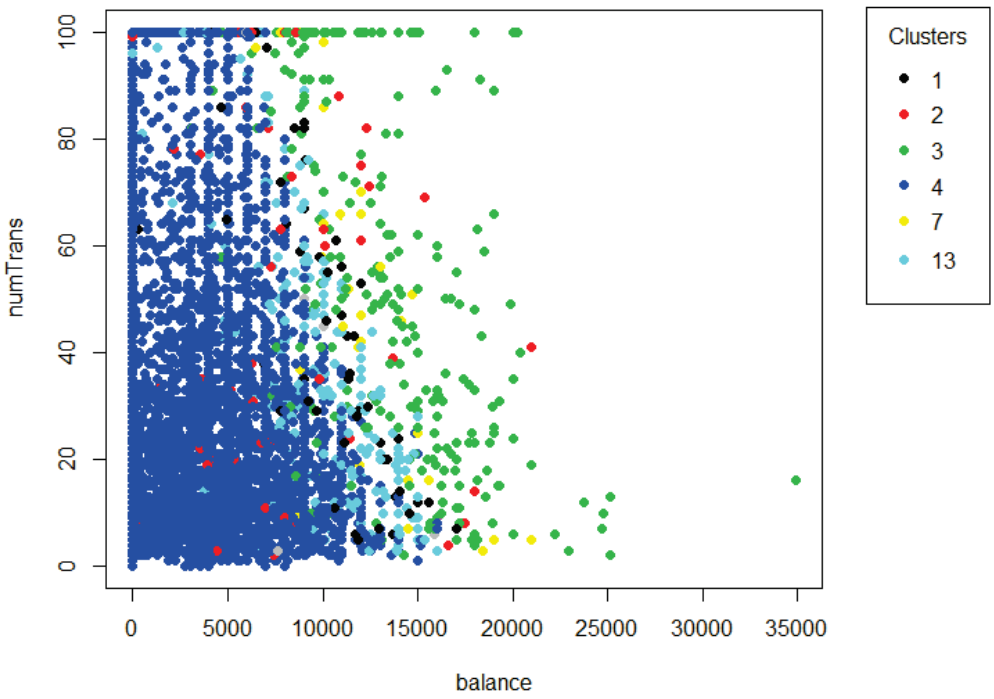


Fig. 5.8 Scatter plot of balance against numTrans colored by the clusters

5.2.3 Validation and Comparison

The validation of K-MICHA is done using a second test set of 20000 observations. Voting and weighted voting are used where the weights are the F_1 scores of the methods. Logistic regression stacking was applied where the target variable fraudRisk, and where the explanatory variables are the outputs of the method. The logistic coefficients of the methods are presented in Table 5.9 where the highest coefficient corresponds to the ANN method. The VIF test was applied to the outputs of the methods representing the explanatory variables to test the multicollinearity. Table 5.8 shows the results for this test. We remark that ANN and LR have high VIF, which means that the removal of one of them is necessary. We kept the ANN method. The LR model results are presented in Table 5.9 with the coefficients for each method.

Table 5.8 VIF Test Results for LR Stacking - case study 1

Variables	VIF
ANN	18.10
LOG	18.71
CoSKNN	3.47
OCCoS	1.85

Table 5.9 LR Stacking Coefficients - case study 1

Variable	Coefficients
Constant	-3.79
ANN	2.16
CoSKNN	1.17
OCCoS	1.49

A CART stacking algorithm is applied, where the target variable is fraudRisk and the explanatory variables are the methods outputs. Table 5.10 shows the importance of the explanatory variables, i.e. the methods, calculated using the CART algorithm and scaled to 100 for comaprison purposes.

Table 5.10 Methods importance according to CART stacking - case study 1

Method	Importance
LR	100
ANN	94.186
CoSKNN	62.316
OCCoS	20.628

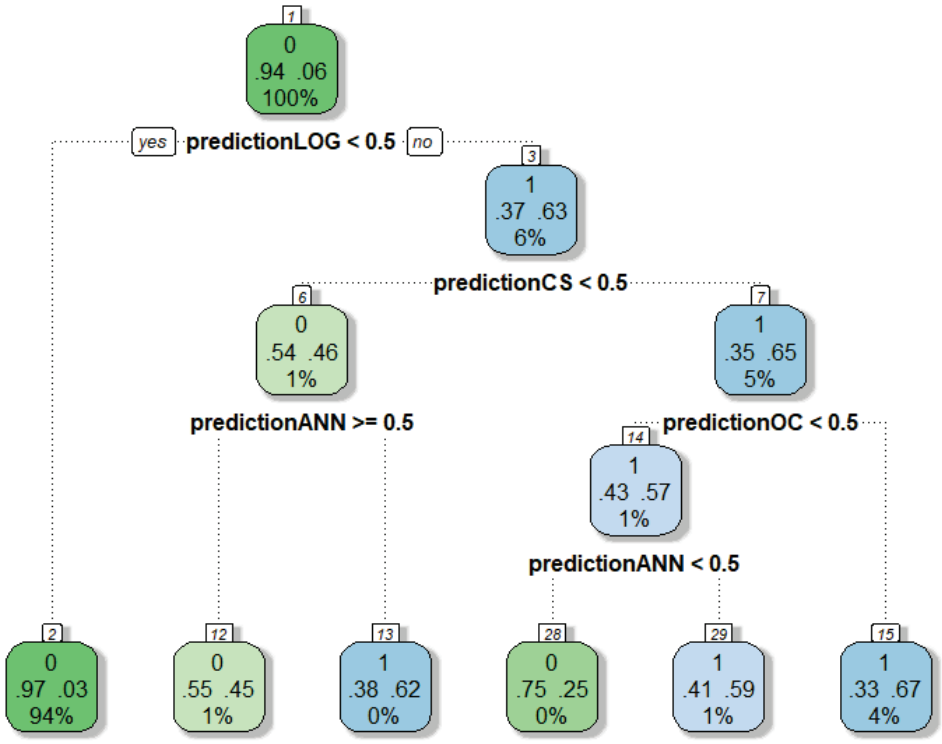


Fig. 5.9 CART stacking tree - case study 1

Fig. 5.9 shows the plot of the final tree obtained. This tree has 6 leaves with class predictions of 0, 0, 1, 0, 1 and 1 respectively.

For example, if the LR prediction of a certain observation is higher than 0.5, then we look at the CoSKNN prediction, if it is less than 0.5, we look at the ANN prediction, if it is less than 0.5, it falls in the node 13 and thus it is classified as a fraud observation. According to both the table and the figure above, LR is the most important method in the case study. It has a great contribution to the partition of the data, the second is the ANN, then CoSKNN, and last OCCoS with a very low contribution.

For the Adaboost algorithm, 10 trees were generated to obtain the final one. The weights of these trees are respectively: 1.68, 1.28, 1.13, 1.02, 0.93, 0.86, 0.79, 0.75, 0.70 and 0.65. For the random forest, 500 trees were generated. The whole forest error rate is equal to 4.17%. The variable importance according to the forest is shown in Table 5.11.

Table 5.11 Variables importance according to RF - case study 1

Method	Importance
balance	100
creditLine	52.13
numTrans	49.28
state	24.42
numIntlTrans	18.30
gender	3.09
cardholder	1.66

In the following, we present the results of the performance measures obtained. Using these results, we will compare K-MICHA with each method alone and with other ensemble learning including Adaboost and RF. Table 5.12 shows the results of the methods according to accuracy, sensitivity and F_1 score.

Table 5.12 The performance K-MICHA vs. other approaches - case study 1

Method	**Accuracy**	**Sensitivity**	F_1 **score**
OCCoS	93.5%	0.59	0.52
CoSKNN	94.7%	0.59	0.56
LR	95.4%	0.59	0.60
ANN	95.4%	0.59	0.60
K-MICHA	**95.3%**	**0.64**	**0.62**
Voting	95.6%	0.56	0.60
Weighted Voting	95.6%	0.59	0.62
Logistic Stacking	95.3%	0.50	0.56
CART Stacking	95.6%	0.53	0.59
Adaboost	94.9%	0.49	0.53
RF	95.9%	0.45	0.56

According to Table 5.12, All accuracy values for single or combination methods are close around 94% which is very logical to obtain when the imbalance ratio in the data is around 6%. For the single classifiers, using the thresholds mentioned in the previous section, we obtained the same sensitivity for all methods, with different accuracy and F_1 score values. It is obvious in this case, that both ANN and LR are the best models, for having the highest F_1 score with a sensitivity of 0.59. Higher accuracy rates were obtained when

applying the combination methodologies. These high values were obtained at the cost of lower sensitivity rates, except for K-MICHA reaching the highest F_1 score along with weighted voting. However, with the same high F_1 score equal to 0.62, K-MICHA achieved a 5% higher sensitivity. The Adaboost and RF algorithms did not improve the sensitivity nor the F_1 score. This is due to the reason that the CART algorithm was not performing well on the data set as a single classifier.

5.3 Case Study 2: Mobile Payment Fraud Detection

For the mobile payment fraud data set, we choose five machine learning algorithms: OCCoS, CoSKNN, LR, CART, and NB. We combined them using K-MICHA and compared it with the same stacking approaches used in the previous application for comparison. In the following, we describe the data set, the training, and testing phases, and finally the results.

5.3.1 Data Description

Data used is synthetic labeled data set of mobile money transactions [165]. The data is a simulation based on a real sample of one-month financial logs extracted from a mobile money service implemented in an African country. It contains 6,362,620 transactions and 11 variables that are described in the following.

1. **step** is a discrete variable representing a unit of time in the real world where 1 step denotes 1 hour. In the whole data set, the steps start at 1 and end at 743. For fraud detection, this variable was removed.
2. **type** is a categorical variable representing the type of the transaction made. Table 5.13 shows the frequencies of this variable's categories.

Table 5.13 Categories frequencies of the variable type

cash-in	**cash-out**	**debit**	**payment**	**transfer**
22.01%	35.16%	0.65%	33.81%	8.37%

Cash-in refers to paying in cash to the account. Cash-out refers to withdraw cash from an account. The debit is similar to cash-out, it represents sending the money from the mobile money service to a bank account. Payment is the process of paying for goods or services. Transfer represents sending money to another user of the service through the mobile money platform.

3. **amount** is a continuous variable that shows the amount of the transaction in the local currency. Table 5.14 shows the distribution of this variable using quantiles. The last three statistics were chosen to show the highly skewed distribution of this data. According to this table, half of the transactions made are less than 74,872. However, just 2% of the transactions are of an amount ranging from 1,019,958 to 92,445,516, which represents a large number of outliers, that cannot be simply removed, and can affect the classifiers.

Table 5.14 Quantiles of the variable amount

Statistics	Value
Minimum	0
1^{st} quartile	13,390
Median	74,872
3^{rd} quartile	208,721
90^{th}percentile	365,423
98^{th} percentile	1,019,958
Maximum	92,445,516

4. **nameOrig** and **nameDest** are two variables representing the ID of the customer who started the transaction, and the one receiving it. Those two variables were later removed since they represent the simple ID and do not affect the fraud.
5. **oldbalanceOrg** is a continuous variable representing the initial account balance of the customer making the operation before the transaction was made.
6. **newbalanceOrig** is a continuous variable representing the new account balance of the customer making the operation after the transaction was made.
7. **oldbalanceDest** is a continuous variable representing the initial account balance of the customer receiving the transaction before it was made.
8. **newbalanceDest** is a continuous variable representing the new account balance of the customer receiving the transaction after it was made.
   It is worth noting that the statistical summary of these four variables is similar to "amount" in terms of high skewness and outliers.
9. **isFraud** is the categorical target variable representing a fraudulent transactions. In this case study, the fraudulent transaction is one made by an agent, to benefit illegally from the customers' account, by transferring money without their knowledge to other accounts, then cashing them out. Out of the 6,362,620 transactions in the data set, only 8,213 are fraud cases. This is an extreme imbalance of 0.12% as shown in Fig. 5.10. This chart represents the scatter plot of the variable amount, where the fraud

observations are colored in red and the legitimate ones are colored in black. The extremely small ratio of fraudulent transactions was problematic in our case. An undersampling was done later to increase this ratio to 5%.

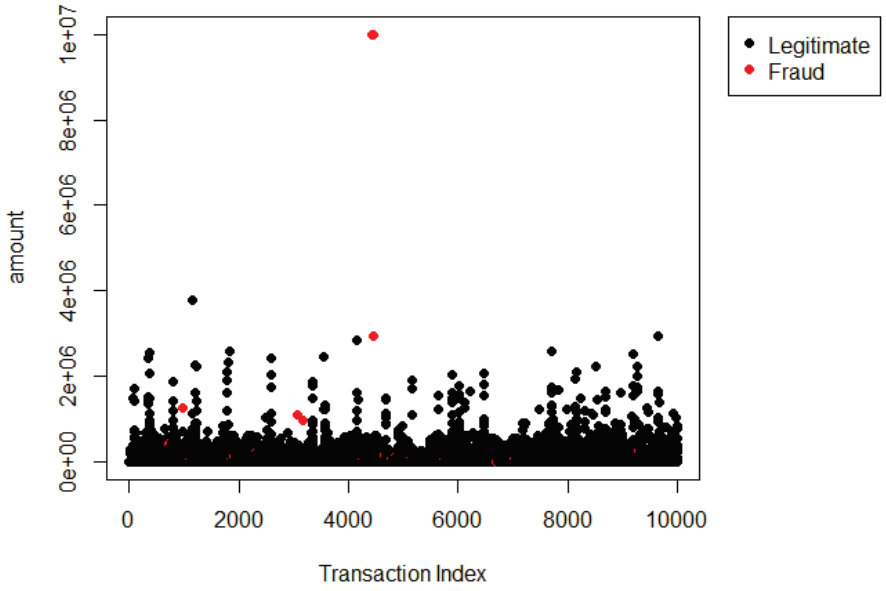


Fig. 5.10 Scatter plot of amount

10. **isFlaggedFraud** is another variable denoting fraud that is detected by the business model. An illegal attempt in this data set according to this variable is an attempt to transfer more than 200,000 in a single transaction. This variable was also removed later.

5.3.2 Implementation and Results

Like the previous application, the first training set differs according to the methods where the OCCoS uses only the fraud observations. The original data set were 6,362,620 transactions. For time consumption and extreme imbalance purposes, an undersampling was done reducing the imbalance ratio to 5%, and reducing the whole size of the data. Table 5.15 shows the description of each data partition and the thresholds used for each method. Fig. 5.11 shows the change of accuracy, sensitivity and F_1 score according to different thresholds, for all the methods used in this case study. These graphs prove the threshold's choices for those methods.

Table 5.15 Data partition and thresholds for the methods - case study 2

Method	Train 1	Test 1 / Train 2	Threshold
OCCoS	6570 fraud observations	16420 observations	0.21
CoSKNN	131400 observations		1.61
LR			0.06
CART			–
NB			0.05

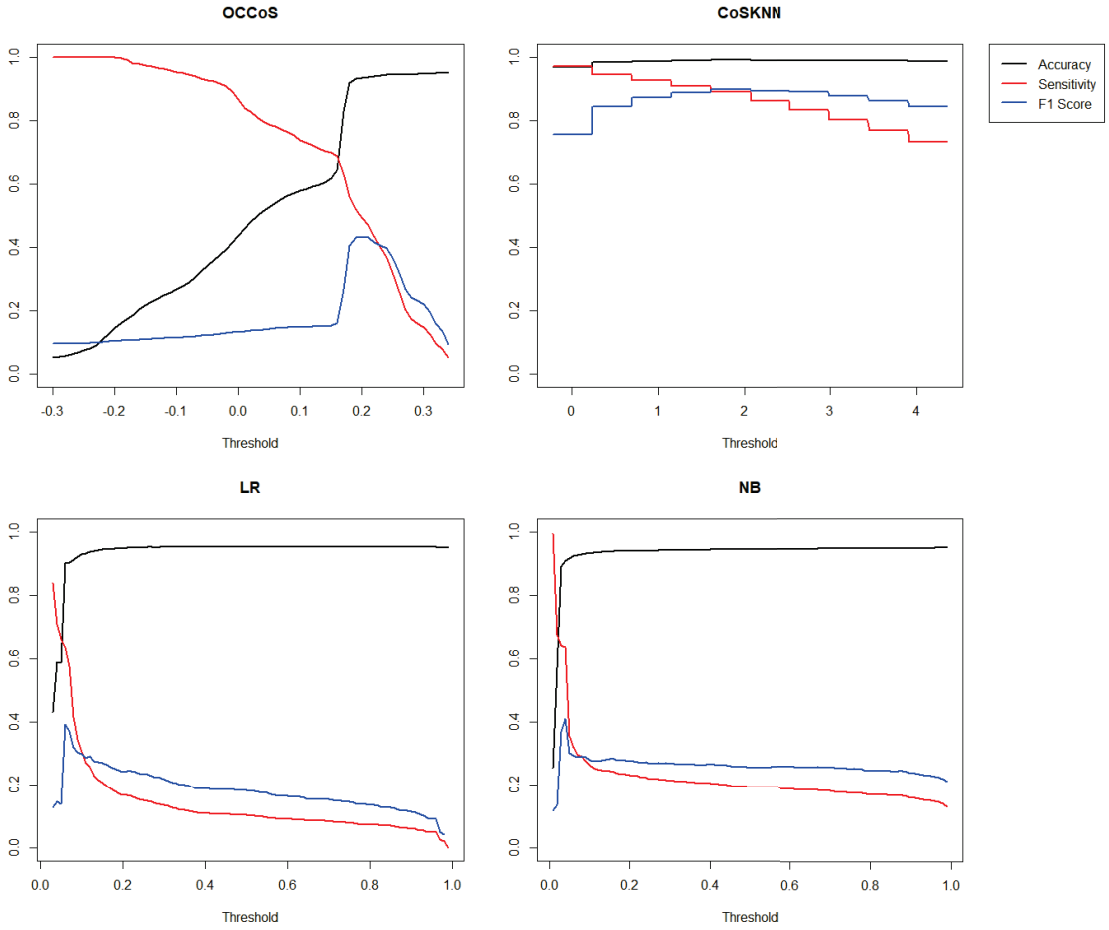


Fig. 5.11 Performance measures variations according to different thresholds - case study 2

For the OCCoS, only the fraud classes were used and the classification was done in the testing step, where the similarity between the new observation and the fraud samples is evaluated. For the CoSKNN, the number of neighbors chosen was 10. Like the previous

case study, for both these methods, the variables are transformed using the mean-standard deviation standardization.

Table 5.16 VIF test results for LR - case study 2

Variables	VIF
amount	5.920
newbalanceDest	148.968
newbalanceOrig	70.805
oldbalanceDest	138.758
oldbalanceOrg	73.475
type	1.172

Table 5.17 LR coefficients - case study 2

Variable	Coefficients
Constant	−4.432
amount	0.0000
oldbalanceDest	0.0000
oldbalanceOrg	0.0000
type	0.324

For the LR, The VIF test was applied to all the six explanatory variables to test the multicollinearity. Table 5.16 shows the results for this test. The four "balance" variables, have VIF values extremely higher than 10. Instead of removing all of them in fear of losing information since the balance variable is an interesting one for fraud detection, we made further analysis that we detail in Section B.1 to decide which variable to keep or remove. In the end, we decide to remove the variable the two destination balance variable. The LR model results are presented in Table 5.17 with the coefficients for each variable, where only the variable called type has a non-zero coefficient. According to Table 5.17, the fraud probability in LR will be calculated following the equation:

$$\frac{e^{X\alpha}}{1+e^{X\alpha}} \quad \text{Where} \quad X\alpha = -4.432 + 0.324 \times \text{type}$$

The CART algorithm is applied to the data set using all variables. Table 5.18 shows the importance of the explanatory variables and Fig. 5.12 shows the plot of the tree obtained. The final tree has 5 leaves with class predictions respectively equal to 0, 0, 1, 0 and 1.

Table 5.18 Variables importance according to CART - case study 2

Variable	Importance
newbalanceDest	100
oldbalanceOrg	86.256
oldbalanceDest	72.794
amount	53.870
type	47.071
newbalanceOrig	2.893

For example in this tree, if the transaction has an amount higher than 756,000, we consider the oldbalanceOrg, if it is higher than 747,000, it falls in node number 7 and is labeled as a fraud.

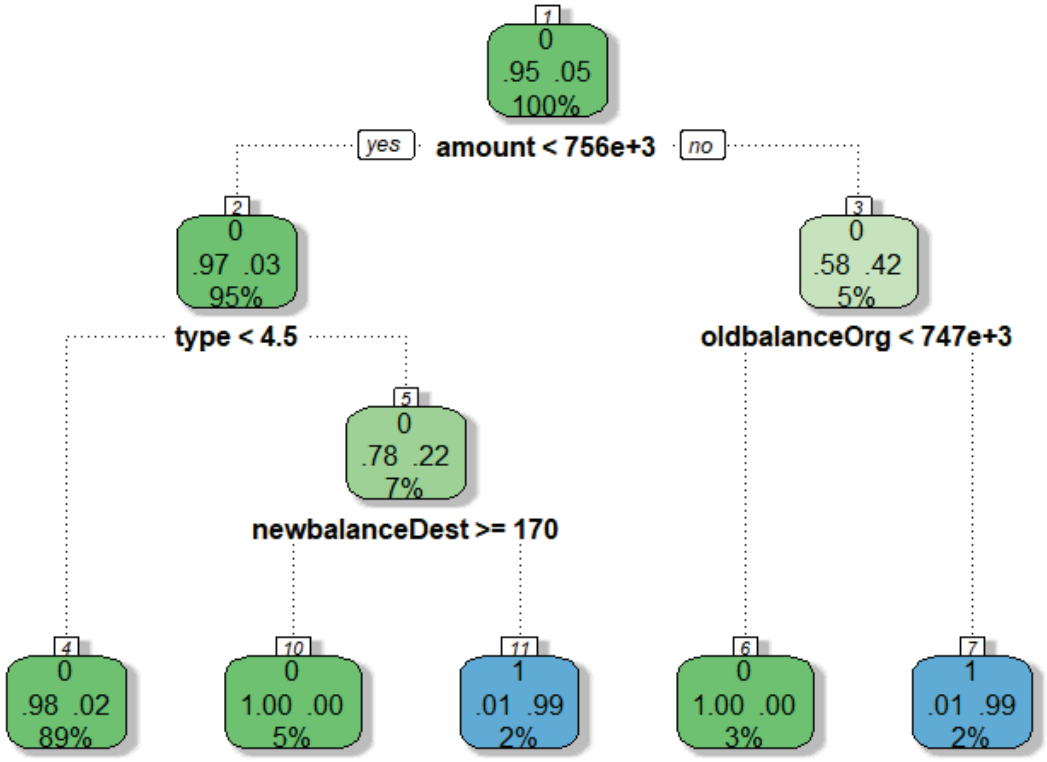


Fig. 5.12 CART tree as single classifier - case study 2

According to the table, the most important variable in discriminating fraud is newbalanceDest, followed by oldbalanceOrg and oldbalanceDest. Finally, the least important is newbalanceOrig, where it is not even included in the tree shown in the figure.

For the NB classifier, conditional fraud probabilities are calculated to find fraud probabilities given the explanatory variables.

The methods mentioned above in this section are built using the first train set (Train 1) and evaluated using the first test set (Test1), to create the second training set to be used for K-MICHA. In Phase II, the k-modes algorithm is then applied to find clusters of the data points and to calculate class probabilities.

Table 5.19 The clusters characteristics - case study 2

Cluster	Cluster Center (modes)	Size	Fraud Probability
1	(1 , 0 , 1 , 0 , 0)	451	0.000
2	(1 , 1 , 1 , 1 , 0)	171	1.000
3	(1 , 0 , 1 , 1 , 0)	2	0.000
4	(1 , 1 , 0 , 0 , 0)	1	1.000
5	(1 , 0 , 0 , 0 , 0)	174	0.017
6	(0 , 0 , 0 , 0 , 0)	14120	0.006
7	(0 , 1 , 1 , 0 , 0)	1	1.000
8	(0 , 0 , 0 , 0 , 1)	6	0.000
9	(1 , 1 , 1 , 1 , 1)	347	1.000
10	(1 , 1 , 1 , 0 , 0)	2	0.000
11	(0 , 0 , 0 , 1 , 0)	236	0.699
12	(1 , 0 , 1 , 0 , 1)	578	0.000
13	(1 , 1 , 1 , 0 , 1)	1	0.000
14	(0 , 0 , 0 , 1 , 1)	11	1.000
15	(0 , 1 , 0 , 1 , 1)	28	1.000
16	(0 , 0 , 1 , 0 , 0)	282	0.000
17	(0 , 1 , 1 , 1 , 0)	3	1.000
18	(0 , 1 , 0 , 0 , 0)	1	0.000
19	(0 , 1 , 0 , 1 , 0)	5	1.000

Table 5.19 shows the characteristics of the clusters obtained for case study 2. The clusters shown here are very distinguishable, they all show an either high or low probability of fraud. The most discriminating clusters that either contains most data points or extremely low or high fraud probability are clusters number 1, 2, 5, 6, 9, 11, 12 and 16. The centers represent the method's predictions that are identical in each cluster respectively NB, CART, LR, CoSKNN and OCCoS. The total number of clusters that we might have is $2^5 = 32$. However, in this case, the final number of clusters obtained was 19. No observations were defining more clusters. According to Table 5.19, the three biggest clusters are clusters number 6, 12 and 1. Cluster 6 contains 14120 observations where all the methods predictions were 0 meaning legitimate transaction. The second big cluster (number 12) is the one having 578 observations where NB, LR, and OCCoS predict 1 and CART and CoSKNN predict 0. Cluster 1 is the one having 451 observations where only NB and LR predict 1. Those three clusters have extremely low fraud probabilities. This is due to the low performance of NB, LR, and OCCoS. The cluster where all methods

predict fraud is cluster 9 and, the fraud probability, in this case, is high, equal 1. Another interesting cluster in Table 5.19 is cluster 11 with high fraud probability showing clearly the importance of CoSKNN, wherein this cluster is the only method predicting fraud. In voting, this would mean a very low fraud probability, whereas, K-MICHA shows a fraud probability of 0.699.

Similarly to the previous case study, after the analysis done for each cluster to find the strengths and weaknesses of each method inside each cluster, we will now consider the original data set's variables. Fig. 5.13 shows the scatter plots of the variables colored by the five most important clusters.

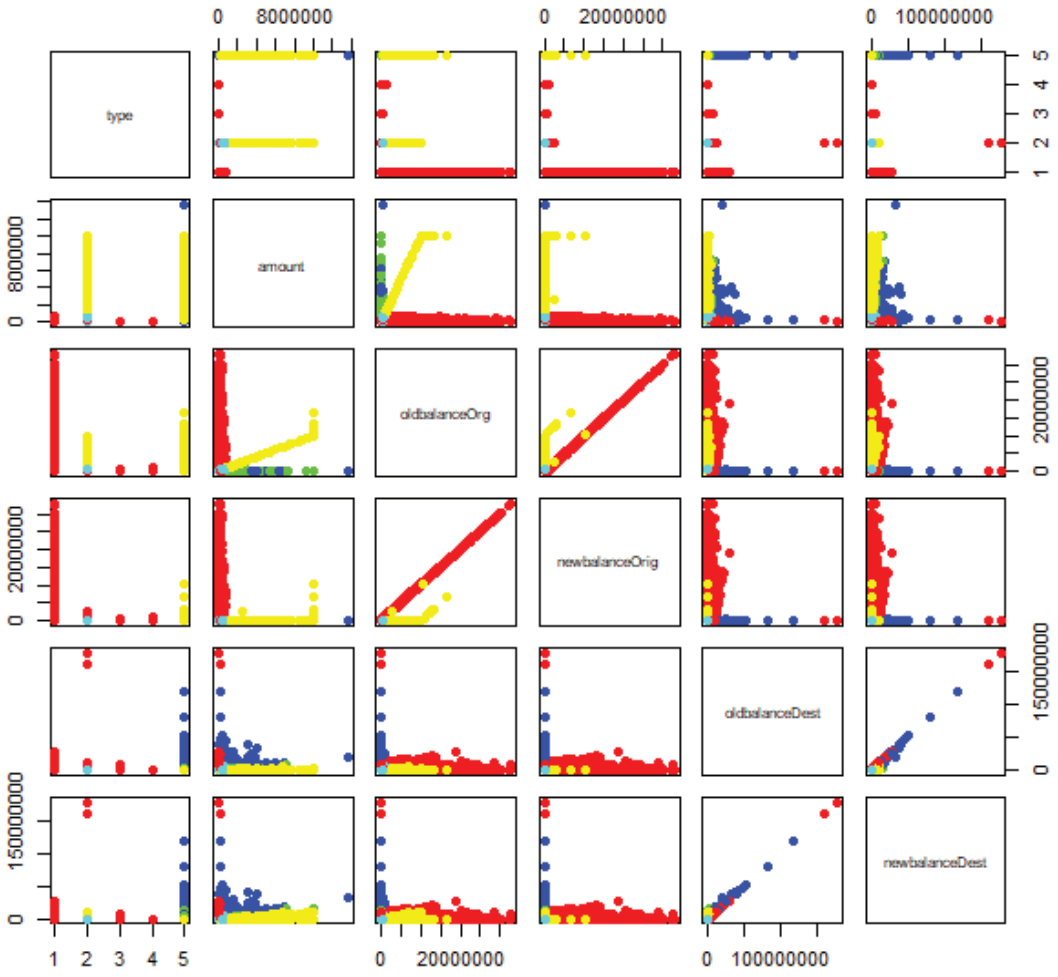


Fig. 5.13 Pairs of the explanatory variables scatter plots colored by the clusters - case study 2

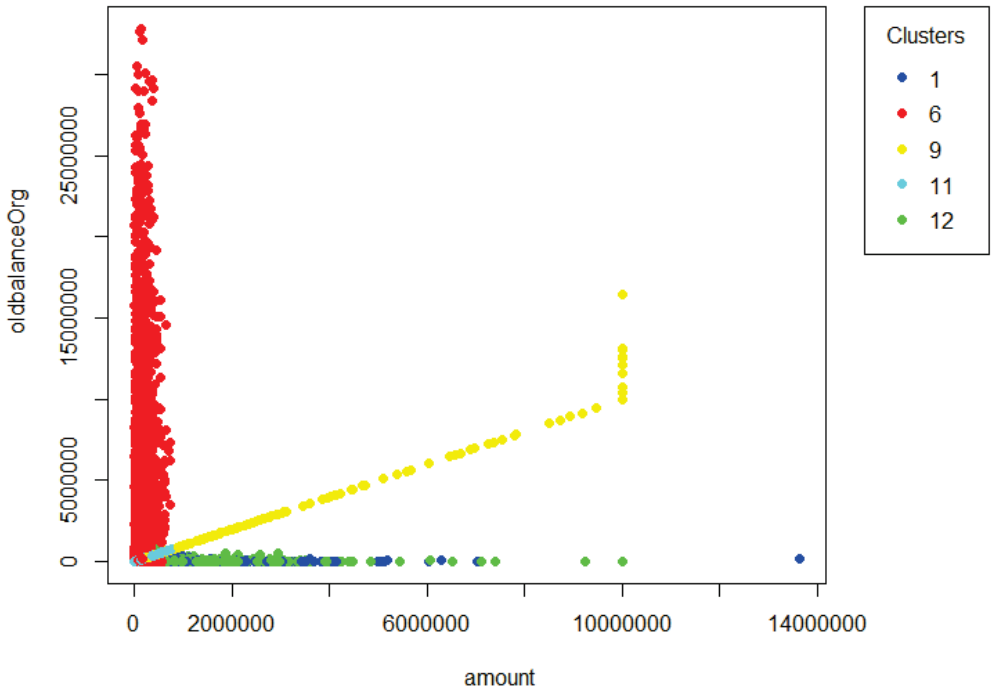


Fig. 5.14 Scatter plot of oldbalanceOrg against amount colored by the clusters

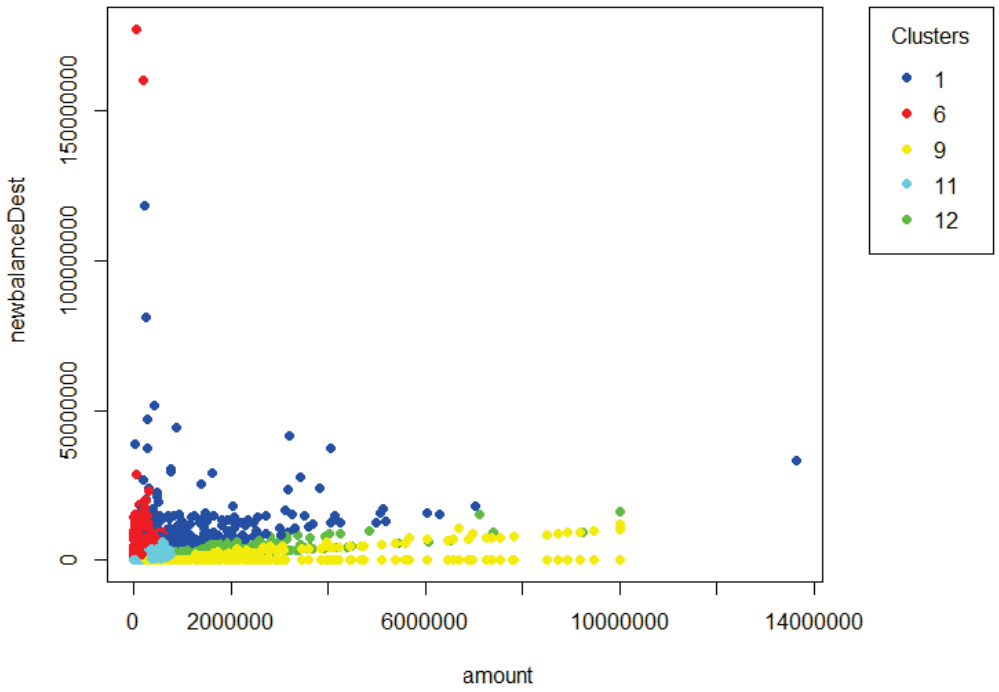


Fig. 5.15 Scatter plot of newbalanceDest against amount colored by the clusters

It is difficult to analyze the clusters in these plots. However, two plots are remarkable, where certain groups of observations can be differentiated by the 5 colors representing different clusters. These two plots are the ones that show the variable oldbalanceOrg against amount and newbalanceDest against amount. These plots are presented in Fig. 5.14 and 5.15, where five clusters are easily distinguished. These clusters are numbers 1, 6, 9, 11 and 12 that correspond to the biggest clusters and interesting ones according to Table 5.19. These plots also show the importance of the variables amount, oldbalanceOrg and newbalanceDest, since these are the variables that highlight the different clusters.

5.3.3 Validation and Comparison

The validation of K-MICHA is done using a second test set of 16440 observations. Voting and weighted voting are used where the weights are the F_1 scores of the methods. Logistic regression stacking was applied where the target variable isFraud, and where the explanatory variables are the outputs of the methods. The logistic coefficients of the methods are presented in Table 5.21 where the highest coefficient corresponds to the CoSKNN method. The VIF test was applied to the outputs of the methods representing the explanatory variables to test the multicollinearity. Table 5.20 shows the results for this test. All the methods had a VIF less than 10, which means that they can all be kept in the LR stacking method. The coefficients are presented in Table 5.21

Table 5.20 VIF Test Results for LR Stacking - case study 2

Variables	VIF
NB	4.20
CART	4.04
LR	4.10
CoSKNN	3.54
OCCoS	2.06

Table 5.21 LR Stacking Coefficients - case study 2

Variable	Coefficients
Constant	-5.11
NB	0.76
CART	7.60
LR	-4.01
CoSKNN	5.99
OCCoS	1.45

A CART stacking algorithm is applied, where the target variable is isFraud and the explanatory variables are the performance of the methods. Table 5.22 shows the importance of the explanatory variables, i.e. the methods, and Fig. 5.16 shows the plot of the tree obtained which is a very small one, with 2 final leaves, using only one split.

Table 5.22 Methods importance according to CART stacking

Method	Importance
CoSKNN	100
CART	68.24

In the tree, the only method used is CoSKNN. This result is then identical to the CoSKNN result. According to both the table and the tree, CoSKNN is the most important method in the case study. The second is the CART algorithm as the single classifier, even though it is not used in the tree. The other method was not included in the tree algorithm, and they had no importance.

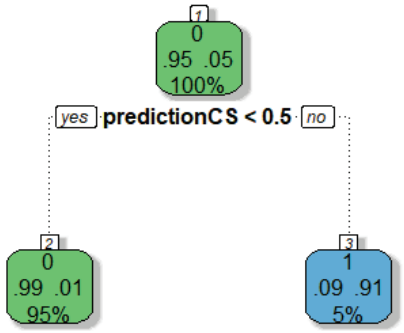


Fig. 5.16 CART stacking tree - case study 2

For the Adaboost algorithm, 10 trees were generated to obtain the final one. The weights of these trees are respectively: 3.26, 2.91, 2.41, 2.14, 1.91, 1.80, 1.53, 1.85, 1.95 and 1.51. For the random forest, 500 trees were generated. The variable importance according to the forest is shown in Table 5.23.

Table 5.23 Variables importance according to RF - case study 2

Method	Importance
oldbalanceOrg	100
amount	76.30
newbalanceDest	64.15
type	48.65
newbalanceOrig	38.82
oldbalanceDest	30.71

In the following, we present the result of the performance measures obtained. Using these results, we will compare K-MICHA with each method alone and with other ensemble learning approaches. Table 5.24 shows the results of the methods according to accuracy, sensitivity and F_1 score.

Table 5.24 The performance of the methods - case stusy 2

Method	**Accuracy**	**Sensitivity**	F_1 **score**
OCCoS	93.7%	0.47	0.43
CoSKNN	99.0%	0.88	0.89
LR	90.2%	0.63	0.39
CART	98.3%	0.67	0.80
NB	90.8%	0.63	0.40
K-MICHA	**98.9%**	**0.91**	**0.90**
Voting	98.2%	0.65	0.78
Weighted Voting	98.2%	0.70	0.82
Logistic Stacking	98.9%	0.88	0.89
CART Stacking	98.9%	0.88	0.89
Adaboost	98.6%	0.72	0.84
RF	98.9%	0.79	0.88

In this case, according to Table 5.24, accuracy measures were not as close as in the case of credit card fraud. They were all ranging from 90% to 99%, with very different sensitivity and F_1 score values. As single classifiers, CoSKNN was the best method, in terms of accuracy, sensitivity and F_1 score. When applying combination methods, we obtained high accuracies, with fewer sensitivity rates or at least as equal as the best method's sensitivity 0.88. Same as the previous case study, K-MICHA achieved a higher sensitivity rate of 0.91 with the same higher F_1 score of 0.90 that was so close to the F_1 scores obtained with other methods with less sensitivity. The Adaboost and RF algorithms, in this case, are performing much better than in case study 1, achieving F_1 scores higher than the CART algorithm as the single classifier. However, their performance is still not as good as combining different algorithms.

5.4 Case Study 3: Auto Insurance Fraud Detection

For the auto insurance fraud detection data set, we choose five machine learning algorithms: CoSKNN, LR, CART, NB, and ANN. We combine them using K-MICHA and compare it with the same stacking approaches used in the previous application for comparison. In the following, we describe the data set, the training, and testing, and finally the results.

5.4.1 Data Description

Data used here is synthetic labeled data set of car accident insurance claims[5]. The original data consists of 1000 observations and 40 variables. In the preprocessing step, we generate simulated observations because 1000 are not enough. We also removed some variables that had no relation to the fraud. The final data set used contains 22295 observations and 27 variables, where the imbalance ratio is 12.2%. The 27 variables we kept are described below through four categories: personal, insurance policy related, car accident related and the fraud variable.

Personal Customer Information

1. **months_as_customer** is a discrete variable indicating the time in months that the insured has been a customer. The variable ranges from 0 to 479.
2. **age** is a continuous variables ranging from 19 to 64 years.
3. **insured_sex** is a categorical variable where 51.05% are female and 48.95% are male.
4. **insured_education_level**, is a qualitative variable of 7 categories as presented here after in Table 5.25

Table 5.25 Insured educational level frequencies

Associate	College	High School	JD	Masters	MD	PhD
8.48%	13.68%	19.22%	18.99%	16.02%	14.72%	8.89%

5. **insured_occupation** is a qualitative variable of 14 categories representing different sectors: clerical, armed forces, craft/repair, executive managerial, farming/fishing, handlers/cleaners, machine operator inspector, other services, private house service, prof-specialty, protective service, sales, tech support, and transport/moving.

[5]Extracted from https://databricks-prod-cloudfront.cloud.databricks.com

6. **insured_relationship** defining the relationship status of the customer, with 6 categories given in Table 5.26.

Table 5.26 Insured relationship status frequencies

husband	not-in-family	other-relative	own-child	unmarried	wife
6.90%	15.82%	22.83%	27.87%	19.06%	7.52%

7. **capital.gains** is a continuous variable that represents the gain realized by the sell of a capital asset (stock, bond or real estate). It ranges from 0 to 100,500.
8. **capital.loss** is a continuous variable, where the sell of a capital asset results in a loss. Fig. 5.17 shows the histogram of capital.gains and capital.loss, these two variables show high skewness where the distribution is mostly concentrated on zero.

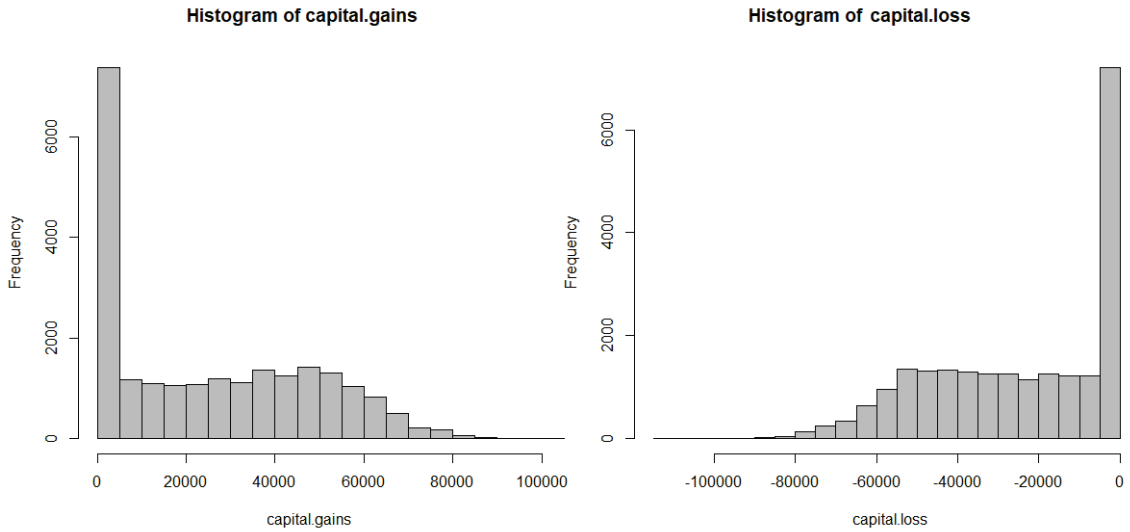


Fig. 5.17 Histograms of capital.gains and capital.loss

Variables Related to the Insurance Policy

9. **policy_deductable** is a discrete variables ranging between 500 and 2000. It indicates the amount that should be covered by the insured in case of an accident before the insurance company pays any expenses.
10. **policy_annual_premium** is the fee paid to the insurance company in exchange for a one-year insurance policy that guarantees payment of benefits in case of a car accident. This variable ranges from 433.33 to 2047.59. Fig. 5.18 shows the distribution of this variable, which is similar to a normal distribution.

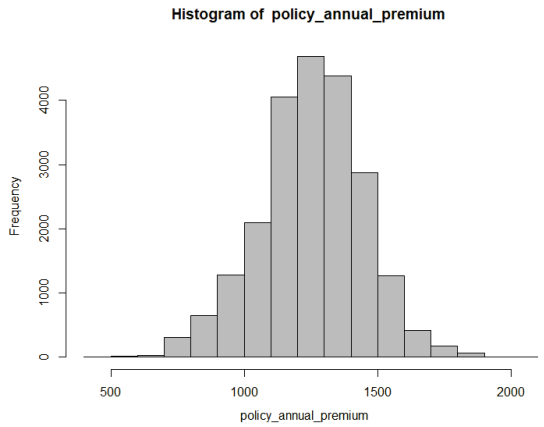


Fig. 5.18 Histograms of policy_annual_premium

11. **auto_make** is a qualitative variable of 14 levels. It represents the brand of the car insured, like Audi, BMW, Chevrolet, Ford, Mercedes *etc.*
12. **auto_year** is a discrete variable representing the year model of the car ranging from 1995 to 2015.

Varibales Related to the Car Accident

13. **incident_type** is a qualitative variable of 4 categories, Multi-vehicle Collision, Parked Car Single Vehicle Collision, and Vehicle Theft.
14. **collision_type** is a qualitative variable of 4 categories: Not Applicable, Front Collision, Rear Collision and Side Collision.
15. **incident_severity** is a qualitative variable of 4 categories: Major Damage, Minor Damage Total Loss and Trivial Damage.
16. **authorities_contacted** is a qualitative variable of 5 categories: Ambulance, Fire, None, Police and Other.
17. **incident_city** is a qualitative variable of 7 categories of cities in the USA: Arlington, Columbus, Hillsdale, Northbend, Northbrook, Riverwood, and Springfield.
18. **number_of_vehicles_involved** is a discrete variable ranging from 1 to 4.
19. **property_damage** is a qualitative variable of 3 categories: Not Applicable, Yes and No.
20. **bodily_injuries** is a discrete variable representing the number of persons injured in the accident if there is.
21. **witnesses** is a discrete variable representing the number of witnesses if there is.
22. **police_report_available** is a qualitative variable of 3 categories: Not Applicable, Yes and No.

23. **total_claim_amount**, **injury_claim**, **property_claim** and **vehicle_claim** are continous variables representing the claims amount. Table 5.27 and Fig. 5.19 show statistical summaries of these variables. According to these statistics, these variables can be assumed to follow normal distribution, where little to no outliers are present.

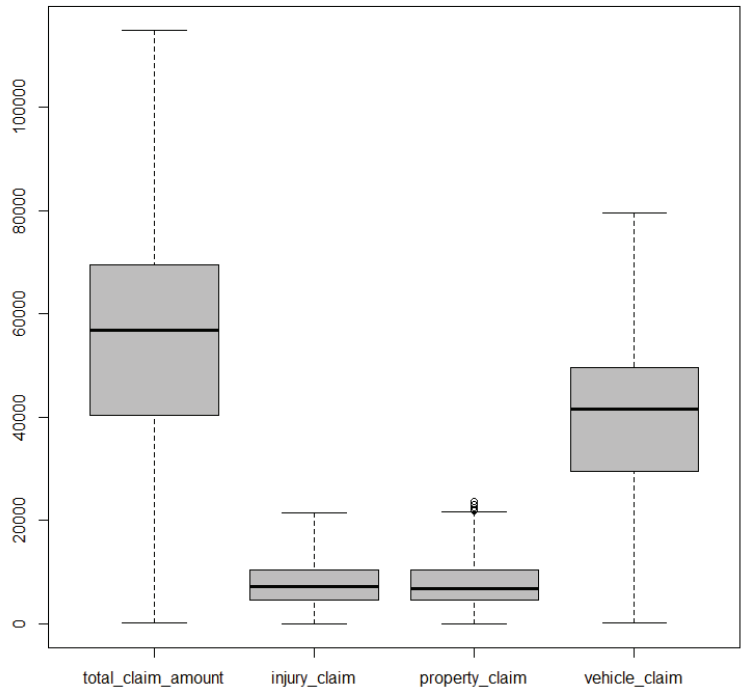


Fig. 5.19 Boxplots of the claims amounts variables

Table 5.27 Statistical characteristics of the claims amounts variables

Variable	Mean	St. Dev.	Min	1^{st} Quartile	3^{rd} Quartile	Max
total_claim_amount	53,862	22,420	100	40,376	69,567	114,920
injury_claim	7,589	4,259	0	4,596	10,494	21,450
property_claim	7,457	4,107	0	4,622	10,339	23,670
vehicle_claim	38,815	15,844	70	29,509	49,538	79,560

The Fraud Variable

fraud_reported is the categorical variable that describes whether fraud is reported or not. This variable will be used as a target variable where 1 denotes a fraudulent claim

and 0 a genuine one. In this data set, we have 2717 fraud claims and 19578 genuine ones. The class imbalance problem is present here with a ratio of 12.2%. Fig. 5.20 represents the scatter plot of the variable policy_annual_premium against total_claim_amount, where the fraud observations are colored in red and the legitimate ones are colored in black.

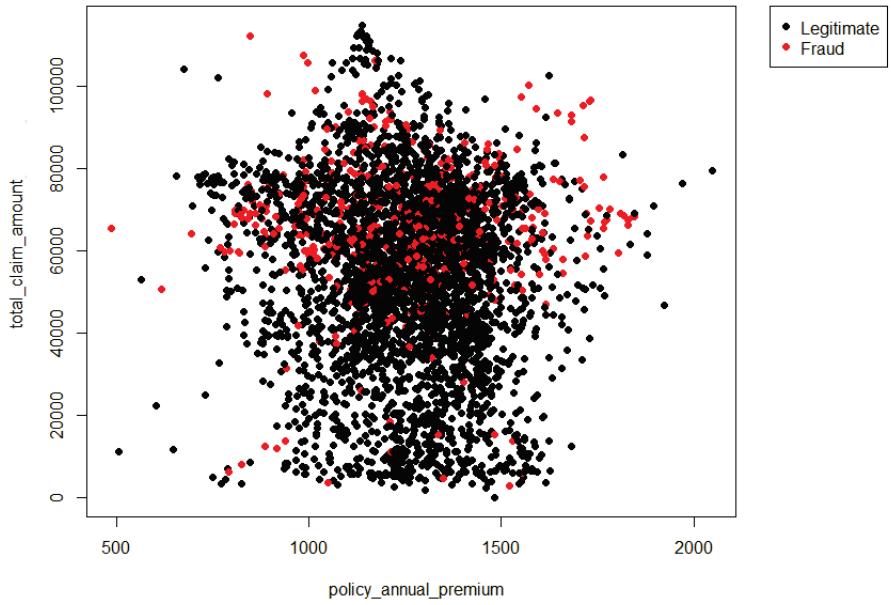


Fig. 5.20 Scatter Plot of annual_premium against total_claim_amount

5.4.2 Implementation and Results

In this case study, all the combined methods are trained using the same training set. Table 5.28 shows the data partition and the thresholds used for each method.

Table 5.28 Data partition and thresholds for the methods - case study 3

Method	Train 1	Test 1 / Train 2	Threshold
CoSKNN	17835 observations	2230 observations	1.60
LR			0.19
CART			–
NB			0.32
ANN			1.53

Fig. 5.21 shows the change of accuracy, sensitivity and F_1 score according to different thresholds, for all the methods used in this case study. These graphs prove the threshold's choices for those methods.

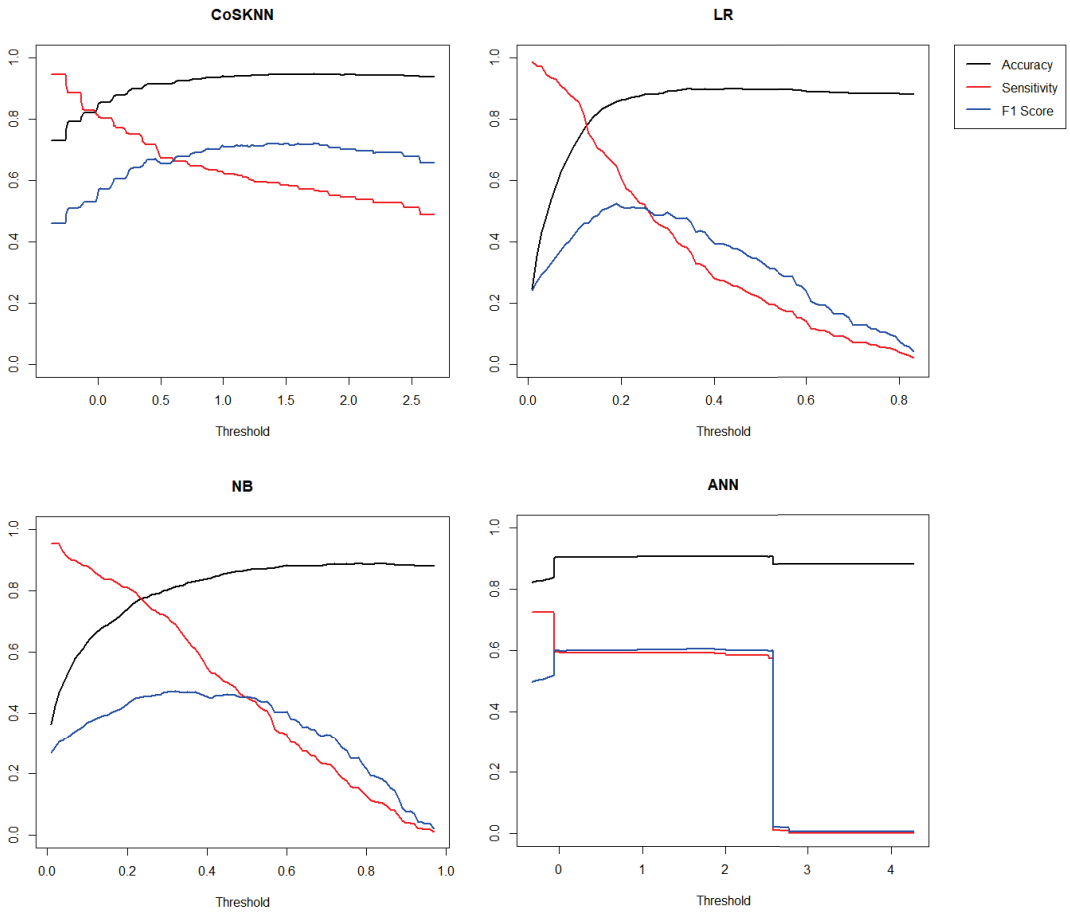


Fig. 5.21 Performance measures variations according to different thresholds - case study 3

For the CoSKNN, the number of neighbors chosen was 25. Like the previous case study, the variables are transformed using the mean/standard deviation standardization.

For LR, the VIF test was applied to all the 26 explanatory variables to test the multicollinearity. Table 5.29 shows the corresponding results. The four variables total_claim_amount, injury_claim, property_claim, and vehicle_claim, have extremely higher VIF values. Instead of removing all of them in fear of losing information since the claim amount is an interesting variable, we made further analysis (details in B.2) to decide which variable to keep or not. In the end, we decide to remove the variable vehicle_claim.

Table 5.29 VIF test results for LR - case study 3

Variables	VIF
age	9.080
authorities_contacted	1.208
auto_make	1.177
auto_year	1.114
bodily_injuries	1.086
capital.gains	1.214
capital.loss	1.178
collision_type	1.655
incident_city	1.207
incident_severity	1.316
incident_type	5.081
injury_claim	56,230,939
insured_education_level	1.223
insured_occupation	1.209
insured_sex	1.158
insured_relationship	1.065
months_as_customer	8.975
number_of_vehicles _involved	5.224
police_report_available	1.086
policy_annual_premium	1.186
policy_deductable	1.285
property_claim	51,961,628
property_damage	1.108
total_claim_amount	1,590,035,981
vehicle_claim	798,625,693
witnesses	1.140

Table 5.30 LR coefficients - case study 3

Variable	Coefficients
Constant	212.467
age	−0.079
auto_year	−0.107
collision_type	−0.167
incident_city	0.106
incident_severity	−1.385
incident_type	0.207
injury_claim	−0.0002
insured_education_level	0.123
insured_occupation	0.150
insured_sex	0.306
insured_relationship	−0.123
months_as_customer	0.004
police_report_available	−0.071
policy_deductable	0.0004
property_claim	−0.0002
total_claim_amount	0.0001
witnesses	0.185

The LR model results are presented in Table 5.30 with the coefficients for each variable. The absence of more variables at the end of the LR model is the reason for a backward optimization process done, to obtain the optimal LR model with less complexity.

The CART algorithm is applied to the data set using all variables. The optimal tree obtained has 50 leaves as explained in Fig. 5.22. This chart represents the error resulting from different trees with different complexity parameter (cp). The optimal tree is the last tree obtained before the first increase in the error.

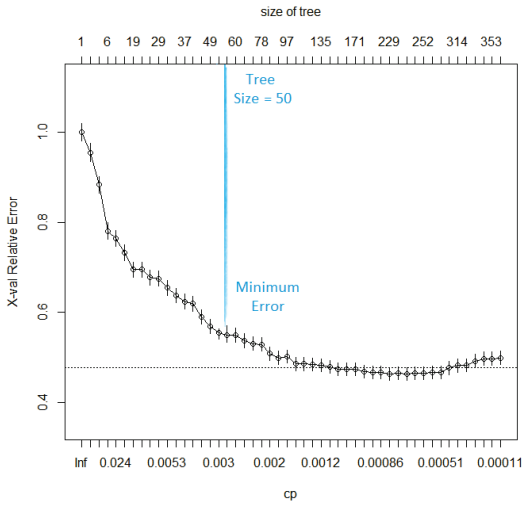


Fig. 5.22 Trees errors

Table 5.31 shows the importance of the explanatory variables. The five most important variables in discriminating fraud are incident_severity, vehicle_claim, total_claim_amount, months_as_customer and insured_relationship.

Table 5.31 Variable importance according to CART - case study 3

incident_severity	100
vehicle_claim	86.688
total_claim_amount	72.873
months_as_customer	62.191
insured_relationship	60.678
capital.gains	58.683
property_claim	53.213
policy_deductable	50.327
insured_occupation	49.996
age	49.108
injury_claim	49.085
auto_make	45.770
policy_annual_premium	45.511
bodily_injuries	44.140
property_damage	40.814
insured_education_level	39.019
capital.loss	37.182
auto_year	35.532
incident_city	25.376
witnesses	25.017
incident_type	24.268
number_of_vehicles_involved	22.612
authorities_contacted	21.216
police_report_available	18.134
insured_sex	15.900
collision_type	15.382

For the ANN, data is first normalized using the minmax normalization, meaning that variables are scaled to a range of [0,1]. A multilayer feedforward network architecture is applied and trained with one input layer of 26 nodes representing the explanatory variables, three hidden layers of 10, 5 and 3 nodes respectively, and an output layer providing the fraud probability as shown in Fig. 5.23. The algorithm used to adjust the weights is the resilient backpropagation algorithm.

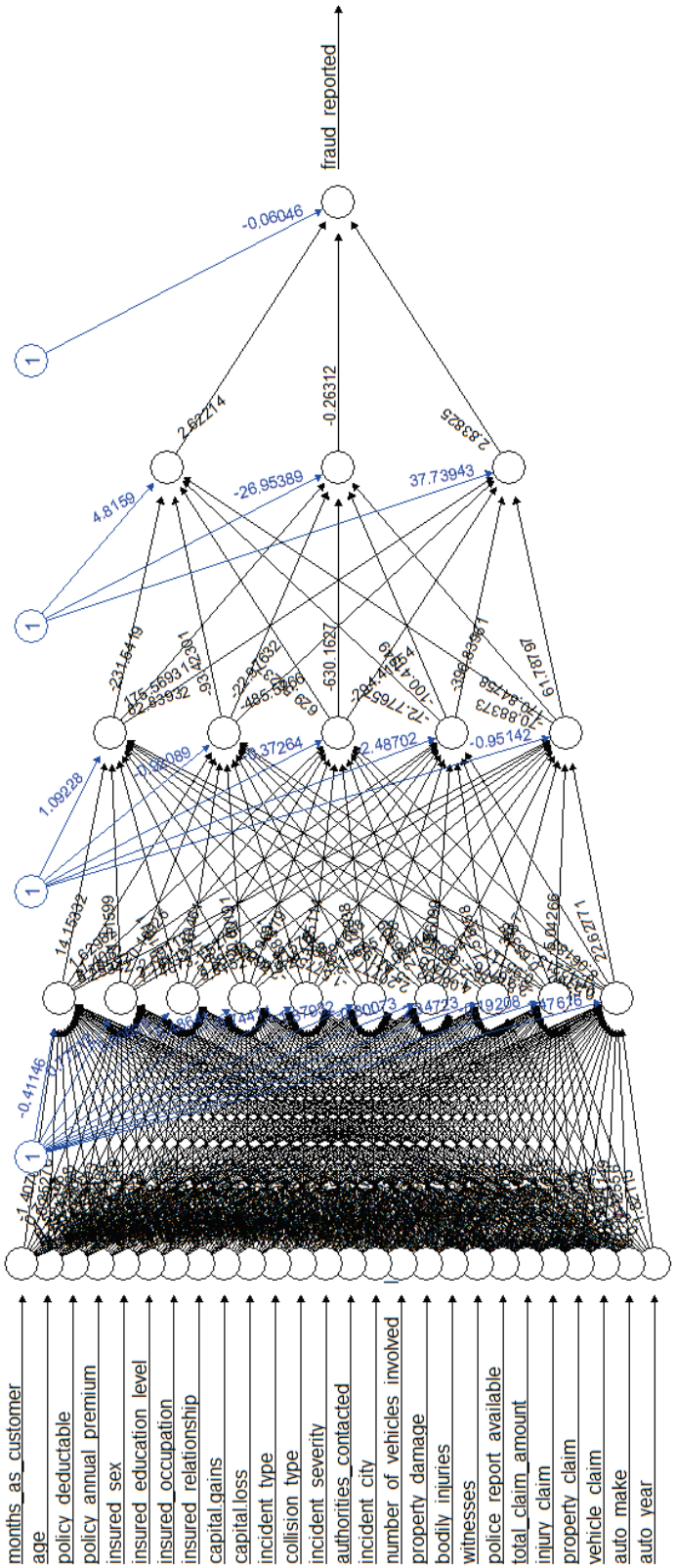


Fig. 5.23 ANN architecture

Similarly to the previous case study, for the NB classifier, conditional fraud probabilities are calculated to find fraud probabilities given the explanatory variables.

These methods are built using the first train set (Train 1) and evaluated using the first test set (Test1), to create the second training set to be used for K-MICHA. In Phase II, the k-modes algorithm is then applied to find clusters of the data points and to calculate class probabilities.

Table 5.32 presents the characteristics of the different clusters. The most discriminating clusters are 4, 14, 18, 24, 25, 27, 28 and 29, these clusters either contain most data points or extremely low or high fraud probability. The centers represent the method's predictions that are identical in each cluster respectively CoSKNN, LR, CART, NB, and ANN. Note that, the total number of clusters that we might have is 32. However, in this case, the final number of clusters obtained is 31. According to Table 5.32, we can analyze the different methods of outputs. The three biggest clusters are the clusters 25, 18 and 28, with the lowest probabilities of fraud respectively: 0.015, 0.134 and 0.093. Cluster 25 consists of observations where all methods predicted legitimate claims. The other two clusters represent observations where just LR and NB predicted fraud. This result gives us an idea of the low performance of these two methods. Cluster 29 is the one where all methods predict fraud with a probability of 0.98. Cluster 4 and 24 present a different combination of methods where the final result is a fraud probability of 1.

In the analysis above, we were trying to understand the strengths and weaknesses of each method inside each cluster. To relate to the data set's original variables, we consider the scatter plots of the variables colored by the five most important clusters. The results were not as clear as the two previous case studies. However, four plots were analyzable, where groups of observations can be slightly differentiated by the 5 colors representing different clusters. These plots are presented in Fig. 5.24, 5.25, 5.26 and 5.27. In these plots, the dominant color is the one corresponding to cluster 25, which is the biggest cluster representing genuine claims. Fig. 5.24 shows the scatter plot of policy_annual_premium against incident_severity, where the distinction is obvious. The category 1 of the severity of the incident which corresponds to "Major Damage" is the category containing the most observations from cluster 29, having a high fraud probability equal to 0.98. Fig. 5.25, 5.26 and 5.27 represent the scatter plots of total_claim_amount against both injury_claim, proprety_claim and policy_annual_premium against vehicle_claim. In the latter, a big part of cluster 25 is distinguishable in this graph. The clusters in these plots are not distinguishable even though they might look separable. The reason might be that a

higher dimensional plot with the combination of other variables is needed to obtain separable clusters.

Table 5.32 The clusters characteristics - case stusy 3

Cluster	Cluster Center (modes)	Size	Fraud Probability
1	(1 , 0 , 1 , 0 , 0)	2	1.000
2	(1 , 1 , 0 , 0 , 1)	2	1.000
3	(0 , 0 , 1 , 0 , 1)	4	0.000
4	(1 , 1 , 1 , 1 , 0)	23	1.000
5	(1 , 1 , 0 , 1 , 1)	8	0.875
6	(0 , 0 , 1 , 1 , 1)	1	0.000
7	(0 , 1 , 0 , 1 , 1)	45	0.333
8	(1 , 0 , 1 , 1 , 1)	4	1.000
9	(1 , 0 , 0 , 0 , 1)	5	1.000
10	(1 , 1 , 0 , 0 , 0)	4	1.000
11	(0 , 1 , 1 , 1 , 0)	13	0.154
12	(0 , 1 , 1 , 0 , 1)	4	0.500
13	(0 , 1 , 1 , 1 , 1)	29	0.483
14	(0 , 0 , 0 , 0 , 1)	41	0.049
15	(0 , 0 , 1 , 0 , 0)	35	0.114
16	(1 , 1 , 1 , 0 , 0)	1	1.000
17	(1 , 1 , 0 , 1 , 0)	4	1.000
18	(0 , 0 , 0 , 1 , 0)	179	0.134
19	(0 , 1 , 0 , 0 , 1)	9	0.778
20	(1 , 1 , 1 , 0 , 1)	4	1.000
21	(0 , 0 , 0 , 1 , 1)	9	0.444
22	(1 , 0 , 0 , 0 , 0)	9	0.556
23	(1 , 0 , 1 , 1 , 0)	1	1.000
24	(1 , 0 , 1 , 0 , 1)	17	1.000
25	(0 , 0 , 0 , 0 , 0)	1501	0.015
26	(1 , 0 , 0 , 1 , 1)	4	1.000
27	(0 , 1 , 0 , 0 , 0)	63	0.048
28	(0 , 1 , 0 , 1 , 0)	108	0.093
29	(1 , 1 , 1 , 1 , 1)	75	0.987
30	(0 , 1 , 1 , 0 , 0)	6	0.667
31	(0 , 0 , 1 , 1 , 0)	20	0.100

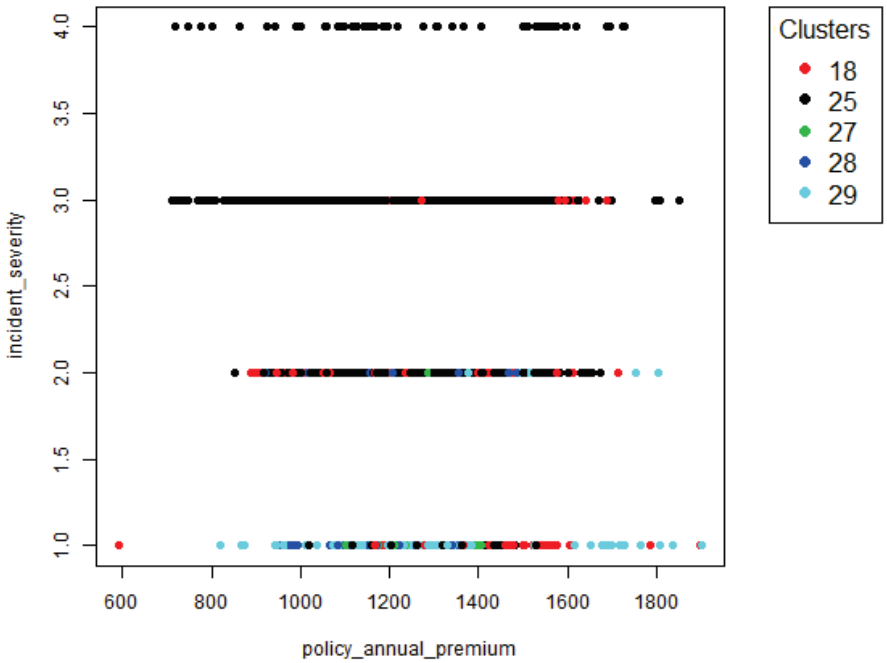


Fig. 5.24 Scatter plot of policy_annual_premium against incident_severity colored by the clusters

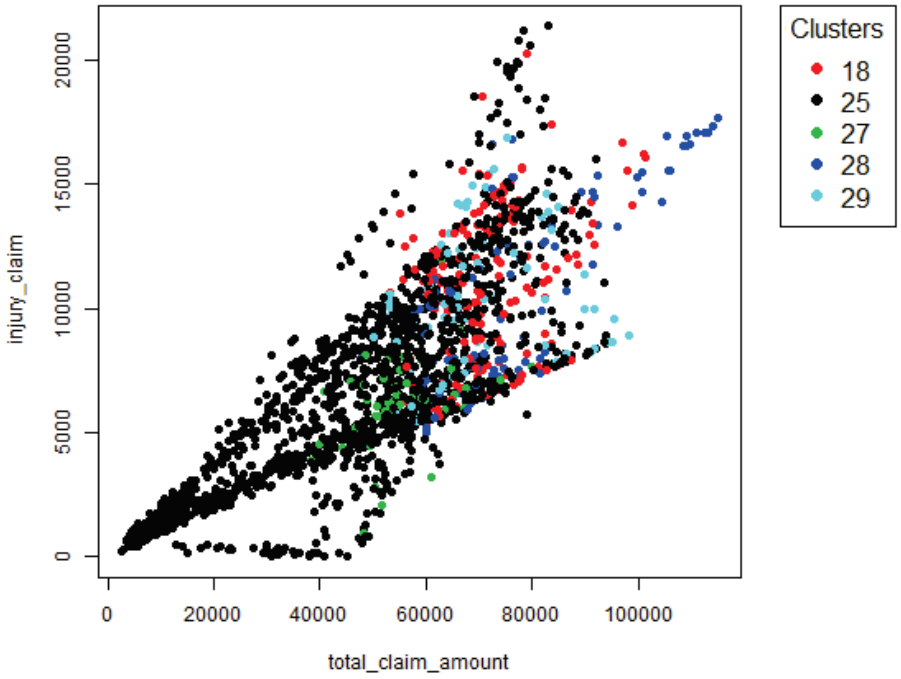


Fig. 5.25 Scatter plot of total_claim_amount against injury_claim colored by the clusters

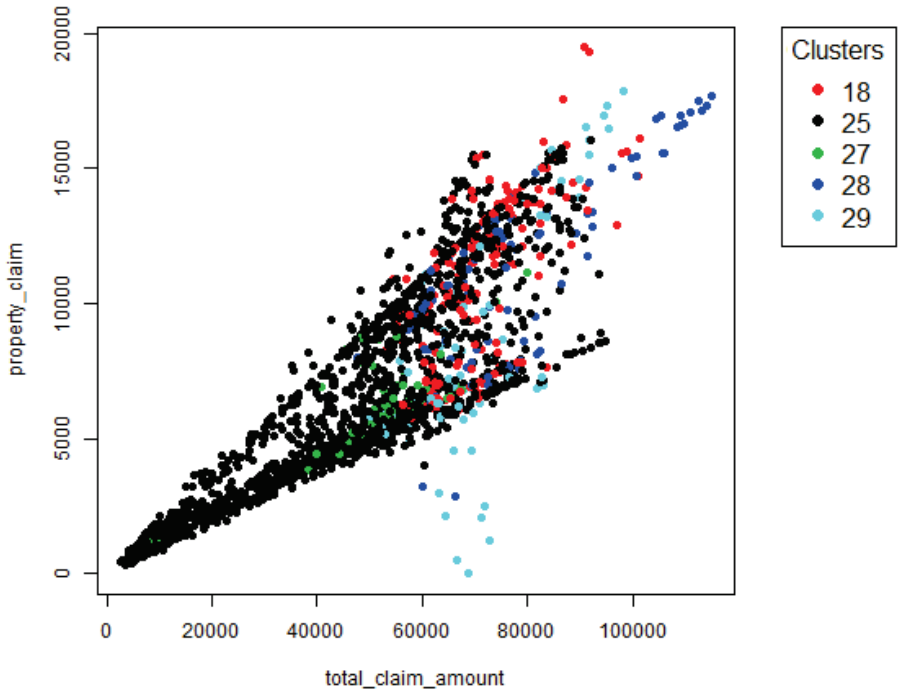


Fig. 5.26 Scatter plot of total_claim_amount against property_claim colored by the clusters

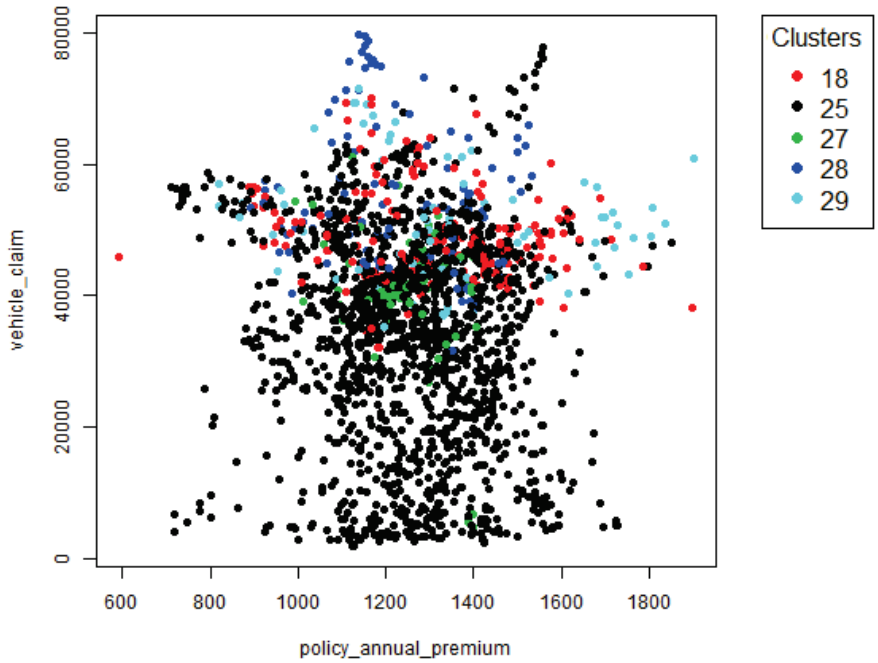


Fig. 5.27 Scatter plot of policy_annual_premium against vehicle_claim colored by the clusters

The reason behind the need for higher dimensional plots is the existence of several important variables that are contributing to the clustering. In other words, there are no two or three dominant important variables.

5.4.3 Validation and Comparison

The validation of K-MICHA is done using a second test set of 2230 observations. Voting and weighted voting are used where the weights are the F_1 scores of the methods.

Logistic regression stacking was applied where the target variable fraud_reported, and where the explanatory variables are the outputs of the methods. The logistic coefficients of the methods are presented in Table 5.34 where the highest coefficient corresponds to the CoSKNN method just like case study 2. The VIF test was applied to the outputs of the methods representing the explanatory variables to test the multicollinearity. Table 5.33 shows the results for this test. All the methods had a VIF less than 10, which means that they can all be kept in the LR stacking method. The coefficients are presented in Table 5.34

Table 5.33 VIF Test Results for LR Stacking - case study 3

Variables	VIF
CoSKNN	1.82
LR	1.77
CART	1.78
NB	1.57
ANN	1.63

Table 5.34 LR Stacking Coefficients - case study 3

Variable	Coefficients
Constant	-3.79
CoSKNN	4.79
LR	0.61
CART	0.92
NB	1.25
ANN	1.42

A CART stacking algorithm is applied, where the target variable is fraud_reported and the explanatory variables are the methods performance. Table 5.35 shows the importance of the explanatory variables, i.e. the methods.

Table 5.35 Methods importance according to CART stacking - case study 3

Method	Importance
CoSKNN	100
CART	9.20
ANN	7.91
LR	2.90
NB	2.49

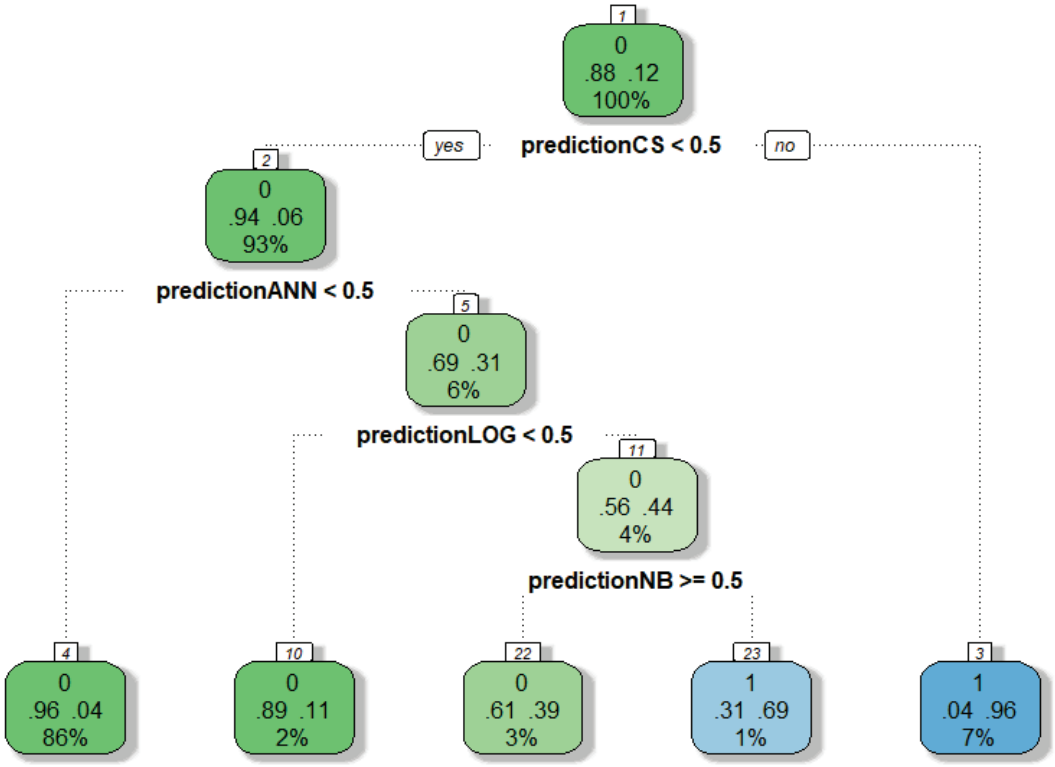


Fig. 5.28 CART stacking tree - case study 3

Fig. 5.28 shows the plot of the final obtained tree. This tree has 5 leaves with class predictions of 0, 0, 0, 1 and 1 respectively. The tree is interpreted similarly to the previous cases. According to both the table and the figure above, CoSKNN is the most important method in the case study. It has a great contribution to the partition of the data, the others are much less important according to the table.

For the Adaboost algorithm, 10 trees were generated to obtain the final one. The weights of these trees are respectively: 1.61, 1.52, 1.46, 1.45, 1.41, 1.46, 1.42, 1.42, 1.45 and 1.41. For the random forest, 500 trees were generated. The variable importance according to the forest is shown in Table 5.36.

Table 5.36 Variable importance according to RF - case study 3

Method	Importance
incident_severity	100.00
policy_annual_premium	57.56
vehicle_claim	55.60
property_claim	51.65
total_claim_amount	49.92
insured_occupation	49.76
auto_year	45.43
months_as_customer	44.37
injury_claim	41.80
policy_deductable	40.69
capital.gains	39.30
age	35.84
incident_city	33.51
auto_make	32.87
capital.loss	32.21
insured_relationship	27.63
authorities_contacted	24.11
insured_education_level	23.70
bodily_injuries	23.36
witnesses	21.57
property_damage	19.89
police_report_available	17.04
collision_type	16.29
number_of_vehicles_involved	14.65
insured_sex	14.02
incident_type	12.91

In the following, we present the result of the performance measures obtained. Using these results, we will compare K-MICHA with each method alone and with other ensemble learning approaches. Table 5.37 shows the results of the methods according to accuracy, sensitivity and F_1 score.

Table 5.37 The performance of the methods - case study 3

Method	**Accuracy**	**Sensitivity**	F_1 **score**
CoSKNN	93.0%	0.50	0.63
LR	82.5%	0.66	0.48
CART	89.0%	0.56	0.55
NB	79.8%	0.65	0.44
ANN	90.3%	0.55	0.58
K-MICHA	**90.8%**	**0.69**	**0.65**
Voting	91.7%	0.43	0.56
Weighted Voting	91.9%	0.62	0.65
Logistic Stacking	92.9%	0.56	0.65
CART stacking	92.2%	0.53	0.62
Adaboost	91.0%	0.68	0.65
RF	92.9%	0.56	0.65

In this case, according to Table 5.37, for the single classifiers, we obtain different results in terms of accuracy and sensitivity measures. None of the single methods alone resulted in the highest values for all three performance measures. The highest F_1 score was obtained using CoSKNN but with the lowest sensitivity rate. Other methods achieved higher sensitivities but with much lower accuracy rates. Same as the previous cases, K-MICHA achieved a higher sensitivity rate of 0.69 with the same higher F_1 score of 0.65 that was obtained with other methods with less sensitivity. The voting combination approach was not improving the results. However, all other combination approaches achieved F_1 scores close to the one we obtained using K-MICHA. The logistic regression, CART stacking and RF combinations improved the accuracy rather than the sensitivity and were close to their results to CoSKNN as the single classifier. The weighted voting was slightly better, whereas the Adaboost algorithm achieved great results that were so close to K-MICHA.

5.5 Big Data Fraud Detection Using H2O Platform in R

We implemented our method K-MICHA using H2O and R to run Big Data sets and analyze them. We used the same credit card fraud data set as in Case Study 1, and we combine LR, ANN, NB, and CART. In the following, we present an overview of H2O and the results of our big data case study.

5.5.1 Introduction to H2O

H2O is the leading open-source in-memory, prediction engine for big data science. It is used by financial institutions, insurance companies, and healthcare companies to implement Artificial Intelligence and deep learning algorithms to solve complex problems[6]. According to H2O documentation, more than 18,000 organizations and 80,000 data scientists use H2O for critical predictions and operations. It is used by 169 Fortune 500 enterprises[7], including 8 of the world's 10 largest banks, 7 of the 10 largest insurance companies, and 4 of the top 10 healthcare companies.

H2O is a Java Virtual Machine (JVM) that is improved by doing in-memory processing of distributed, parallel machine learning algorithms on clusters[8]. The cluster used can be set off on the user's laptop, on a server, or across a multi-node cluster.

Using in-memory compression, H2O deals with billions of data rows in-memory, even with a small cluster. H2O's platform includes interfaces for Javascript, R, Python, Excel/Tableau and Flow, as shown in the top layer of the diagram in Fig. 5.29. It is designed to run in standalone mode, on Hadoop, or within a Spark cluster as shown in the bottom layer of Fig. 5.29. This figure also shows the components of each node in the H2O cluster (i.e. each JVM process). Each node is split into three layers: language, algorithms, and core infrastructure. The language layer consists of an expression evaluation engine for R and the Shalala Scala layer. The algorithms layer consists of algorithms automatically provided by H2O like parse algorithms used to load data sets, machine learning and prediction algorithms and scoring tools for model evaluation. The core layer handles resource, Memory and CPU management.

[6] www.h2o.ai

[7] Also known as Global 500 which is an annual ranking of the top 500 corporations worldwide

[8] We should distinguish between a computer cluster which is a group of coupled computers that work together and cluster analysis which a technique for statistical data analysis like k-means and k-modes as the one we used in K-MICHA

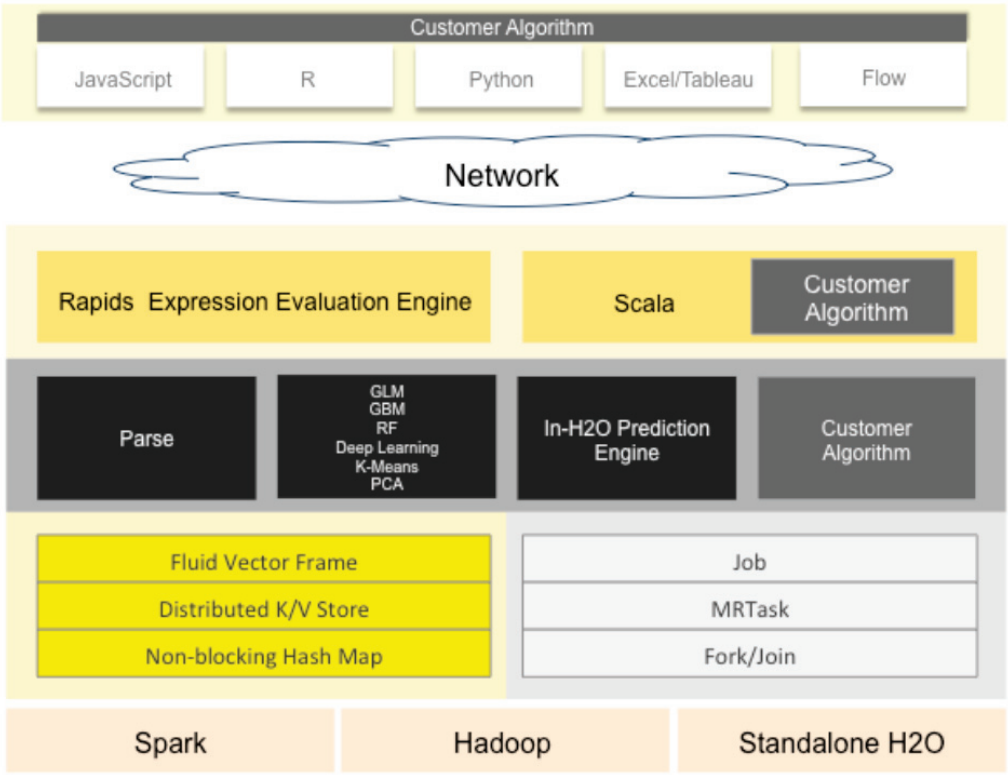


Fig. 5.29 The H2O platform diagram
Source: H2O architecture documentation[9]

H2O packages for all interfaces provide the most common machine learning algorithms, like generalized linear models (linear regression, logistic regression, *etc.*), Naïve Bayes, principal components analysis, k-means clustering, random forest, gradient boosting and deep learning. With H2O, users can build thousands of models and compare the results to get the best predictions.

5.5.2 Application to Credit Card Fraud Data Set

We used the credit card fraud original data set of 10 Million observations, which only a small part of it was used in Case Study 1[10]. Here, we use the H2O R package to run K-MICHA using four machine learning algorithms, LR, ANN, NB, and CART. We compare the results of the single classifiers of H2O and K-MICHA's results.

The H2O cluster's specifications are provided below:

[9] http://docs.h2o.ai/h2o/latest-stable/h2o-docs/architecture.html
[10] Available at http://packages.revolutionanalytics.com/datasets/

- *version:* 3.26.0.2
- *version age:* 2 months and 24 days
- *total nodes:* 1
- *total memory:* 13.42 GB
- *total cores:* 4
- *allowed cores:* 4
- *R Version:* 3.5.0 (2018-04-23)

Whereas in the previous experiments, we used a simple physical node with the following experiments:

- *Operating System*: Windows 10
- *System Type*: 64 bit
- *Processor*: Intel(R) Core (TM) i7 - 7500U
- *Processing Speed*: 2.70GHz 2.90 GHz
- *Memory*: 16 GB
- *R version* 3.5.0 (2018-04-23)

Table 5.38 The clusters characteristics - H2O

Cluster	Cluster Center (modes)	Size	Fraud Probability
1	(0 , 0 , 0 , 0)	899034	0.019
2	(1 , 1 , 1 , 1)	37109	0.727
3	(1 , 0 , 0 ,1)	1781	0.322
4	(1 , 1 , 0 , 0)	2619	0.415
5	(0 , 1 , 1 , 0)	1708	0.296
6	(1 , 0 , 1 , 0)	1456	0.383
7	(0 , 0 , 1 , 1)	1950	0.186
8	(0 , 1 , 0 , 1)	256	0.367
9	(1 , 1 , 0 , 1)	6598	0.446
10	(0 , 0 , 1 , 0)	28914	0.100
11	(1 , 0 , 0 , 0)	1629	0.333
12	(0 , 0 , 0 , 1)	5289	0.234
13	(1 , 1 , 1 , 0)	8456	0.484
14	(1 , 0 , 1 , 1)	649	0.459
15	(0 , 1 , 1 , 1)	1608	0.305
16	(0 , 1 , 0 , 0)	944	0.286

The data set was decomposed into a train set of 8 million observations and two test sets each of one million observations. Table 5.38 presents the characteristics of the different clusters. The centers represent the method's predictions that are identical in each cluster respectively LR, ANN, NB, and CART. The total number of clusters that we have is 16. The most two remarkable clusters are clusters number 1 and 2 where respectively all methods predict legitimate observations and all methods predict fraud, with the lowest and highest and fraud probabilities of 0.019 and 0.727.

The difference between these results and the ones in the previous case studies is stability and robustness. Due to a large number of observations in the clusters, the fraud probabilities assigned are more reliable. Unlike the other cases, we don't have clusters of 1 or 2 observations. In this case, the smallest cluster is of 256 observations. Besides, the highest possible number of clusters was reached, meaning there is no future observation that might create new clusters.

Table 5.39 shows the results of the single classifiers and K-MICHA. The accuracy values, the sensitivity and the F_1 score are presented in this table, as well as the time needed to process these algorithms in seconds are provided.

Table 5.39 The performance of the methods - H2O

Method	**Accuracy**	**Sensitivity**	**F_1 score**	**Processing Time**
LR	95.4%	0.62	0.61	12.70 sec.
ANN	95.4%	0.62	0.61	96.11 sec.
NB	93.0%	0.60	0.51	9.22 sec.
CART	95.1%	0.55	0.57	7.85 sec.
K-MICHA	**95.3%**	**0.63**	**0.62**	**8.37 sec.**

It is proven in this case also that K-MICHA is improving the sensitivity and the F_1 score. The performance measures are similar to Case Study 1. However, in this case, the single classifiers alone are performing better than the single classifiers in the traditional framework. The table also shows the time needed to run these algorithms where the most time consuming is the ANN algorithm that takes 96.11 seconds for training. However, in comparison with R alone, this data set was not analyzable.

5.6 Discussion

In this chapter, we present three case studies to validate K-MICHA in terms of performance and prediction ability. We also compare with the implementation of K-MICHA in a Big Data framework. Each of the three case studies used for validation emphasis a specific aspect. We were interested in proving the efficiency of our approach regardless of methods outputs. In the first case study, the methods performance was similar, whereas, in the second and third cases, the results were very different in terms of all performance measures considered. In the second case study, one method was achieving better results than all other methods according to all performance measures. However, in the third case study, none of the methods alone was achieving high results in all performance measures. This highlights the need for a certain combination technique to use the best of each method and obtain better results. In all cases, K-MICHA achieves the highest sensitivity with the highest F_1 score. According the results showed in the Tables 5.12, 5.24 and 5.37, voting and weighted voting decrease the sensitivity. Our approach can ignore the low performing methods that voting and weighted voting were biased to, or find the relation between them and other methods. Logistic and CART stacking are performing as the best method's performance in all three cases, whereas, K-MICHA can exceed these sensitivities with the same F_1 score. On the other hand, we compare with Adaboost and RF. Their results are very dependant on the base classifier used, which was CART. Note that, if that classifier was already performing good, the improvements were significant. However, if it was not of the good performing algorithms, the results were not satisfactory.

Table 5.40 gives a comparison of the different strengths and weaknesses of all the methods we compared K-MICHA with. This table summarizes the performance of all these methods in terms of different data mining features. The green circles in this table represent a positive quality, the red ones represent a negative one, and the yellow ones correspond to a fair one. This table allows to better illustrate the analysis and comparison between these methods, thus highlighting the importance and added value of K-MICHA.

Besides, we compared the results of the three case studies with the implementation of K-MICHA using the H2O Big Data platform and R. This implementation allowed us to analyze larger data than the ones used in the three case studies, which lead to more stable and reliable results. In terms of processing speed, the total time taken to create K-MICHA with the four machine learning algorithms was 2.23 minutes.

Table 5.40 Comparative table for ensemble learning methods

Method	Handling large number of methods	Ability to deal with weak method	Fraud Detection	Dependency on strong method	Time consumption	Easy interpretation	Assumptions	Overfitting
Voting	🟢	🟢	🔴	🟢	🟢	🟢	🟢	🟢
Weighted Voting	🟢	🔴	🔴	🟢	🟢	🟡	🟢	🟢
Logistic Stacking	🟢	🟡	🟢	🟡	🟢	🟡	🔴	🟢
CART Stacking	🟡	🟡	🟡	🟡	🟡	🟡	🟢	🟡
Adaboost	–	–	depends on base	–	🔴	🔴	🟢	🔴
RF	–	–	depends on base	–	🔴	🔴	🟢	🔴
K-MICHA	🔴	🟢	🟢	🟢	🟡	🟡	🟢	🟢

Chapter 6

Conclusion and Future Work

6.1 Conclusion

Fraud is a long-standing critical problem that has severe impacts on a large spectrum of industrial use cases. There are different sorts of frauds identified in different domains including banking and finance. Over the years, fraud has been the nucleus of scientific research in the financial sector. The banking industry has invested resources including a large sum of money, experts (e.g., data scientists, financial engineers) to develop a solution that is more capable than the traditional state of the art technologies. However, many investigative studies conducted by eminent research organizations such as Forrester, Gartner, IDC, etc. revealed that the trend of fraud incidence is still upward. This unearths that there is a scope to improve the methodologies for detecting frauds.

Furthermore, the advent of extremely powerful Big Data technologies has given the rise to a plentitude of opportunities for the industries to alter the means of fraud detection. The Big Data technologies empowered the industries to develop innovative solutions that leverage the power of data in detecting fraud with high accuracy and high speed which could never be done by the traditional technologies. Although data-driven fraud detection opens ample opportunities, some complex challenges still need exhaustive research.

Chapter 1 provided a comprehensive introduction of frauds, different types of frauds. the financial impact and challenges involved in fraud. The class imbalance problem is a well-known problem that this research aimed to tackle – was explained intelligibly in Chapter 1. Furthermore, the goal and objectives were described and contributions briefly explained in this Chapter.

The open problems of fraud detection turned fraud analytics into an appealing research area. Especially, financial fraud analytics has drawn a huge interest to the researchers. The continuous increase in financial fraud worldwide has encouraged data scientists and researchers to find systems able to reach a high fraud detection rate. Most fraud detection systems are machine learning based, where historical datasets are used to train the models. In these data sets, the proportion of fraud observations is extremely small, which leads to classifiers biased towards legitimate and non-fraudulent observations. To tackle this issue, many researchers introduced cost sensitive approaches and special algorithms, where sometimes ensemble learning is used.

Chapter 2 presented a review of all existing solutions for detecting different types of frauds. The review was focused on the strength and weaknesses of the state of the art fraud detection technologies. The literature review helped to identify the candidate technologies. In addition to the literature review, an experimental study was conducted with the most suitable candidate technologies. The results were presented and discussed in this Chapter 2.

In this research, we introduced two approaches: a Cost-Sensitive Cosine Similarity K-Nearest Neighbor (CoSKNN) as a single classifier, and a K-modes Imbalance Classification Hybrid Approach (K-MICHA) as an ensemble learning methodology that we also implemented in a Big Data framework using H2O and R.

Chapter 3 described the first contribution that ism CoSKNN for detecting frauds. CoSKNN aims to tackle the imbalance problem by using cosine similarity instead of the Euclidean distance and by introducing a score for the classification.

In Chapter 4, I discussed the second contribution 'K-MICHA' of this thesis. In K-MICHA, we introduced a hybrid diversification approach using the k-modes clustering algorithm. We also applied it to a credit card, mobile payment, and auto insurance fraud data sets. K-MICHA aims to group similar data points in terms of the outputs of the classifiers. Then, calculating the fraud probabilities in the obtained clusters to use them for detecting frauds of new transactions. In the three case studies, we combine different algorithms.

Chapter 5 provided the detail of the evaluation that was performed with CoSKNN and K-MICHA. This chapter presented the performance of simple voting KNN using both Euclidean distance and cosine similarity, a distance weighted KNN also using both euclidean distance and cosine similarity, a cost-sensitive KNN approach, a decision tree approach, a one-class classification Support Vector Machine (SVM) and CoSKNN. The comparison was done by applying these methods to a credit card fraud data set

with imbalance, using multiple performance measures, mostly relying on AUPRC and F_1 score. This experiment proved that CoSKNN is outperforming all the other methods.

Besides Chapter 5 presented the comparison of the performance K-MICHA with the performance of the single classifiers. We also compare with other ensemble learning methods like stacking using voting, weighted voting, logistic regression, and CART. We also compared it with Adaboost and random forest. Based on the comparison done in the three cases, we proved the efficiency of our approach, K-MICHA which allowed us to:

1. Reach the highest fraud detection rate possible with the highest F_1 score.
2. Comprise the high fraud detection rate with an acceptable decrease in the overall accuracy of the model.
3. Ignore the low performing methods and find the relation between them and other methods.
4. Distinguish between different categories of observations in the data set, to combine the best of each method.

Besides, K-MICHA had experimented with a relatively larger data set using Big Data platform H2O and R, which lead to improvements in the classification and fraud detection. The H2O case study allowed us to:

1. Prove the efficiency of K-MICHA as an ensemble learning technique. Same as before, we reach the highest sensitivity rate possible with the highest F_1 score.
2. Obtain more reliable and stable results due to the availability of a larger number of observations than the one used in the previous case studies.
3. Obtain results in a relatively short period. The total time taken was 2.23 minutes, which may not be considered very fast analysis, but it is a great added value compared to the fact that we were not able to process the data using R alone.

6.2 Future Work

The are several inviting avenues of future work. The discussion and results mentioned earlier points towards a related potential limitation of the proposed approach and suggest possible improvements to my combination approach. I observed one important limitation of K-MICHA which is the small clusters that were obtained when the dataset is small and without using H2O, which leads to doubts regarding the reliability of the fraud probabilities assigned to these clusters. More advanced cluster analysis should

be automated to find the optimal set of final clusters. This optimal set should take into account:

1. Finding a specific threshold specific to each data set defining the acceptable number of observations in the clusters.
2. The possibility of combining certain clusters.
3. The performance of the methods, and their contribution to the final clusters obtained.
4. Assigning size-based weights to the clusters to quantify the reliability of the fraud probabilities obtained.
5. Relying on more than one cluster to define the final probability. A k-Nearest cluster approach similar to KNN might be a possible solution.

So far, we applied K-MICHA in a one node H2O cluster. A multi-node cluster is an essential step for the validation of the Big Data framework of K-MICHA.

The major suggested improvement would be a realtime framework for K-MICHA, where transactions or observations are analyzed in realtime and a fraud prediction is generated immediately. In this framework, learning would be incremental where the algorithms should be updated in realtime or very short periods basis to keep track of new possible patterns and strategies of fraud and to provide more accurate predictions.

References

[1] KPMG. Global banking fraud survey - the multi-faceted threat of fraud: Are banks up to the challenge ?, 2019.

[2] J. Han, M. Kamber, and J. Pei. *Data Mining: Concepts and Techniques*. Morgan Kaufman Publishers, 2003.

[3] B. Baesens, V. Van Vlasselaer, and W. Verbeke. *Fraud analytics using descriptive, predictive, and social network techniques: A guide to data science for fraud detection*. John Wiley & Sons, 2015.

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

[5] C. Phua, V. Lee, K. Smith, and R. Gayler. A comprehensive survey of data mining-based fraud detection research. pages 1–14, 2010.

[6] Y. Kou, C.-t. T. Lu, S. Sirwongwattana, Y. P. Huang, and S. Sinvongwattana. Survey of fraud detection techniques. *IEEE International Conference on Networking Sensing and Control*, pages 749–754, 2004.

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

[8] A. Abdallah, M. A. Maarof, and A. Zainal. Fraud detection system: A survey. *Journal of Network and Computer Applications*, 68:90–113, 2016.

[9] M. Kharote and V. P. Kshirsagar. Data Mining Model for Money Laundering Detection in Financial Domain. *International Journal of Computer Applications*, 85(16):61–64, 2014.

[10] O. Gottlieb, C. Salisbury, H. Shek, and V. Vaidyanathan. Detecting Corporate Fraud : An Application of Machine Learning. pages 1–5, 2006.

[11] K. Golmohammadi and O. R. Zaiane. Data Mining Applications for Fraud Detection in Securities Market. *The European Intelligence and Security Informatics Conference*, pages 107–114, 2012.

[12] L. W. Vona. *Fraud data analytics methodology: The fraud scenario approach to uncovering fraud in core business systems*. John Wiley & Sons, 2017.

[13] T. Coughlin. 175 zettabytes by 2025, 2018. Forbes.

[14] R. Jacobson. 2.5 quintillion bytes of data created every day. how does cpg and retail manage it?, 2013. IBM.

[15] D. D. Spann. *Fraud Analytics: Strategies and Methods for Detection and Prevention*. John Wiley & Sons, 2014.

[16] B. Robinson and J. Winteregg. A-Z of banking fraud 2016, 2016. Temenos and NetGuardians.

[17] Facts + statistics: Identity theft and cybercrime, 2019. Insurance Information Institute.

[18] Fraud the facts 2019- the definitive overview of payment industry fraud, 2019. UK Finance.

[19] M. Muhlbaier, A. Topalis, and R. Polikar. Learn.NC: Combining ensemble of classifiers with dynamically weighted consult-and-vote for efficient incremental learning of new classes. *IEEE Transactions on Neural Networks*, 20(1):152–168, 2009.

[20] H. He and E. A. Garcia. Learning from Imbalanced Data. *IEEE Transactions On Knowledge And Data Engineering*, 21(9):1263–1284, 2009.

[21] C. Phua, D. Alahakoon, and V. Lee. Minority report in fraud detection: classification of skewed data. *ACM SIGKDD explorations newsletter*, 6(1):50–59, 2004.

[22] Z.-H. Zhou and X.-Y. Liu. Training cost-sensitive neural networks with methods addressing the class imbalance problem. *IEEE Transactions on Knowledge and Data Engineering*, 18(1):63–77, 2006.

[23] S. Ertekin, J. Huang, and C. L. Giles. Active learning for class imbalance problem. *The 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 823–824, 2007.

[24] M. Wasikowski and X.-w. Chen. Combating the small sample class imbalance problem using feature selection. *IEEE Transactions on knowledge and data engineering*, 22(10):1388–1400, 2010.

[25] S. Wang and X. Yao. Multiclass imbalance problems: Analysis and potential solutions. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 42(4):1119–1130, 2012.

[26] P. Richhariya and P. K. Singh. Evaluating and emerging payment card fraud challenges and resolution. *International Journal of Computer Applications*, 107(14):5 – 10, 2014.

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

[28] Z. Qin, A. T. Wang, C. Zhang, and S. Zhang. Cost-sensitive classification with k-nearest neighbors. In *Knowledge Science, Engineering and Management*, pages 112–131. Springer Berlin Heidelberg, 2013.

[29] R. J. Bolton and D. J. Hand. Statistical fraud detection: A review. *Statistical science*, 17(3):235–249, 2002.

[30] D. J. Weston, D. J. Hand, N. M. Adams, and C. Whitrow. Plastic card fraud detection using peer group analysis. *Advances in Data Analysis and Classification*, 2(1):45–62, 2008.

[31] K. Ramakalyani and D. Umadevi. Fraud Detection of Credit Card Payment System by Genetic Algorithm. *International Journal of Scientific & Engineering Research*, 3(7):1–6, 2012.

[32] P. J. Bentley, J. Kim, G.-h. Jung, and J.-u. Choi. Fuzzy Darwinian Detection of Credit Card Fraud. pages 1–4, 2000.

[33] A. Srivastava, A. Kundu, S. Sural, and S. Member. Credit Card Fraud Detection Using Hidden Markov Model. *IEEE Transactions on Dependable and Secure Computing*, 5(1):37–48, 2008.

[34] S. Esakkiraj and S. Chidambaram. Predictive Approach for Fraud Detection Using Hidden Markov Model. *International Journal of Engineering Research & Technology*, 2(1):1–7, 2013.

[35] J. S. Mishra, S. Panda, and A. K. Mishra. A Novel Approach for Credit Card Fraud Detection Targeting the Indian Market. *International Journal of Computer Science*, 10(3):172–179, 2013.

[36] A. Brabazon, J. Cahill, P. Keenan, and D. Walsh. Identifying Online Credit Card Fraud using Artificial Immune Systems. In *IEEE Congress on Evolutionary Computation*, pages 1–7, 2010.

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

[38] D. Sanchez, M. A. Vila, L. Cerda, and J. M. Serrano. Association rules applied to credit card fraud detection. *ScienceDirect*, 36(2):3630–3640, 2009.

[39] Y. Sahin, S. Bulkan, and E. Duman. A cost-sensitive decision tree approach for fraud detection. *Expert Systems with Applications*, 40(15):5916–5918, 2013.

[40] A. C. Bahnsen, A. Stojanovic, and D. Aouada. Cost Sensitive Credit Card Fraud Detection using Bayes Minimum Risk. *The 12th International Conference on Machine Learning and Applications*, pages 333–338, 2013.

[41] A. E. Pasarica. Card fraud detection using learning machines. pages 29–45, 2014.

[42] Y. Sahin and E. Duman. Detecting Credit Card Fraud by Decision Trees and Support Vector Machines. *International Multiconference of Engineers and computer scientists*, pages 442–447, 2011.

[43] V. R. Ganji and S. N. P. Mannem. Credit card fraud detection using anti-k nearest neighbor algorithm. *International Journal on Computer Science and Engineering (IJCSE)*, 4(6):1035–1039, 2012.

[44] S. Ghosh and D. L. Reilly. Credit Card Fraud Detection with a Neural-Network. *The 27th Annual Hawaii International Conference on System Sciences*, pages 621–630, 1994.

[45] R. Dorronsoro, F. Ginel, S. Carmen, and C. S. Cruz. Neural Fraud Detection in Credit Card Operations. *IEEE Transactions on Neural Networks*, 8(4):827–834, 1997.

[46] V. Zaslavsky and A. Strizhak. Credit card fraud detection using self-organizing maps. *Information and Security*, 18:48–63, 2006.

[47] M. Syeda, Y.-q. Zbang, Y. Pan, and C. Science. Parallel Granular Neural Networks for Fast Credit Card Fraud Detection. *The IEEE International Conference on Fuzzy Systems*, pages 572–577, 2002.

[48] S. Maes, K. Tuyls, B. Vanschoenwinkel, and B. Manderick. Credit Card Fraud Detection Using Bayesian and Neural Networks. *The 1st International NAISO Congress on Neuro Fuzzy Technologies*, 2002.

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

[50] B. Subashini and K. Chitra. Enhanced System for Revealing Fraudulence in Credit Card Approval. *International Journal of Engineering Research & Technology*, 2(8):936–949, 2013.

[51] N. Mahmoudi and E. Duman. Detecting credit card fraud by Modified Fisher Discriminant Analysis. *Expert Systems with Applications*, 42(5), 2014.

[52] T. Fawcett and F. Provost. Adaptive Fraud Detection. *Data Mining and Knowledge Discovery*, 1(3):291–316, 1997.

[53] Y. Moreau, H. Verrelst, and J. Vandewalle. Detection of mobile phone fraud using supervised neural networks: A first prototype. *The 7th International Conference on Artificial Neural Networks*, pages 1065–1070, 1997.

[54] M. Taniguchi, M. Haft, J. Hollmén, and V. T. Siemens. Fraud Detection In Communications Networks Using Neural And Probabilistic Methods. *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 1241–1244, 1998.

[55] U. Murad and G. Pinkas. Unsupervised profiling for identifying superimposed fraud. *The 3rd European Conference on Principles of Data Mining and Knowledge Discovery*, pages 251–261, 1999.

[56] S. Rosset, U. Murad, E. Neumann, Y. Idan, and G. Pinkas. Discovery of fraud rules for telecommunications – challenges and solutions. *The Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 409–413, 1999.

[57] K. C. Cox, S. G. Eick, G. J. Wills, and R. J. Brachman. Visual Data Mining: Recognizing Telephone Calling Fraud. pages 1–6, 1997.

[58] M. H. Cahill, D. Lambert, M. Ave, and D. X. Sun. Detecting fraud in the real world. pages 1–18, 2000.

[59] C. Cortes, D. Pregibon, and C. Volinsky. Communities of interest. *The 4th International Conference on Advances in Intelligent Data Analysis*, pages 105–114, 2001.

[60] C. Cortes, D. Pregibon, C. Volinsky, and T. S. Labs. Computational Methods for Dynamic Graphs. *Journal of Computational and Graphical Statistics*, 12(4):950 – 970, 2003.

[61] P. A. Estevez, M. H. Claudio, and C. A. Perez. Subscription fraud prevention in telecommunications using fuzzy rules and neural networks. *Expert Systems with Applications*, 31(2):337–339, 2006.

[62] C. S. Hilas and J. N. Sahalos. User Profiling for Fraud Detection in Telecommunication Networks. 2005.

[63] O. Persons. Using Financial Statement Data to Identify Factors Associated with Fraudulent Financial Reporting. *Journal of Applied Business Research*, 11(3):38–46, 1995.

[64] M. Beasley. An empirical analysis of the relation between board of director composition and financial statement fraud. . *The Accounting Review*, 71(4):443–465, 1996.

[65] J. V. Hansen, J. B. McDonald, W. F. Messier, Jr., and T. B. Bell. A generalized qualitative-response model and the analysis of management fraud. *Management Science*, 42(7):1022–1032, 1996.

[66] C. T. Spathis. Detecting false financial statements using published data : some evidence from Greece. *Managerial Auditing Journal*, 17(4):179–191, 2002.

[67] E. Feroz, T. Kwon, V. Pastena, and K. Park. The efficacy of red flags in predicting the SEC's targets: an artificial neural networks approach. *International Journal of Intelligent Systems in Accounting, Finance and Management*, 9(3):145–157, 2000.

[68] M. Krambia-Kapardis, C. Christodoulou, and M. Agathocleous. Neural networks: the panacea in fraud detection ? *Managerial Auditing Journal*, 25(7):659–678, 2010.

[69] B. Hoogs, T. Kiehl, C. Lacomb, and D. Senturk. A genetic algorithm approach to detecting temporal patterns indicative of financial statement fraud. *Intelligent Systems in Accounting, Finance and Management*, 15(1-2):41–56, 2007.

[70] J. W. Lin, M. I. Hwang, and J. D. Becker. A fuzzy neural network for assessing the risk of fraudulent financial reporting. *Managerial Auditing Journal*, 18(8):657–665, 2003.

[71] F. Liou. Fraudulent financial reporting detection and business failure prediction models: A comparison. *Managerial Auditing Journal*, 23(7):650–662, 2008.

[72] J. Perols. Financial statement fraud detection: An analysis of statistical and machine learning algorithms. *AUDITING: A Journal of Practice & Theory*, 30(2):19–50, 2011.

[73] W. Zhou and G. Kapoor. Detecting evolutionary financial statement fraud. *Decision Support Systems*, 50(3):570–575, 2011.

[74] E. Kirkos, C. Spathis, and Y. Manolopoulos. Data Mining techniques for the detection of fraudulent financial statements. *Expert Systems with Applications*, 32(4):995–1003, 2007.

[75] P. Ravisankar, V. Ravi, G. R. Rao, and I. Bose. Detection of fi nancial statement fraud and feature selection using data mining techniques. *Decision Support Systems*, 50(2):491–500, 2011.

[76] J. Neville, D. Jensen, J. Komoroske, K. Palmer, and H. Goldberg. Using Relational Knowledge Discovery to Prevent Securities Fraud Categories and Subject Descriptors. *the 11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, page 449–458, 2005.

[77] D. Diaz, B. Theodoulidis, and P. Sampaio. Expert Systems with Applications Analysis of stock market manipulations using knowledge discovery techniques applied to intraday trade prices. *Expert Systems With Applications*, 38(10):12757–12771, 2011.

[78] Z. Ferdousi and A. Maeda. Unsupervised Outlier Detection in Time Series Data. *The 22nd International Conference on Data Engineering Workshops*, pages 51–56, 2006.

[79] H. Ögüt, M. Mete Doganay, and R. Aktas. Detecting stock-price manipulation in an emerging market: The case of Turkey. *Expert Systems with Applications*, 36(9):11944–11949, 2009.

[80] M. Artis, M. Ayuso, and M. Guillén. Detection of automobile insurance fraud with discrete choice models and misclassified discrete models claims. *The Journal of Risk and Insurance*, 69(3):325–340, 2002.

[81] G. Dionne and R. Gagne. Replacement Cost Endorsement and Opportunistic Fraud in Automobile Insurance. *Journal of Risk and Uncertainty*, 24(3):213–230, 2002.

[82] R. A. Derrig and K. M. Ostaszewski. Fuzzy Techniques of Pattern Recognition in Risk and Claim Classification. *The Journal of Risk and Insurance*, 62(3):447–482, 1995.

[83] P. L. Brockett, X. Xia, R. A. Derrig, and P. L. Brockett. Using Kohonen's Self-Organizing Feature Map to Uncover Automobile Bodily Injury Claims Fraud. *The Journal of Risk and Insurance*, 65(2):245–274, 1998.

[84] B. Stefano and B. Gisella. Insurance fraud evaluation: a fuzzy expert system. *The 10th IEEE International Conference on Fuzzy Systems.*, pages 1491–1494, 2001.

[85] T. Ormerod, N. Morley, L. Ball, C. Langley, and C. Spenser. Using ethnography to design a mass detection tool (mdt) for the early discovery of insurance fraud. *Extended Abstracts on Human Factors in Computing Systems*, pages 650–651, 2003.

[86] R. M. Musal. Expert Systems with Applications Two models to investigate Medicare fraud within unsupervised databases. *Expert Systems With Applications*, 37(12):8628–8633, 2010.

[87] P. A. Ortega, C. J. Figueroa, and G. A. Ruz. A medical claim fraud/abuse detection system based on data mining: A case study in Chile. *The International Conference on Data Mining*, pages 224–231, 2006.

[88] M. Tang, B. S. U. Mendis, D. W. Murray, Y. Hu, and A. Sutinen. Unsupervised Fraud Detection in Medicare Australia. *The 9th Australasian Data Mining Conference*, pages 103–110, 2011.

[89] W.-s. Yang and S.-y. Hwang. A process-mining framework for the detection of healthcare fraud and abuse. *Expert Systems with Applications*, 31(1):56–68, 2006.

[90] L. Bermudez, J. Perez, M. Ayuso, E. Gomez, and F. Vazquez. A Bayesian dichotomous model with asymmetric link for fraud in insurance. *Insurance Mathematics and Economics*, 42(2):779–786, 2008.

[91] J. Pérez, J. Muguerza, O. Arbelaitz, I. Gurrutxaga, and J. I. Martin. Consolidated tree classifier learning in a car insurance fraud detection domain with class. *The Third international conference on Advances in Pattern Recognition*, pages 381 – 389, 2005.

[92] R. Bhowmik. Detecting Auto Insurance Fraud by Data Mining Techniques. *Journal of Emerging Trends in Computing and Information Sciences*, 2(4):156–162, 2011.

[93] P. L. Brockett and L. L. Golden. A comparison of neural network , statistical methods , and variable choice for life insurers' financial distress prediction. *The Journal of Risk and Insurance*, 73(3):397–419, 2006.

[94] W. Xu, S. Wang, D. Zhang, and B. Yang. Random Rough Subspace based Neural Network Ensemble for Insurance Fraud Detection. *Fourth International Joint Conference on Computational Sciences and Optimization*, pages 2–6, 2011.

[95] J. Pathak and S. L. Summers. A fuzzy-based algorithm for auditors to detect elements of fraud in settled insurance claims. *Managerial Auditing Journal*, 20(6):632–644, 2005.

[96] H. Tao, L. Zhixin, and S. Xiaodong. Insurance Fraud Identification Research Based on Fuzzy Support Vector Machine with Dual Membership. *International Conference on Information Management, Innovation Management and Industrial Engineering*, pages 457–460, 2012.

[97] T. E. Senator, H. G. Goldberg, M. A. Cottini, A. F. U. Khan, W. M. Llamas, and M. P. Marrone. The FinCEN Artificial Intelligence System : Identifying Potential Money Laundering from Reports Transactions. *The 7th Conference on Innovative Applications of Artificial Intelligence*, pages 156–170, 1995.

[98] B. Chartier and T. Spillane. Money laundering detection with a neural network. *Business Applications of Neural Networks*, pages 159–172, 2000.

[99] S. N. Wang and J. G. Yang. A Money Laundering Risk Evaluation Method Based on Decision Tree. *The Sixth International Conference on Machine Learning and Cybernetics*, pages 283–286, 2007.

[100] L.-T. Lv, N. Ji, and J.-L. Zhang. A RBF neural network model for anti-money laundering. *The International Conference on Wavelet Analysis and Pattern Recognition*, pages 209–215, 2008.

[101] X. Wang and G. Dong. Research on money laundering detection based on improved minimum spanning tree clustering and its application. *The 2nd International Symposium on Knowledge Acquisition and Modeling*, pages 62–64, 2009.

[102] A. S. Larik and S. Haider. Clustering based anomalous transaction reporting. *Procedia Computer Science*, 3:606–610, 2011.

[103] L. Keyan and Y. Tingting. An improved support-vector network model for anti-money laundering. *The 5th International Conference on Management of e-Commerce and e-Government*, pages 193–196, 2011.

[104] D. G. Perez and M. M. Lavalle. Outlier detection applying an innovative user transaction modeling with automatic explanation. *IEEE Electronics, Robotics and Automotive Mechanics Conference*, pages 41–46, 2011.

[105] R. Liu, X. l. Qian, S. Mao, and S. z. Zhu. Research on anti-money laundering based on core decision tree algorithm. *Chinese Control and Decision Conference (CCDC)*, pages 4322–4325, 2011.

[106] E. A. Lopez-Rojas and S. Axelsson. Money Laundering Detection using Synthetic Data. *The 27th annual workshop of the Swedish Artificial Intelligence Society*, pages 33–40, 2012.

[107] N. S. Khan, A. S. Larik, Q. Rajput, and S. Haider. A Bayesian Approach for Suspicious Financial Activity Reporting. *International Journal of Computers and Applications*, 35(4):181–187, 2013.

[108] G. Krishnapriya, M. Phil, and M. Prabakaran. Money laundering analysis based on time variant behavioral transaction patterns. *Journal of Theoretical and Applied Information Technology*, 67(1):12–17, 2014.

[109] N. Heidarinia, A. Harounabadi, and M. Sadeghzadeh. An Intelligent Anti-Money Laundering Method for Detecting Risky Users in the Banking Systems. *International Journal of Computer Applications*, 97(22):35–39, 2014.

[110] C. R. Alexandre and J. Balsa. A multiagent based approach to money laundering detection and prevention. *International Conference on Agents and Artificial Intelligence*, pages 230–235, 2015.

[111] C. Suresh, K. T. Reddy, and N. Sweta. A Hybrid Approach for Detecting Suspicious Accounts in Money Laundering Using Data Mining Techniques. *International Journal of Information Technology and Computer Science*, 5:37–43, 2016.

[112] W. Lee and S. Stolfo. Data mining approaches for intrusion detection. *The 7th USENIX security symposium*, pages 79–94, 1998.

[113] S.-P. W. Shieh and V. D. Gligor. A pattern oriented intrusion model and its applications. *The IEEE Computer Society Symposium on Research in Security and Privacy*, pages 327–342, 1991.

[114] W.-H. Ju and Y. Vardi. A hybrid high-order Markov chain model for computer intrusion detection. *Journal of Computational and Graphical Statistics*, 10(2):277–295, 2001.

[115] S. Forrest, S. Hofmeyr, Somayaji, A., and T. Longstaff. A sense of self for Unix processes. *The IEEE Symposium on Security and Privacy*, pages 120–128, 1996.

[116] J. Ryan, M. Lin, and R. Miikkulainen. Intrusion detection with neural networks. *AAAI Workshop: AI Approaches to Fraud Detection and Risk Management*, pages 72–79, 1997.

[117] M. Schonlau, W. DuMouchel, W. H. Ju, A. F. Karr, M. Theus, and Y. Vardi. Computer intrusion: detecting masquerades. *Statistical Science*, 16(1):58–74, 2001.

[118] K. Yoshida, F. Adachi, T. Washio, H. Motoda, T. Homma, A. Nakashima, H. Fujikawa, and K. Yamazaki. Density Based Spam Detector. *IEICE transactions on information and systems*, E87-D(12):2678–2688, 2004.

[119] T. Lane and C. . Brodley. An Empirical Study of Two Approaches to Sequence Learning for Anomaly Detection. *Machine Learning*, 51:486–493, 2003.

[120] W. Fan, M. Miller, S. Stolfo, W. Lee, and P. Chan. Artificial Anomalies to Detect Unknown and Known Network Intrusions. *IEEE International Conference on Data Mining*, pages 123–130, 2001.

[121] K. Sequeira and M. Zaki. ADMIT : Anomaly-based Data Mining for Intrusions. *The 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 386–395, 2002.

[122] S. Hawkins, H. He, G. Williams, and R. Baxter. Outlier Detection Using Replicator Neural Networks. *International Conference on Data Warehousing and Knowledge Discovery*, pages 170–180, 2002.

[123] G. Williams, R. Baxter, H. He, H. Hawkins, and L. Gu. A Comparative Study of RNN for Outlier Detection in Data Mining . *IEEE International Conference on Data Mining*, pages 709–712, 2002.

[124] B. Idris and B. Shanmugam. Novel Attack Detection Using Fuzzy Logic and Data Mining. *International Conference on Security & Management*, pages 26 – 31, 2006.

[125] T. Peng and W. Zuo. Data Mining for Network Intrusion Detection System in Real Time. *International Journal of Computer Science and Network Security*, 6(2):173–177, 2006.

[126] A. Patcha and J. Park. A Game Theoretic Approach to Modeling Intrusion Detection in Mobile Ad Hoc Networks. *IEEE Workshop on Information Assurance and Security*, pages 10–11, 2004.

[127] N. Dalvi, P. Domingos, Mausam, S. Sanghai, and D. Verma. Adversarial Classification. *The 10th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 99–108, 2004.

[128] K. Veeramachaneni, I. Arnaldo, A. Cuesta-Infante, V. Korrapati, C. Bassias, and K. Li. AI^2 : Training a big data machine to defend. *IEEE 2nd International Conference on Big Data Security on Cloud (Big Data Security)*, pages 49–54, 2016.

[129] J. J. Xu, Y. Lu, and M. Chau. P2P Lending Fraud Detection: A Big Data Approach. *Pacific-Asia Workshop on Intelligence and Security Informatics*, pages 71–81, 2015.

[130] H. Hormozi, M. K. Akbari, E. Hormozi, and M. S. Javan. Credit cards fraud detection by negative selection algorithm on hadoop (To reduce the training time). *The 5th Conference on Information and Knowledge Technology*, pages 40–43, 2013.

[131] S. Kamaruddin and V. Ravi. Credit Card Fraud Detection using Big Data Analytics : Use of PSOAANN based One-Class Classification. *The International Conference on Informatics and Analytics*, pages 33:1–33:8, 2016.

[132] G. S. Sadasivam, S. Mutyala, H. Dasaraju, P. P. Bhanu, and S. M. Lakshme. Corporate governance fraud detection from annual reports using big data analytics. *International Journal of Big Data Intelligence*, 3(1):51–60, 2016.

[133] A.-R. Bologa, R. Bologa, and A. Florea. Big Data and Specific Analysis Methods for Insurance Fraud Detection. *Database Systems Journal*, 4(4):30–39, 2013.

[134] P. Dora and G. H. Sekharan. Healthcare Insurance Fraud Detection Leveraging Big Data Analytics. *International Journal of Science and Research*, 4(4):2073–2076, 2015.

[135] J. T. S. Quah and M. Sriganesh. Real-time credit card fraud detection using computational intelligence. *Expert Systems with Applications*, 35(4):1721–1732, 2008.

[136] A. Kundu, S. Panigrahi, S. Sural, and A. K. Majumdar. Blast-ssaha hybridization for credit card fraud detection. *IEEE Transactions on Dependable and Secure Computing*, 6(4):309–315, 2009.

[137] K. K. Sherly and R. Nedunchezhian. Boat adaptive credit card fraud detection system. *The IEEE International Conference on Computational Intelligence and Computing Research*, pages 1–7, 2010.

[138] T. Minegishi and A. Niimi. Detection of Fraud Use of Credit Card by Extended VFDT. *World Congress on Internet Security*, pages 152–159, 2011.

[139] J. Hollmen and V. Tresp. Call-based Fraud Detection in Mobile Communication Networks using a Hierarchical Regime-Switching Model. *Conference on Advances in neural information processing systems II*, pages 889–895, 1999.

[140] B. Sun, F. Yu, K. Wu, Y. Xiao, and V. C. M. Leung. Enhancing security using mobility-based anomaly detection in cellular mobile networks. *IEEE Transactions on Vehicular Technology*, 55(4):1385–1396, 2006.

[141] A. Krenker, M. Volk, U. Sedlar, J. Bešter, and A. Kos. Bidirectional artificial neural networks for mobile-phone fraud detection. *Electronics and Telecommunications Research Institute Journal (ETRI)*, 31(1):92–94, 2009.

[142] M. L. Huang, J. Liang, and Q. V. Nguyen. A visualization approach for frauds detection in financial market. *The 13th International Conference Information Visualisation*, pages 197–202, 2009.

[143] C. Francis, N. Pepper, and H. Strong. Using Support Vector Machines to Detect Medical Fraud and Abuse. *The 33rd Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 8291–8294, 2011.

[144] Y.-h. Tsai, C.-h. Ko, and K.-c. Lin. Using Common KADS Method to Build Prototype System in Medical Insurance Fraud Detection. *Journal Of Networks*, 9(7):1798–1802, 2014.

[145] S. Cho. Incorporating Soft Computing Techniques Into a Probabilitistic Intrusion Detection System. *IEEE Transactions on Systems, Man and Cybernetics*, 32(2):154–160, 2002.

[146] G. Prasad, Y. Dhanalakshmi, V. Kumar, and I. Babu. Modeling An Intrusion Detection System Using Data Mining And Genetic Algorithms Based On Fuzzy Logic. *International Journal of Computer Science and Network Security*, 8(7):319–325, 2008.

[147] Y. Dhanalakshmi and R. Babu. Intrusion Detection Using Data Mining Along Fuzzy Logic and Genetic Algorithms. *International Journal of Computer Science and Network Security*, 8(2):27–32, 2008.

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

[149] F. N. Ogwueleka. Data Mining Application in Credit Card Fraud Detection System. *Journal of Engineering Science and Technology*, 6(3):311–322, 2011.

[150] R. Patidar and L. Sharma. Credit Card Fraud Detection Using Neural Network. *International Journal of Soft Computing and Engineering*, 1:13–14, 2011.

[151] H. Farvaresh and M. Mehdi. A data mining framework for detecting subscription fraud in telecommunication. *Engineering Applications of Artificial Intelligence*, 24(1):182–184, 2011.

[152] W. Chai, B. K. Hoogs, and B. T. Verschueren. Fuzzy ranking of financial statements for fraud detection. *The IEEE International Conference on Fuzzy Systems*, pages 152–158, 2006.

[153] M. J. Lenard, A. L. Watkins, and P. Alam. Effective use of integrated decision making: An advanced technology model for evaluating fraud in service-based computer and technology firms. *Journal of Emerging Technologies in Accounting*, 4(1):123–137, 2007.

[154] S. Kotsiantis, E. Koumanakos, D. Tzelepis, and V. Tampakas. Forecasting Fraudulent Financial Statements Using Data Mining. *International Journal of Computational Intelligence*, 3(2):104–110, 2006.

[155] S. Chen. Detection of fraudulent financial statements using the hybrid data mining approach. *SpringerPlus*, 5(89):1–16, 2016.

[156] M. Blume, C. Weinhardt, and D. Seese. Using network analysis for fraud detection in electronic markets. *Information management and market engineering*, pages 102–112, 2006.

[157] G. Williams and Z. Huang. Mining the Knowledge Mine: The Hot Spots Methodology for Mining Large Real World Databases. *The 10th Australian Joint Conference on Artificial Intelligence.*, pages 340–348, 1997.

[158] G. J. Williams. Evolutionary hot spots data mining. *The 3rd Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 184–193, 1999.

[159] S. Viaene, R. A. Derrig, and G. Dedene. A Case Study of Applying Boosting Naive Bayes to Claim Fraud Diagnosis. *IEEE Transactions on Knowledge and Data Engineering*, 16(5):612–620, 2004.

[160] N. Le Khac, S. Markos, and M.-T. Kechadi. A data mining-based solution for detecting suspicious money laundering cases in an investment bank. *The 2nd International Conference on Advances in Databases Knowledge and Data Applications*, pages 235–240, 2010.

[161] M. Kuhn and K. Johnson. *Applied Predictive Modeling*. Springer, 2013.

[162] L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

[163] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[164] Z. Huang. Extensions to the k-means algorithm for clustering large data sets with categorical values. *Data Mining and Knowledge Discovery*, 2(3):283–304, 1998.

[165] E. A. Lopez-Rojas, A. Elmir, and S. Axelsson. PaySim: A financial mobile money simulator for fraud detection. *The 28th European Modeling and Simulation Symposium-EMSS*, 2016.

[166] Y. Freund and R. Schapire. Experiments with a new boosting algorithm. *The 13th International Conference on Machine Learning*, pages 148–156, 1996.

[167] C.-C. Chang and C.-J. Lin. LIBSVM : a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3, Article 27), 2011.

[168] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. SMOTE: synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16(1):321–357, 2002.

[169] B. Schölkopf, J. C. Platt, J. C. Shawe-Taylor, A. J. Smola, and R. C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471, 2001.

Appendix A

Classification Methods for Fraud Detection

The most common and known classification methods used for fraud detection are briefly described in this appendix. We will also describe the approaches used to tackle the class imbalance issue, that we used in our exparimental study [27]. In the following, the "positive" observations represent the minority group of the data, *i.e.* the fraud (denoted 1). The "negative" observations represent the other class (denoted 0).

A.1 Machine Learning Classification Algorithms

A.1.1 Decision Tree (C5.0)

One of the most common decision tree algorithms, it's an advanced version of the C4.5 algorithm that has additional features like boosting and assigning different costs to classes [161]. These two algorithms differ from the CART algorithm by using the cross entropy (information statistics and information gain) instead of Gini index when evaluating the splits. A split is a "*variable* < (or >) *value*" rule that is used to divide a certain node in two daughter nodes. These splits need to be evaluated to find the one that best discriminates the target variable. When using C4.5 or C5.0, this evaluation is done by calculating the information gain after each split, the greater this quantity is, the better. this is calculated as follows:

$$gain(\text{split}) = info(\text{prior to split}) - info(\text{after the split})$$

Where

$$info(\text{prior to split}) = -\left[\frac{N_1}{N} \times \log\frac{N_1}{N}\right] - \left[\frac{N_0}{N} \times \log\frac{N_0}{N}\right]$$

Where N_1, N_0 and N are respectively the frequency of positives, negatives and the total number of observations in the parent node. *info*(after the split) is sum of the *info* for the two resulting nodes: *greater than split* and *less than split*, each multiplied by $\frac{n_i}{N}$, where n_i is the number of observations in the node.

A.1.2 Support Vector Machines (SVM)

SVM is a classification tool aiming to find the hyperplane that best separates data points in two classes [167]. Formally, given a training vector x_i in $\mathbb{R}^n$, $i = 1,\ldots,l$, n is the number of exploratory variables, and l is the number of observations in the train set. $y \in \mathbb{R}^l$ taking the values of 1 and -1. The binary classification is done by solving the following optimization problem.

$$\text{minimize} \quad \frac{1}{2}||w||^2 + C\sum_{i=1}^{l}\zeta_i$$

$$\text{subject to} \begin{cases} y_i \times (w^T\Phi(x_i) + b) \geq 1 - \zeta_i \\ \zeta_i \geq 0, i = 1,\ldots,l \end{cases}$$

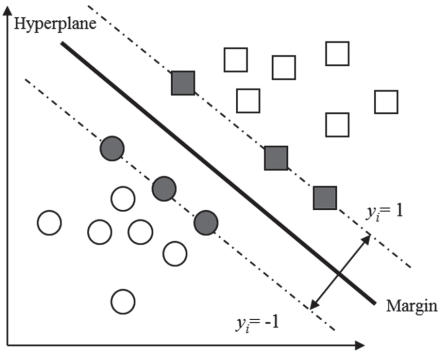


Fig. A.1 SVM classification

The hyperplane equation is defined as: $w^T\Phi(x_i) + b$, where w is a vector of weights, $\Phi(x_i)$ maps x_i into a higher-dimensional space. Slack variables ζ_i are added, to allow for some errors or miscalculations, in case these points are not linearly separable. C is a cost parameter > 0 associated with these errors. The aim of minimizing $\frac{1}{2}||w||^2$ is to maximize the distance between the two margins which is equal to $\frac{2}{||w||^2}$ in order to find the hyperplane that best separates the two classes (Fig.A.1).

A.1.3 Artificial Neural Network (ANN)

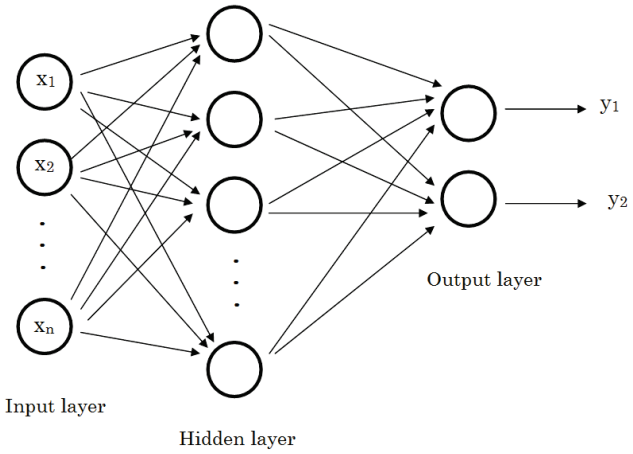


Fig. A.2 The multilayer perceptron model

ANN is a connection of multiple neurons or nodes. The multilayer feed-forward perceptron is an ANN architecture formed of several layers, an input, one or more hidden layers and an output layer (Fig. A.2). The first layer contains the input nodes representing the exploratory variables. These inputs are multiplied with a specific weight and transferred to each of the hidden layer's nodes where they are added together with a certain bias. An activation function is then applied to this summation to produce the output of the neuron, that will be transferred to the next layer. Finally, the output layer that provides the system's response, one of the classes 1 or 0. The weights used are first set randomly, then using the training set these weights are adjusted to minimize the error using specific algorithms like back-propagation [2].

A.1.4 Naïve Bayes (NB)

NB uses Bayes conditional probability rule for classification [2]. This method consists of finding a class to the new observation that maximizes its probability given the values of the variables. Specifically, we want to find the value of Y that maximizes $P(Y/X_1,X_2,\ldots,X_n)$.
Using the Bayes theorem:

$$P(Y/X_1,X_2,\ldots,X_n)=\frac{P(X_1,X_2,\ldots,X_n/Y)P(Y)}{P(X_1,X_2,\ldots,X_n)}$$

Maximizing $P(Y/X_1,X_2,\ldots,X_n)$ is equivalent to maximizing $P(X_1,X_2,\ldots,X_n/Y)$ which can be easily estimated from the historical data; assuming class-conditional indepen-

dence among variables:

$$P(X_1, X_2, \ldots, X_n / Y) = P(X_1/Y)P(X_2/Y)\ldots P(X_n/Y)$$

This assumption is not always verified or realistic. Another limitation of this method is the discretization of the continuous variables which means that some information may be lost, or these variables are assumed approximately normally distributed which may not be true.

A.1.5 Bayesian Belief Network (BBN)

These networks are graphical models for probabilistic relationships between a set of variables. They were developed to relax the *independence assumption* in NB, and thus allowing for dependencies among variables [2].

Random variables are represented as nodes and conditional dependencies between variables are represented as arcs between nodes. Each node is linked to a conditional probability table that generates probabilities of the node's variable conditionally to values of the parent's node.

The first step is to find a structure for the network. It may be constructed by human experts or inferred from the data. Several algorithms exist for learning the network topology from the training data given observable variables. Once this topology is found, training the network is straightforward from the historical data as in NB. It consists of computing the conditional probability tables entries, as is similarly done when computing the probabilities involved in naïve Bayesian classification.

Same as NB, continuous variables are either discretized or assumed normally distributed. Also, this method assumes that each node is independent of its non descendants given its parents in the graph, this is known as the *Markov condition*.

A.1.6 Logistic Regression (LR)

Logistic regression is a type of generalized linear models [161]. Using simple linear regression is inappropriate when the variable to be predicted is binary. The vector $\alpha = (\alpha_0, \alpha_1, \alpha_2, \ldots, \alpha_n)$ represents the coefficients, $X = (1, X_1, X_2, \ldots, X_n)$ the exploratory variables and ϵ the model's error. The linear model is then:

$$Y = \alpha_0 + \alpha_1 X_1 + \alpha_2 X_2 + \cdots + \alpha_n X_n + \epsilon = X\alpha + \epsilon$$

Linear regression assumes the continuity of the response variable Y. Therefore, a logit link function g over $[0,1]$ in $\mathbb{R}$ is introduced, to force the linear combination of the variables to take values between 0 and 1: $g(p)=X\alpha$, where p is the probability of positives we are estimating. The logit function is defined as:

$$g(p)=\ln\frac{p}{1-p} \quad \text{with} \quad p=\frac{\exp^{X\alpha}}{1+\exp^{X\alpha}}$$

A.1.7 K-Nearest Neighbour (KNN)

This method consists of using the k nearest points to the one we aim to predict [2]. A specific norm is used to measure the distance between points. The new observation is assigned the class with the majority of the k nearest points. The norm usually used to measure the distance between two observations p and q (an observation is $\in\mathbb{R}^n$) is the euclidean distance:

$$d(p,q)=\sqrt{\sum_{i=1}^{n}(p_i-q_i)^2}$$

A.1.8 Artificial Immune Systems (AIS)

AIS are concerned with extracting the role of the immune system to create computational and predictive systems. One of the most common algorithms is the Negative Selection Algorithm (NSA). The negative selection is done through two phases, the *detector set generation* and the *classification* [130].

In the first phase, random observations are generated and compared to the observations in the *self* set (i.e legitimate cases). If these random observations match a self observation, then they are rejected. Otherwise they are considered as fraud detectors and form the detector set. However, in the classification phase, if a new observation matches at least one detector, it is classified as a fraud.

The matching is measured using the euclidean distance and a specific threshold. A huge amount of time it takes to find the detector set is a disadvantage of this method specially when using a high threshold or big data.

A.2 Imbalanced Classification Approaches

There are two tactics for imbalanced classification approaches. The first one is applied on the data as a prepossessing step to balance classes, like oversampling, undersampling,

etc. The other one is implemented within the classification algorithm like cost-sensitive approaches or one-class classification.

A.2.1 Random Oversampling (RO)

It's a technique used to balance the classes by simply replicating observations as needed until balance between classes is reached. Our purpose is to modify the behavior of the classification model to concentrate on both minority class and majority class equally.

More advanced oversampling techniques were developed. The most common one is the Synthetic Minority Oversampling Technique (SMOTE), where minority class observations are oversampled by taking each of them and introducing synthetic examples along the line segments joining any or all of the k minority class nearest neighbors. Neighbors from the k-nearest neighbors are chosen depending upon the amount of over sampling required [168].

A.2.2 One-Class Classification (OCC)

This approach uses data from one class only (usually the minority class) and learns its characteristics. In this case, the classification is done in the testing phase. After training the algorithm with one class, it should be able to determine whether a certain observation belongs to the minority class or not.

OCC SVM: The purpose of this method is to find a "small" region capturing most of the training data points. This is done by estimating a function f taking the value 1 if a point is in this region and -1 elsewhere [169]. First, points are mapped using $\Phi : X \rightarrow F$. Then, in this feature space F, these points are separated from the origin with maximum margin using a hyperplane of equation: $w^T \Phi(x_i) = \rho$. Our aim is to maximize the margin that is equal to $\frac{\rho}{\|w\|}$. The optimization problem is then:

$$\text{minimize} \quad \frac{1}{2}\|w\|^2 + \frac{1}{\nu l}\sum_{i=1}^{l}\zeta_i - \rho$$

$$\text{subject to} \quad \begin{cases} w^T \Phi(x_i) \geq \rho - \zeta_i \\ \zeta_i \geq 0, i = 1, \ldots, l \end{cases}$$

Where (w, ρ) are a weight vector and offset parameter for the hyperplane in space F.

AANN: This is an approach that is usually used as a OCC ANN, using unlabeled or one class data [131]. It's an ANN with a specific architecture, with same number of nodes in the input and output layers. The AANN is trained using only the exploratory variables of the minority class. Then the average error of the training phase is calculated and used as threshold for classification, if the average error of the estimation of a new observation is higher than this threshold, then it belongs to the majority class.

A.2.3 Cost-Sensitive models (CS)

The basic idea behind CS models is to assign higher weight to the minority class. It's equivalent to specifying a higher cost to wrongly classify a minority observation.

CS C5.0: Assiging costs to the decision tree is different according to each algorithm. When using CART algorithm, the costs are added to the Gini index when splitting the data . For the C5.0, costs are implemented to the decision boundaries, not in the training algorithm [161], so the revised decision boundary for classifying an observation into class 1 is:

$$\frac{p_1}{p_0} > \frac{C_{1/0}}{C_{0/1}} \frac{\pi_0}{\pi_1}$$

Where π_i is the prior probability of an observation to be in class i. p_i and $C_{j/i}$ are respectively the estimator of the probability and the cost of wrongly classifying an observation of class i as j.

CS SVM: It is done by assigning weights to each class. Instead of having just one cost parameter C like the one in SVM (section A.1.2), two parameters C^+ and C^- are added as follows [167]:

$$\text{minimize} \quad \frac{1}{2}||w||^2 + C^+ \sum_{y_i=1} \zeta_i + C^- \sum_{y_i=-1} \zeta_i$$

$$\text{subject to} \quad \begin{cases} y_i \times (w^T \Phi(x_i) + b) \geq 1 - \zeta_i \\ \zeta_i \geq 0, i = 1, \ldots, l \end{cases}$$

Appendix B

Variable Selection in Logistic Regression

A high VIF is a result of high colinearity between the variables. However, the removal of all variables may result in unnecessary loss of information. In other words, when multiple variables are resulting in high VIF, they might all be correlated to the same variable, thus the removal of just this one variable might solve the problem.

B.1 Case Study 2: Mobile Payment Fraud Detection

For case study 2 with the mobile payment fraud detection data set, we investigate the scatter plot of the four variables that show high VIF: oldbalanceOrg, newbalanceOrg, oldbalanceDest and newbalanceDest (Fig. B.1).

A high linearity is present between oldbalanceOrg and newbalanceOrg, and between oldbalanceDest and newbalanceDest. This means that the removal two variables, specifically just one from each linearity couple of variables should solve the issue. We removed then the newbalanceOrg and newbalanceDest. The new VIF results are shown in Table B.1 proving our analysis.

Table B.1 New VIF test results for LR - case study 2

Variables	VIF
type	1.158
amount	1.056
oldbalanceOrg	1.150
oldbalanceDest	1.024

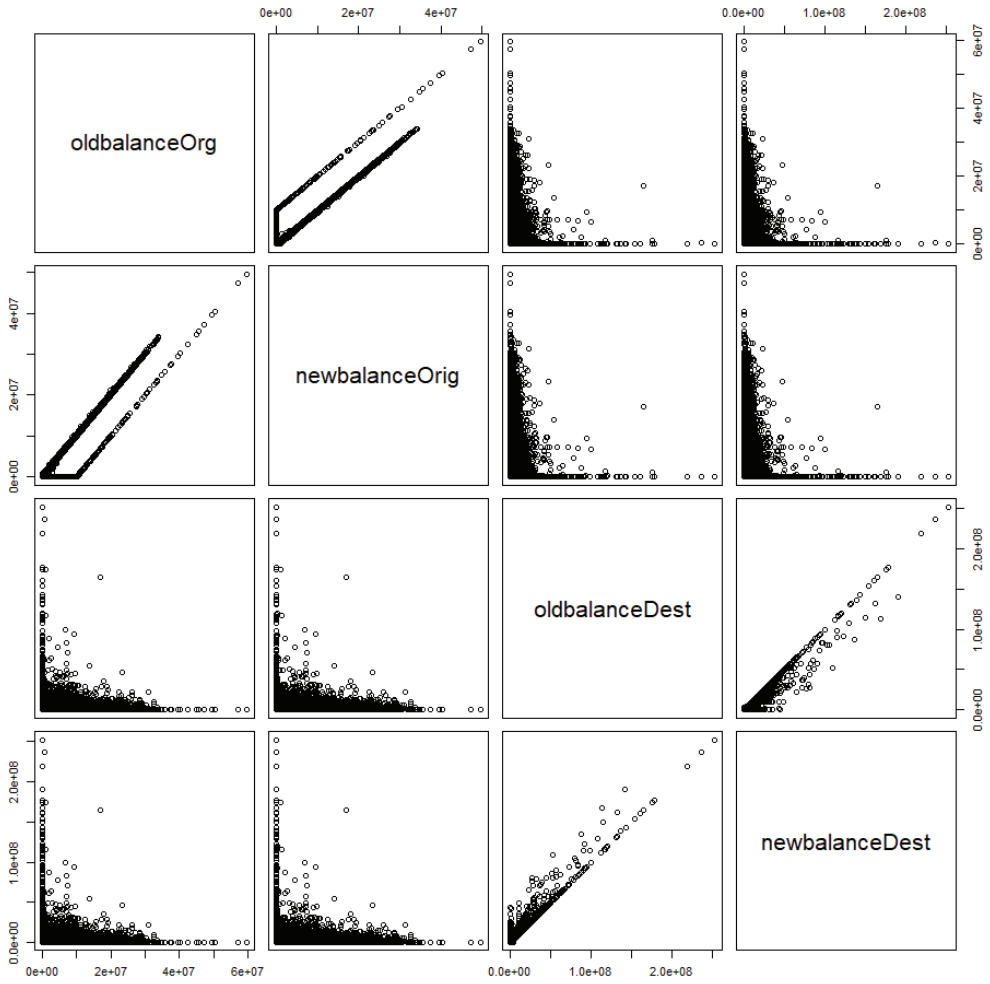


Fig. B.1 Pairs of scatter plot of variables with high VIF - case study 2

.

B.2 Case Study 3: Auto Insurance Fraud Detection

For case study 3 with the auto insurance fraud detection data set, we investigate the scatter plot of the four variables that show high VIF: total_claim_amount, injury_claim, property_claim and vehicle_claim (Fig. B.2). The linearity is clearly present in these plots between the two variables total_claim_amount and vehicle_claim, which means that the removal of just one of them should solve our model. We removed then the vehicle_claim variable. The new VIF results are shown in Table B.2 proving our analysis.

Table B.2 New VIF test results for LR - case study 3

Variables	VIF
age	9.140
authorities_contacted	1.206
auto_make	1.182
auto_year	1.108
bodily_injuries	1.088
capital.gains	1.210
capital.loss	1.197
collision_type	1.601
incident_city	1.220
incident_severity	1.375
incident_type	5.208
injury_claim	3.575
insured_education_level	1.235
insured_occupation	1.195
insured_sex	1.184
insured_relationship	1.071
months_as_customer	9.105
number_of_vehicles_involved	5.378
police_report_available	1.093
policy_annual_premium	1.154
policy_deductable	1.265
property_claim	4.442
property_damage	1.107
total_claim_amount	9.629
witnesses	1.139

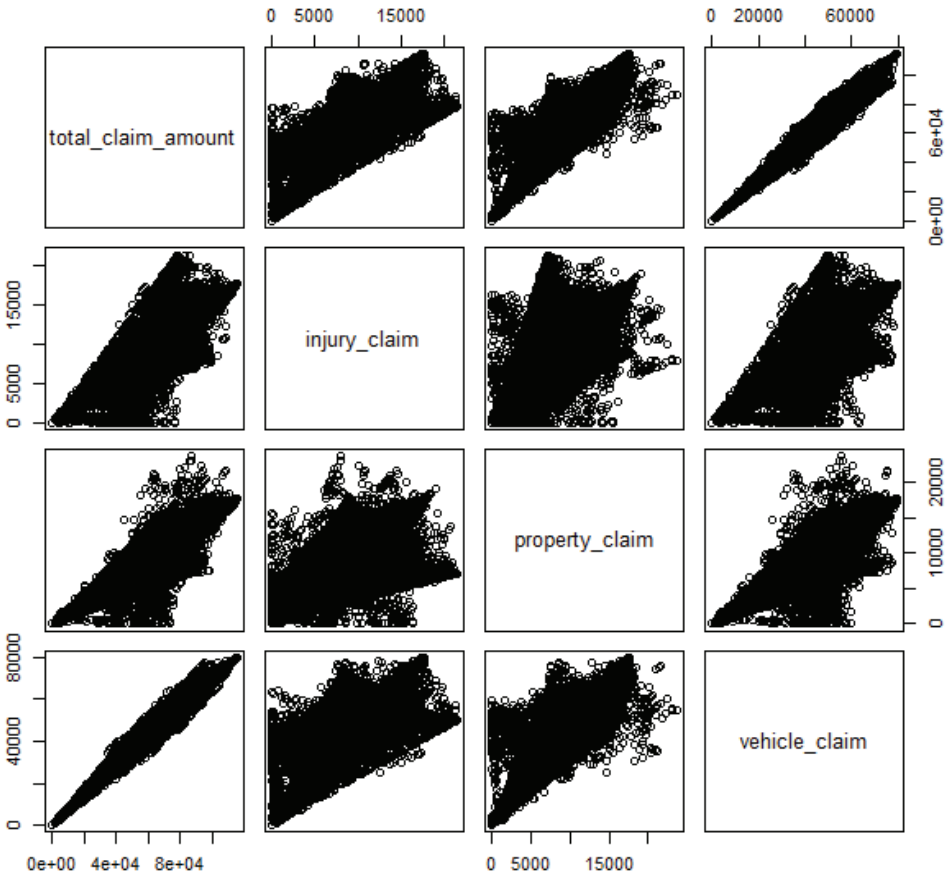


Fig. B.2 Pairs of scatter plot of variables with high VIF - case study 3