Comparative study of Machine learning techniques for loan fraud prediction

---

Rinae Tshivhidzo

Supervisor(s):
Dr Wilbert Chagwiza

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

in the

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

30 September 2022

Declaration

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

Rinae Tshivhidzo

30 September 2022

Abstract

Loan fraud is a major and growing problem. Due to this problem, millions of amounts are lost yearly. There has been a significant amount of research on fraud prediction. However, there has been less research on loan fraud prediction. It could be because of a lack of data available for analysis. This research aimed to compare machine learning algorithms to predict fraud practices in loan administration to find techniques that present the most accurate results. The dataset used is the Kaggle fraud detection dataset for the year 2019. Four machine learning algorithms, random forest, extreme gradient boost, adaptive boosting, and multilayer perceptron, were evaluated. The results obtained during the first attempt show that extreme gradient boost performed best compared to the other three models, with an area under the curve score (AUC) of 0.74. The results from the second attempt show that adaptive boosting performed best with an AUC score of 1.00, followed by extreme gradient boosting with an AUC score of 0.94.

Acknowledgements

Most importantly, acclaim and gratitude to the Almighty God for His showers of blessings throughout my journey of completing the research report successfully. To my supervisor, Dr. W. Chagwiza, I would like to express my deep and sincere appreciation for his guidance throughout this research journey. It was a great privilege and honour to work under his guidance. The DSI-NICIS National e-Science Postgraduate Teaching and Training Platform (NEPTTP) support towards this research is hereby acknowledged". Many thanks to my parents for their support and sacrifices for educating and preparing me for my future. Last but not least, I would like to thank all those who supported me through the journey of completing this research.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Fraud is a continually rising threat with far-reaching consequences in areas such as financial institutions, corporate institutions, and the government. In addition, the focus can be too centered on financial loss. However, in reality, fraud has far more of an influence than this. Fraud can cause substantial harm to people, public institutions, services, and the environment. Any sort of fraud, whether it is committed by opportunistic people or major criminal network groups, can have devastating repercussions. It can diminish trust in business, destabilise investment funds and influence the cost of living [24].

In the past two years, 47% of organizations reported experiencing fraud, which is the highest recorded level in the previous 20 years. Since 2018, the percentage of organizations that experience fraud has generally remained steady. However, the survey of 1,296 executives from 53 nations and regions revealed a growing threat from outside perpetrators. Approximately 70% of organizations that experienced fraud said the most frustrating occurrence was caused by an external attack or coordination between external and internal sources [50].

Identity theft, check fraud, internet sales, credit card fraud, and loan fraud are some of the most common types of fraud. [20][66]. Identity fraud is when someone steals your identity and uses your information to steal money. Cheque fraud occurs when an individual pays for something with a cheque knowing it does not have enough funds or when an individual forges a stolen cheque. Internet sales fraud involves selling fake items or taking payment with no intention of delivering the item. Credit

card fraud occurs when fraudsters obtain people's credit card information and use them without the owner's knowledge.

The type of fraud addressed in this project report include loan fraud. Loan fraud occurs when an individual falsely applies for a loan, either a personal or business loan. For example, if the borrower lies on their loan application, the bank or other lending establishment will suffer financial setbacks in their business operations. Money lenders can also commit loan fraud against prospective borrowers. An illustration of this is a bank giving a false advance application to a likely borrower in a real estate transaction. Providing a false application could make the purchaser endure monetary hardships [23]. According to [52], the need for loans is increasing on a daily basis as people's requirements grow. However, some people default on payments, resulting in some losses.

Loan fraud can happen in many different ways. The most frequent types of loan fraud include fake bank accounts, fake information, account hacking, and customer identity theft. With loan fraud involving a fake bank account, the borrower creates a fake bank account and uses it to apply for the loan. With fake information loan fraud, a borrower uses false information to apply for a loan. With account hacking loan fraud, a fraudster hacks into an existing bank account and applies for a loan through it. With customer identity theft loan fraud, an employee steals a customer's identity to apply for a loan [11]. Despite the outstanding efforts of financial institutions, law enforcement, and the government, fraudsters devise new ways to weaken the possible protective models [54].

Machine learning is one of the biggest subjects this decade, and financial institutions are turning to it for fraud solutions. Companies are increasingly looking to invest in machine learning to improve their solutions. Machine learning combines numerous computer algorithms with statistical modeling to enable the computer to carry out tasks without having to be explicitly programmed.It employs data mining algorithms to uncover hidden patterns in massive amounts of volatile data and to make informed conclusions based on the revealed insights. [51]. Machine learning algorithms such as decision tree, naïve Bayes, logistics regression, k-nearest neighbour, random forest, support vector machine (SVM) and artificial neural networks

(ANN) have been tested, and some have been compared for fraud prediction. According to the results obtained by [39], decision tree, SVM, and ANN show the best results in terms of accuracy. In this study, four machine learning algorithms , namely, random forest, adaptive boosting (Adaboost), extreme gradient boosted machines (Xgboost) and multilayer perceptron (MLP), were proposed and their performances compared.

1.2 Motivation

With the rapid growth of the internet, big data, and easy access to information, it is easy for fraudsters to commit fraud. Fraud has become a big issue, and companies need to stay one step ahead of fraudsters. Traditional fraud prevention algorithms like password protection are not enough [36]. Various algorithms for fraud prediction have been proposed and tested throughout the years. However, it is important to constantly develop fraud prediction algorithms to prevent fraudsters from finding their way around these algorithms . There is a huge amount of research that has been conducted in the area of fraud prediction. However, there is less research on loan fraud, and as a result, the rate of loan fraud is increasing [7]. Hence, there is a need for more research on loan fraud prediction to fill in the gap.

1.3 Problem Statement

Loan fraud is a common occurrence in financial institutions. Nowadays, fraudsters can be creative and intelligent. They take time to build consistent account activity and credit history that appear legitimate to get larger loans and higher credit lines. Due to the large number of applications that financial institutions review daily, it is difficult to manually identify fraud situations. It takes a long time to identify and much longer to reject fraudulent applications, which causes a huge loss of money [43]. In the past, case-based reasoning, analogy-based reasoning, and statistical algorithms have been employed to solve the problem of fraud. However, these algorithms alone are no longer sufficient due to accuracy capabilities limitations and evolving fraudulent strategies [18]. Hence, we want to investigate the effectiveness of machine learning algorithms in predicting loan fraud.

1.4 Research Question

In this project report, we investigate the following research questions:

i) What features are the most accurate predictors of loan fraud?

1.5 Research Aims and Objectives

1.5.1 Research Aims

This research aims to compare machine learning algorithms for the prediction of fraud practices in loan administration so as to find algorithms which present the most accurate results.

1.5.2 Objectives

I) To identify patterns in the dataset using visualisation and statistical methods.

II) To identify imbalanced data technique to increase the overall predictive accuracy;

III) To identify the main significant variables for loan fraud prediction; and

IV) To identify machine learning algorithms that gives the best results.

1.6 Scientific Contribution

This research will add more knowledge to the existing literature. It will help financial companies to verify and supervise every loan transaction and identify fraudulent transactions automatically using the trained machine learning models . Hence, reducing the cost of fraud management.

1.7 Limitations

Loan prediction is a field that has not been explored that much; therefore, there is limited literature. The process of dataset cleaning and manipulation was time-consuming since the dataset had a lot of noise. During the training process, MLP took time to train.

1.8 Overview

The rest of the project report is organised as follows:

Chapter 2 is the literature review were we review other related studies. Statistical and machine learning algorithms for loan and credit card detection are discussed.

Chapter 3 includes a detailed description of the dataset, including data collection, data structure and data pre-processing. It also provides a detailed description of four chosen machine learning algorithms , random forest, Xboost, Adaboost and MLP, and evaluation matrices such as area under the curve, accuracy, recall, precision and f1-score.

Chapter 4 covers discussion of data exploration results, feature creation and feature importance results, and results from the four proposed machine learning algorithms , random forest, Xgboost, Adaboost and MLP.

Chapter 5 covers the summary of the research project as well as the overall conclusion and recommendations.

Chapter 2

Literature Review

2.1 Introduction

Fraud deeds lead to major losses, and as a solution to this problem, researchers are trying to find fraud detection and preventions techniques. Several techniques have been presented and tested. Some of which are discussed below. Given that there is not enough research on the focal problem of this research, we will refer to the literature on credit card fraud.

There are two types of techniques that are usually used for fraud detection or prediction, namely supervised and unsupervised techniques. Supervised techniques use a dataset with records labelled as fraud or no-fraud, while unsupervised techniques use a dataset with unlabelled records [7][8]. Unsupervised techniques usually applied are outlier detection techniques. Several algorithms have previously been used to solve financial fraud; algorithms such as support vector machine (SVM), decision trees, neural networks, random forest, hidden Markov chains, to mention but a few, with SVM being the most used [7].

2.2 Loan fraud

Research on loan fraud prediction using decision tree was done on [18], with 9 features based on feature extraction. Effective data preprocessing was performed before the model was trained. Results obtained and presented by a confusion matrix show that by using decision tree, the false positive rate can be reduced. Cross validation was used to avoid overfitting the model. The same study was carried

out using the kernel support vector machine (KSVM) and naïve Bayes [19][17]. A comparison was done, and KSVM had the highest accuracy with a very low false positive rate and a very high true positive rate. Machine learning algorithms can be used to achieve dependable accuracy on which financial institutions can rely.

In [71], deep neural network (DNN) is used for loan default prediction and is compared with naïve Bayes, decision tree, and k-nearest neighbor (KNN). Feature creation was done by computing the sum , minimum, maximum, average, and variance of some features. The result shows that DNN is suitable for loan default prediction with an accuracy of 0.73. Compared with the other techniques, it improved the f1-score by 25%. However, because the dimension of the input vectors was not very large, adding more hidden layers could not improve the model's performance. Although DNN's accuracy rating was good, its recall scores still need to be improved.

In [73], an algorithm based on knowledge graph and neural network for loan fraud detection was proposed. In this algorithm, a borrower's call history dataset is combined with a loan transaction dataset to build a call network graph. Then, a random walk is completed, and the Word2Vec vector model is used to make the prediction. The accuracy obtained after the model was built is approximately 60%. However, this algorithm is prone to noisy datasets, which affect the prediction performance.

In [69], a study on loan fraud prediction using a hybrid supervised and unsupervised learning model was done. Privacy protection was a concern with the dataset used since it contained a lot of sensitive personal information. To be able to filter away meaningless data while preserving useful information without knowing the meaning of the data, kernel principal analysis (KPCA) and Xgboost algorithms were combined to make a new unsupervised and supervised hybrid model, KPXgboost. A grid search was employed to prevent overfitting. The models, Xgboost and KPXgboost were compared, and the results showed that although both models performed well, the new hybrid model slightly performed better than the Xgboost model. This study gives a new angle on detecting fraud while not compromising customers' privacy. However, training the proposed model is time-consuming.

2.3 Single Models

As of late, random forest and SVM are some of the algorithms that have acquired noticeable quality, with high performance across a scope of applications. In [5] SVM and random forest were used along with logistics regression on a credit card transaction real-life dataset. The results suggest that random forest performed better on average. But the dimensionality of the dataset used has limited its performance, including SVM in credit card fraud.

In [72] two different types of random forest were used to detect credit cards. The basis classifiers used by the two random forests—"random tree-based random forest" and "CART-based random forest"—differ. They did three experiments. The first experiment compared the performance of two random forests on the same subset. The second experiment compares the performance against the percentage of legitimate and fraudulent transactions in a subset. The third experiment shows the performance on a much larger dataset. The results of the first experiment showed that a CART-based random forest performed best. Although the model produced great results on a small dataset in the first experiment, it did not do as well on the second and third experiments on a large dataset because of problems such as imbalanced data.

In [13] supervised machine learning algorithms for detecting credit card fraud were compared. The dataset used consists of 284 807 transactions and 14 featues. Random undersampling was used to balance the data and the top 5 featues were selected for the model training. The performance of the models was assessed based on G-mean, recall, precision, accuracy, true positive, false positive, and specificity. The comparison made in this study was based on xgboost, random forest, logistics regression, SVM, stacked classifiers, gradient boosting, decision tree, MLP, k-nearest neighbor (KNN), and naive bayes. The results obtained suggest that stacked classifiers, random forest and Xgboost, performed better. However, other models could achieve the same performance if featue importance was done for all models.

According to [38] and [29] KNN algorithm is effective in detecting fraud. It has been proven that the algorithm effectively reduces false alarm rates while also increasing

the fraud detection rate. According to the results ofa study by [3], in comparison to logistics regression and naive Bayes, KNN performed best for each metric examined. This is due to the fact that there were no false positives detected by the kNN. The comparison of the three algorithms was based on the Matthews correlation coefficient (MCC), precision, sensitivity, specificity, accuracy, and balanced classification rate (BCR). On a highly unbalanced dataset, a hybrid of undersampling and oversampling is used. The results show that using hybrid sampling on a highly imbalanced dataset considerably improves binary classification performance.

In [10] k-means and genetic algorithms to detect credit card fraud were proposed. The proposed algorithms were tested on a dataset with 1,500 transactions. A genetic algorithm is an efficient optimisation algorithm. The k-means algorithm classifies credit card transactions based on the distinct factor values and creates clusters. However, an increase in the input size results in outliers. In order to solve that problem, a genetic algorithm is applied to the clusters produced by k-means, thereby providing an optimised detection of fraud. Although the proposed algorithm performed well, it was tested on a smaller dataset.

In [4], several classification algorithms trained on public datasets to analyse the correlation between specific attributes and fraud were examined. These algorithms were KNN, SVM, logistic regression, random forest, Naive bayes, and MLP. Initially, training was done on a highly imbalanced dataset and, according to the results obtained, testing on a highly imbalanced dataset results in a high number of false positives, very low precision, and very high recall. Random undersampling was done to reduce dataset imbalances. According to the results obtained after random undersampling, random forest performed best. However, SVM has shown the best performance in credit card fraud detection under real-world conditions. This study was limited by the small set of fraudulent transactions and also features that were encrepted.

In [40], a highly imbalanced dataset was used for fraud detection. Various machine learning algorithms, namely logistics regression, SVM, decision tree, random forest, and stacked classifiers, were used. The Synthetic Minority Oversampling Technique (SMOTE) was used to balance the training dataset. The proposed algorithms were trained on a small dataset and tested on a large and highly skewed

dataset. Although the models were able to classify fraud transactions, they also misclassified genuine transactions as fraud.

Random forest, logistic regression, and decision trees were used in a machine learning credit card fraud detection study. The dataset used was highly imbalanced and to balance the dataset, oversampling was done, which resulted in 40% genuine transactions and 60% fraud transactions. The experiment was first done on 5 features, then 10, and finally all features. The accuracy of the algorithms increased with an increasing number of features. This shows that performance is also affected by the number of features used. In this study, it was said that "accuracy, error rate, sensitivity, and specificity are used to evaluate different variables for three algorithms." However, only accuracy results were given.

In [46], a comparision of supervised and unsupervised algorithms was done. Six supervised algorithms, namely,KNN, SVM, Xgboost, decision tree, logistics regression, and random forest and four unsupervised algorithms, namely, auto encoder, constrained Boltzmann machine, one-class SVM, and generative adversarial networks, were assessed. To evaluate the performance, 5-fold cross-validation in terms of the Area Under the Receiver Operating Curve (AUROC) was used. Xgboost performed best with an AUROC of 0.99 within the supervised algorithms, and for the unsupervised algorithms, restricted Boltzmann machine performed best with an AUROC of 0.96. The results suggest that supervised models performed better than unsupervised models. The performance of supervised learning is greatly limited by label dataset availability and dataset imbalance, whereas unsupervised learning is not limited by these issues.

In [33] several machine learning algorithms were implemented and compared for insider fraud detection occurring in banks. Machine learning algorithms used are artificial neural networks (ANN), gradient boosting, logistics regression, deep learning autoencoder network, and random forest. The models were fitted yearly, from 2007-2019, using observed cases of fraud and financial data from four quarters. The models are then used to predict fraud in the banking sector for the following year, which is 2020. Each model performed best in 6 out of 12 years of training. To improve the performance of the models, dimensional reduction and standardisation were done. Additionally, unnecessary correlated variables were removed. The

result shows that ANN, gradient boosting, and random forest performed best. The overall conclusion we can draw from this study is that the quality of the dataset affects the performance of the models. Hence, steps such as removing correlated variables,data standardisation, and dimensional reduction can potentially enhance the performance of the models.

In [49] to detect credit card fraud, they used an auto-encoder and a restricted Boltzmann machine. Three datasets were used for this study. Keras was used as a high-level neural network API. Of the three datasets, auto-encoder and restricted Boltzmann machine produced the highest AUC score and accuracy on the larger dataset. Based on the results of this study, we can conclude that the proposed algorithms are suitable for larger datasets.

In [41], MLP was applied with a Chebyshev functional link artificial neural network (CFLANN) and decision tree for credit card fraud over two datasets. The three algorithms were compared in terms of the amount of time it took to detect credit card fraud and their accuracy score. In terms of time, decision tree was the quickest, followed by CFANN. In terms of accuracy, MLP performed best in both datasets. From this study, we can conclude that the simplicity of a model does not mean it is the most accurate.

In [25] three algorithms, logistics regression,kNN, and naive Bayes, were used for fraud detection. Three different proportions of datasets were used, 50:50,34:66, and 25:75. Random undersampling was used for unbalanced dataset. The performance was measured using f-measure, AUC, sensitivity, specificity, accuracy, and precision. When comparing the three, it was discovered that logistic regression performed the best. Better results can be obtained by using random undersampling techniques before training the prediction model.

In [53], SVM was proposed for fraud prediction, and random forest was proposed for feature selection. Random forest was chosen for feature selection due to its suitability for a large dataset. Accuracy, AUC, and recall were proposed as metrics to evaluate the performance. The results show that SVM based on a random forest model gives good accuracy and decreases the false-positive transaction rate by increasing the sensitivity rate.

In [55], logistics regression was compared with ANN using a highly imbalanced dataset. The result suggests that ANN performs better than logistics regression. The logistics regression overfitted the training dataset as it increased, and as the accuracy increased, the number of frauds detected in the test dataset decreased due to the biased character of the class distribution of the training dataset.

2.4 Statistical Models

In a study done by [31], a two-sequence alignment technique that includes misuse identification and anomaly detection was proposed. Their technique involves a profile analyser to identify whether an incoming sequence of transactions on a certain credit card is similar to the prior spending sequence of the authentic cardholder. The deviation analyser then examines the unusual transactions that the profile analyser identified for possible alignment with prior fraudulent behavior. On the basis of the observations made by the two analyzers, the final decision regarding the nature of a transaction is made. This technique, however, is unable to identify fraud in real time.

A detailed survey on the performance of hidden markov model (HMM) for detecting credit card fraud was presented by [60]. HMM is used to identify the cardholder's spending profile by clustering the training set. Based on how much the cardholder spends, the spending profile is divided into low, medium, and high categories. A set of probabilities is allocated for the total number of transactions for each cardholder, and the total number of transactions is coordinated with the cardholder's category . If the transaction falls within the assigned threshold, then the transaction is legitimate; otherwise, it is fraudulent.

In [61], an HMM was used to model patterns of transactions for credit card processing operations and fraud detection. As a starting point, they performed multiple experiments to identify the ideal combination of HMM parameters, such as the number of states, the length of the sequence, and the threshold value. Then, an HMM was trained using the normal behavior of a cardholder. If the trained HMM rejects the incoming transaction, it is classified as a fraudulent transaction. A comparative study with another fraud detection system was then done, and the results

suggest that the proposed technique did better, and it is also suitable for a large dataset.

Fraud was detected using a Multiple Semi-Hidden Markov model (MSHMM), and parameters were optimised using the Cuckoo search algorithm [48]. The Cuckoo Search algorithm decides the number of states and its model parameters. The main goal of this study was to free clients from the necessity of statistical knowledge by automating the use of the MSHMM. Precision, recall, and f-measure were used to validate the effectiveness of the existing MSHMM and the planned optimsed MSHMM. The results imply that the proposed technique is highly accurate. Optimisation is crucial to increase detection accuracy.

In [12] time series analysis for credit card detection using data with parameters amount and time of the transactions was done. Algorithms for detection at two different levels are employed. Level one fraud detection uses a distance-based algorithm to determine whether an incoming transaction is fraudulent or not .At level two, the label prediction technique is used to identify the next transaction, which is then compared with the actual transaction.If there is a difference, the transaction is fraudulent. By examining a cardholder's transaction history, anomalies in a transaction are found. If a transaction is considered to be fraudulent, the cardholder is requested to continue the transaction by asking a secret question; if the cardholder fails to respond the transaction will be cancelled. The approach used reduces the false-positive rate and ensures that genuine transactions are not rejected.

2.5 Hybrid Models

In [21], a probabilistic-based algorithm was used for detecting fraudulent transactions. The model used was optimised with Baum-Welsh and hybrid posterior-Viterbi algorithms. The model was evaluated using true positive rate, false positive rate, accuracy, precision, and receiver operating characteristics (ROC) curve . The dataset used was of four card holders, which is a very small set. The results obtained show that optimisation of parameters helped with improving the performance of the model, although it still commits some false positive errors.

In [14], boosting algorithms were used to evaluate the performance of credit card fraud transactions. Initially, machine learning algorithms such as naïve Bayes, decision tree ,random forest, and logistic regression were used on the employed dataset to find the algorithm with the least accuracy. It is then chosen as the weak learner, which in this case is decision tree. Boosting algorithms such as Adaboost, Xgboost, and gradient boost are then used to create hybrid models. The result shows that Xgboost is the most precise and accurate in predicting fraud. Based on findings of this study, we can say that boosting algorithms do help weak learners do better.

In [22], K-means clustering with MLP was applied as a hybrid model along with K-means with Hidden Markov model (HMM). K-means clustering was used to group credit card transactions suspected to be fraudulent and those that were not fraudulent because the dataset used for the study did not include the target class. Three experiments were conducted,the first experiment with a 66% split and the second and third with an 80% split. The results obtained showed that MLP with k-means clustering using an 80% split performed better, with an overall classification accuracy of 51.5%. The accuracy of the model is not as good, but promising results can be obtained if some data cleaning techniques are employed.

2.6 Summary

The abovementioned researces show that SVM, MLP, decision tree,and random forests are the most commonly used machine learning techniques for fraud detection. Hybrid techniques often outperform single algorithm techniques. One of the difficult aspects that the studies tried to address is the imbalance in the fraud detection dataset. Because of the ratio of genuine transactions to fraudulent transactions, trained machine learning algorithms may result in misclassification.In most studies, feature selection has been shown to be a useful way to improve how well machine learning algorithms work.

Chapter 3

Research Methodology

3.1 Introduction

This chapter includes a detailed description of the dataset, including data collection, data structure, and data preprocessing. It also provides a detailed description of four chosen algorithms, namely, Random forest,Adaboost, Xgboost, and MLP, as well as evaluation matrices, namely, accuracy, recall, precision, f1-score, Mattews correlation coefficient(MCC), Cohen's kappa, area under the curve(AUC), and Kolmogorov Smirnov statistics (KS) charts.

3.2 Research design

This research project aimed to compare four machine learning algorithms, namely, random forest, Adaboost, Xgboost, and MLP, to predict fraud practices in loan administration to find techniques that present the most accurate results.

The research project was carried out as follows: Exploratory data analysis was performed in order to discover trends, anomalies, patterns, or relationships within the dataset. Label encoding was used to convert each categorical label into distinct numerical values. The most important features were selected using random forest. The dataset was then split into 80% training set and 20% testing set. The four models, random forest, Adaboost, Xgboost and MLP, were trained before applying any imbalanced data sampling technique to see how the models would perform. To improve the performance of these models, random undersampling was done. The models were trained again to get the final results.

3.3 Research Method

This section describes the four algorithms, mentioned above, that will be used to carry out the research. It includes a more detailed discussion of the four techniques.

3.3.1 Random Forest

A random forest is made up of many independent decision trees that work together as an ensemble throughout the training process [68]. To create each tree in the forest, a small collection of input features is randomly selected at each node to be split on, and the optimal split is determined using features from the training set. It combines the results of each decision tree to generate the output. The tree is generated using the CART (classification and regression tree) algorithm without pruning in order to optimize size. CART is a machine learning technique for creating prediction models from supplied data. The models are created by fitting a prediction model within each division of the data space, which is done recursively. Hence, the partitioning can be represented as a decision tree [68]. The subspace randomisation schemes are combined with "bagging, to resample the training data set each time another individual tree is grown, with replacement" [6]. Bagging is a technique used to reduce variance in algorithms with large variance, such as decision trees. It has been proven that the random forest algorithm is resistant to overfitting and offers a good estimation of the generalisation error [58]. Random forest technique can also handle large amounts of data with hundreds of different variables. The technique is appropriate for challenging projects since it quickly handles variables [64].

Random forest was also used for feature selection. Feature selection using random forest falls under the embedded techniques category. Embedded techniques combine the capabilities of wrapper and filter and are employed by techniques that have their own built-in feature selection methods. Embedded techniques have the following advantages, they are highly accurate, they have better generalisation, and they are interpretable [15]. Figure 3.1 below shows an overview of a Random forest:

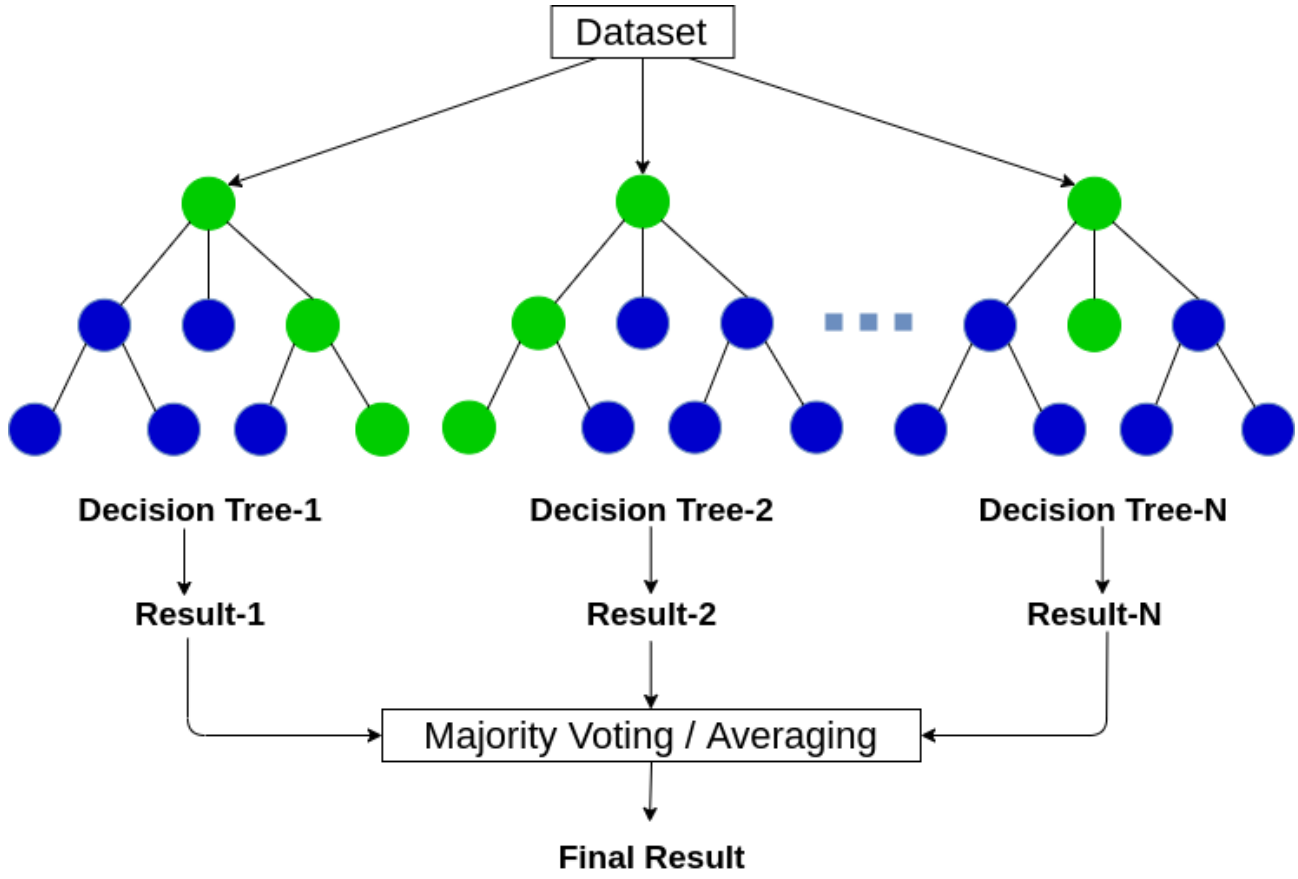


FIGURE 3.1: An overview of a Random forest. Adapted from [67].

Random Forest Algorithm

Figure 3.2 below shows a Random forest algorithm :

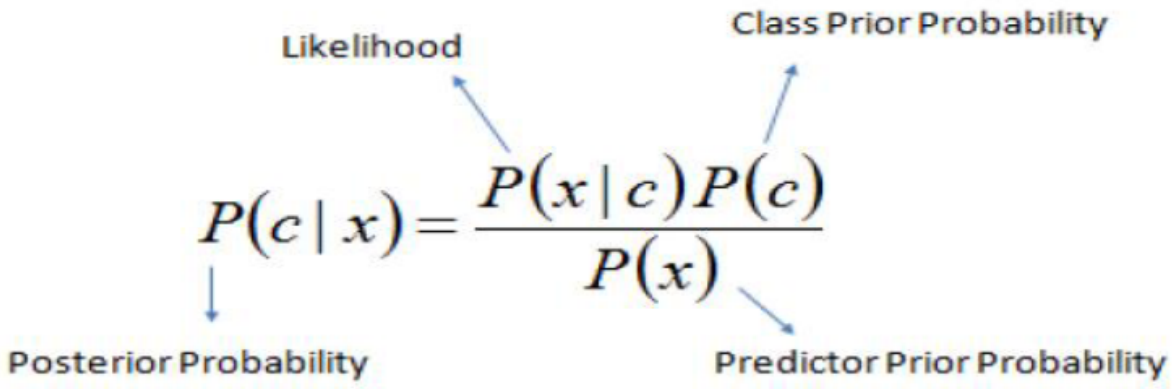


FIGURE 3.2: Random forest algorithm. Adapted from [58].

3.3.2 Extreme Gradient Boosting

The basis of Xgboost is gradient boosting. It is applicable to supervised learning problems such as regression, classification, and ranking. Gradient boosting is a regression and classification algorithm. It produces a model in the form of an ensemble, typically a decision trees. To give better performance, Xgboost uses a regularised model formalisation to control overfitting [45]. When compared to other algorithms, Xgboost is an effective and simple with good performance and accuracy. It uses the power of parallel processing. It uses multiple central processing units cores to implement the model. Unlike many algorithms, it can handle missing values. It performs tree pruning effectively, that is, it makes splits to the maximum specified depth and then starts pruning the tree backwards [30].

Figure 3.3 below shows gradient boosting algorithm:

Input: training set $\{(x_i, y_i)\}_{i=1}^{n}$, a differentiable loss function $L(y, F(x))$, number of iterations M.

Algorithm:

1. Initialize model with a constant value:

$$F_0(x) = \arg\min_{\gamma} \sum_{i=1}^{n} L(y_i, \gamma).$$

2. For m = 1 to M:
   1. Compute so-called *pseudo-residuals*:

$$r_{im} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F(x)=F_{m-1}(x)} \quad \text{for } i = 1, \ldots, n.$$

   2. Fit a base learner (e.g. tree) $h_m(x)$ to pseudo-residuals, i.e. train it using the training set $\{(x_i, r_{im})\}_{i=1}^{n}$.
   3. Compute multiplier γ_m by solving the following one-dimensional optimization problem:

$$\gamma_m = \arg\min_{\gamma} \sum_{i=1}^{n} L\left(y_i, F_{m-1}(x_i) + \gamma h_m(x_i)\right).$$

   4. Update the model:

$$F_m(x) = F_{m-1}(x) + \gamma_m h_m(x).$$

3. Output $F_M(x)$.

FIGURE 3.3: Gradient Boosting Algorithm. Adapted from [56].

3.3.3 Adaptive Boosting

Another ensemble algorithm is Adaboost, which is similar to random forest. It aims to combine weak algorithms to build one strong algorithm. The weak learners in Adaboost are decision trees, logistics regression, etc [32]. A single algorithm might do a poor job of classifying the elements. If we combine several algorithms, choose a training set for each iteration, and provide the proper amount of weight to the final vote, we can improve the overall accuracy. Adaboost keeps the algorithm iteratively by selecting the training set depending on prior training accuracy. The accuracy attained determines the weighting of each trained model at any iteration. Adaboost provides weight to every training item after a classifier has been trained at any level. The misclassified item is given a higher weight so that it has a greater probability when it appears in the training subset of the following model. Following the training of each model, the weight is assigned to the model based on accuracy. In order for a more accurate model to have a greater impact on the outcome, it is given a higher weight because the input parameters are not jointly

optimised. By applying Adaboost, weak models' accuracy can be increased [58].

Figure 3.4 below shows Adaboost algorithm:

Given: $(x_1, y_1), \ldots, (x_m, y_m)$ where $x_i \in \mathscr{X}$, $y_i \in \{-1, +1\}$.
Initialize: $D_1(i) = 1/m$ for $i = 1, \ldots, m$.
For $t = 1, \ldots, T$:

- Train weak learner using distribution D_t.
- Get weak hypothesis $h_t : \mathscr{X} \to \{-1, +1\}$.
- Aim: select h_t with low weighted error:

$$\varepsilon_t = \Pr_{i \sim D_t} [h_t(x_i) \neq y_i].$$

- Choose $\alpha_t = \frac{1}{2} \ln \left(\frac{1 - \varepsilon_t}{\varepsilon_t} \right)$.
- Update, for $i = 1, \ldots, m$:

$$D_{t+1}(i) = \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

where Z_t is a normalization factor (chosen so that D_{t+1} will be a distribution).

Output the final hypothesis:

$$H(x) = \operatorname{sign} \left(\sum_{t=1}^{T} \alpha_t h_t(x) \right).$$

FIGURE 3.4: Adaboost Algorithm. Adapted from [63].

3.3.4 Multilayer Perceptron

MPL commonly known as multilayer feed-forward neural networks, is the most frequently used neural network for a variety of problems. Its basis is supervised learning . MLP consists of three important layers, the input layer, hidden layer, and output layer, with neurons and processing units. Information flows from the input layer through the hidden layer to the output layer [47]. The input layer is the first layer that receives the input information. The hidden layers are in the middle. The hidden layers perform mathematical computation on the input data. The output layer gives the results that are obtained through computations performed by the middle layer [37]. Figure 3.5 below shows an overview of a MLP:

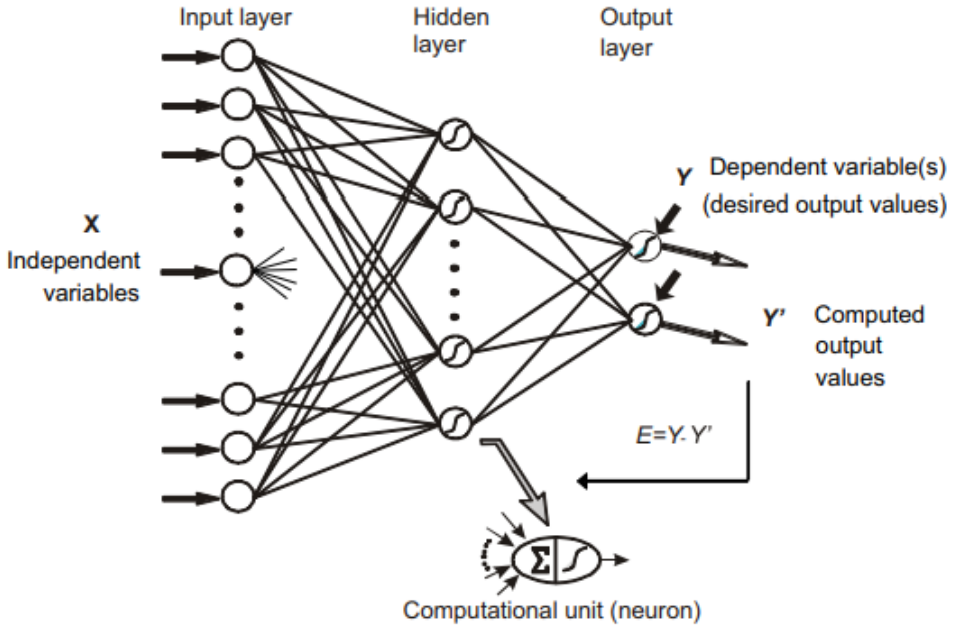


FIGURE 3.5: Three-layered feed-forward neural network.
Adapted from [37].

Neurons are the building blocks of neural networks. Neurons are computational units that have weighted input signals and use an activation function to produce an output signal [9]. Each neuron has a bias, which refers to an input that is weighted and has a constant value of 1.0. Most weights are initially set to small random values, like those between 0 and 0.3. The weighted inputs are added together and run through an activation function [9]. In simple terms, an activation function is one that assists the network in learning complex patterns in data. It receives the output signal from the preceding neuron as input and transforms it such that the next neuron can accept it as input. The ability of an activation function to introduce non-linearity into a neural network is its most important feature [26].

An overview of the strengths and limitations of the techniques discussed above is provided in Table 3.1 [44] [2]:

TABLE 3.1: Comparison Of Chosen Techniques

Model	Strengths	Limitations
Random Forest	Less affected by noise and generalises better reducing variance	More hyperparameter tuning is required
AdaBoost	Less prone to overfitting,easy to understand and visualise	Prone to noisy dataset
Xgboost	It is fast and generalises better reducing variance	Difficult to understand, visualise and tune
MLP	Works well with large dataset	Difficult computation and time cosuming

3.4 Data and data pre-processiong

The dataset is the Kaggle fraud detection dataset for the year 2019. This dataset contains two files, the application dataset with 122 features and 307511 loan applications and the previous application dataset with 37 features and 1670214 loan applications. Features include demographic features such as gender and age, geographic features such as the client's living area, and transactional features such as credit amount.

The dataset contains 24825 applications that are classified as fraudulent, which is 8.1% of the whole dataset. Looking at the percentage, it is clear that the dataset is highly imbalanced. Since the distribution ratio of classes plays a considerable role in the model's accuracy, data preprocessing is essential.

Good data preparation allows for practical analysis to reduce errors and inaccuracies that can occur during the model training stage. Data preparation is the process of cleaning and transforming raw data. It is a critical stage before processing data. It is good to understand the data we are working with and to gather insights about the data before analysing and processing it. The process of doing this is called exploratory data analysis (EDA). [16] defines EDA as a process to analyse and summarise the main characteristics of a dataset, usually using data visualisation techniques. It helps with determining how to manipulate the data best. For this research, EDA is done with the help of data visualisation tools such as summary statistics and graphs.

Data cleaning is the process of ensuring that the data is correct, usable, and consistent. Filling in or removing missing values is the most significant step in the data cleaning process. Another important step is checking for outliers. Outliers are extreme values that fall feather way from other observations.

Feature selection is a crucial step in data preprocessing. Feature selection is a technique of choosing variables that are most important in the given dataset, in which case random forest was used. Selecting the most relevant features can reduce the chances of overfitting, improve accuracy, and reduce training time.

Since machine learning algorithms find it difficult to learn when classes are not balanced and, considering the given dataset is highly imbalanced, a random undersampling technique was used. Random undersampling removes samples from the training dataset that belong to the majority class to balance the class distribution. This approach is different from oversampling, which includes adding samples to the minority class. [9].

3.5 Performance Evaluation Metrics

In machine learning, performance measure is an essential step. To measure the performance of our models, evaluation metrics such as accuracy, recall, precision, F1-score, Mattews correlation coefficient(MCC), and Cohen's kappa will be used. Additionally, we will use AUC-ROC and Kolmogorov Smirnov statistics (KS) charts.

Accuracy is the ratio of correct predictions to total observations. Precision talks about how accurate the model is in predicting positive values. Recall is the number of actual positives that the model captures by labeling them as positive. F1 score is the function of precision and recall [59].Accuracy, precision, recall, and F1 score are calculated using the following formulas:

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{3.1}$$

$$Precision = \frac{TP}{TP + FP} \tag{3.2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3.3}$$

$$F1 = 2 \times \frac{precision \times recall}{precision + recall} \tag{3.4}$$

Where

TP is true positive;
TN is true ngative;
FP is false positive; and
FN is false negative.

The MCC in machine learning is used to measure the quality of binary classifications. It takes into consideration true negatives, true positives, false negatives, and false positives. Statistics show that MCC yields high scores if and only if the prediction returns good rates for these four categories [65]. It is also considered a balanced measure which can be used regardless of whether the classes are of altogether different size [35].MCC lies between -1 and 1, with 1 being the "best agreement between actual values and predicted values and 0 being no agreement between actual values and predicted values" [65]. The formula to calculate mcc is given as follows:

$$MCC = \frac{TN \times TP - FN \times FP}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{3.5}$$

Cohen's kappa is a measurement used to assess the performance of a classification model. It is computed based on the confusion matrix. However, rather than computing the overall accuracy, it takes into account imbalanced classes [70]. Cohen's Kappa is calculated using the following formula:

$$K = \frac{P_0 - P_e}{1 - P_e} \tag{3.6}$$

Where
P_0 is the overall accuracy of the model; and
P_e is the measure of agreement between the model predictions and the actual values.

"Cohen's Kappa is always less than or equal to 1".Value less than zero means the classifier is not useful [27].

KS statistics measures the performance of classification models. It is a measure of the degree of separation between the positive and negative distributions. The K-S is 100 if the scores partition the population into two separate groups, in which one group contains all the positives and the other all the negatives. The K-S is likely to fall between 0 and 100 in most classification models. The higher the value, the better the model is at separating the positive from negative cases [62].

AUC-ROC is a performance measure for classification problems. ROC (receiver operating characteristic curve) is a probability curve, and AUC (area under the ROC curve) is a measure of separability. "The AUC tells us how much the model can distinguish between classes". A higher AUC implies that the model is better at predicting classes [42]. "AUC lies between 0 and 1. AUC of 1 means the classifier is perfect, AUC of 0.5 means the classifier is random in its prediction, and if AUC lies between 0.6 and 0.9, then the classifier is considered a good classifier". Below is area under the curve for ROC curves [28].

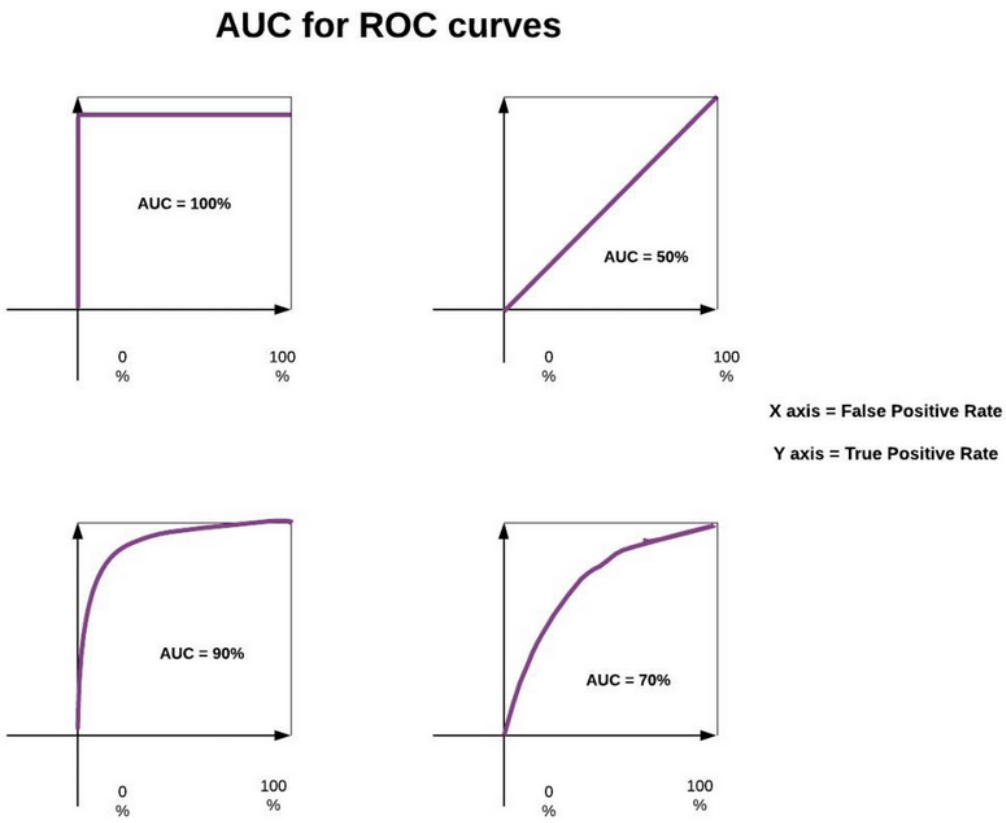


FIGURE 3.6: AUC for ROC curves. Adapted from [28].

3.6 Systems Specifications

3.6.1 Hardware Specification

The specification of computer systems used to implement proposed models is as follows:

- Device Name: Lenovo Thinkpad Laptop (Core i7 9th Gen)
- Processor: Intel Core i7-9750H
- RAM: 16GB
- System Type: 16-bit operating system

3.6.2 Software Specification

The software specification is as follows:

- Programming language: python
- visualisation and exploration of the data: Jupyter notebook
- To create figures: Matplotlib
- Operating system: Windows 10

3.7 Summary

This chapter discussed a detailed description of the dataset, including data collection, data structure, and data preprocessing. It also provides a detailed description of the four chosen algorithms, random forest, Xgboost, Adaboost, and MLP; and evaluation matrices used; accuracy, precision, recall, f1-score, Mattews correlation coefficient(MCC), and Cohen's kappa will be used. The AUC-ROC and Kolmogorov Smirnov statistics (KS) charts are also included.

Chapter 4

Results and Discussion

4.1 Introduction

In this chapter, we will discuss the experimental results we attained. First, we will discuss the data exploration results, data preparation. Then we will discuss feature importance. The most important features are selected. Finally, we will discuss the results from the four implemented machine learning models, random forest, XGboost, Adaboost, and MLP.

4.2 Exploratory Data Analysis(EDA)

One of the most challenging steps we had to do was data preparation since our data had many features. Trying to understand those features was also challenging. Initial analysis of the data in order to find trends, anomalies, patterns, or relationships within the data using summary statistics and graphical representations was done. The process of doing this is called exploratory data analysis (EDA). It is used to learn what our data can tell us with the help of summary statistics and graphical representations. It is always a good practice to start by trying to understand what the data tells us and trying to find as many insights as possible from it.

4.2.1 Data size and structure

The original dataset comprises of two csv files, which contain the application data with 307511 observations and 122 features and the previous application data with

1670214 observations and 37 features. We have one target variable, with 1 representing clients who have not paid their loan installments for more than X days and 0 representing all the other cases. To do the analysis, we used the application data csv file. 5.3 contains a detailed description of the dataset.

4.2.2 Categorical analysis

We looked at the distribution of the class variable as shown in the plot in Figure 4.1 below. The percentage of people who did not pay their loan is 8.1% and the percentage of people who have paid their loan is 91.9% as shown in the pie plot below. This is an extremely imbalanced dataset. Loans that were repaid outnumber loans that were not repaid. This, of course, we had to account for when splitting the data between the training and the testing data.

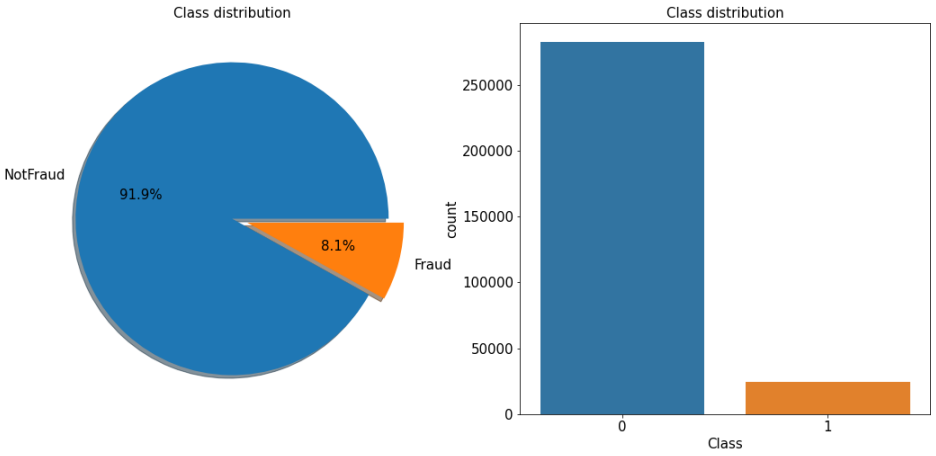


FIGURE 4.1: Distribution of the Class variable

We look at the distribution of the class variable with respect to contract type. From the plot in Figure 4.2, it can be seen that the percentage of cash loans is extremely higher than the number of revolving loans for both defaulters and non-defaulters.

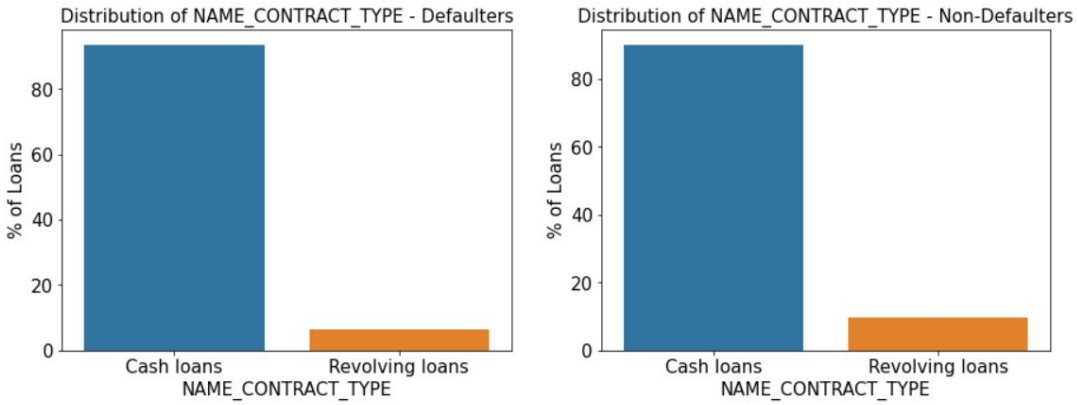


FIGURE 4.2: Distribution of the Class variable with contract type

We look at the distribution of the class variable with gender. From the plot in Figure 4.3, it can be seen that the percentage of females taking loans is higher than the number of males for both defaulters and non-defaulters.

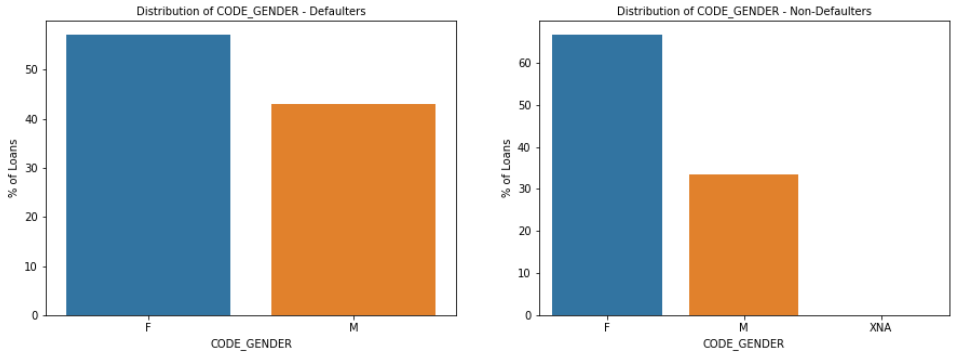


FIGURE 4.3: Distribution of the Class variable with gender

We look at the distribution of the class variable with occupation type. As shown in the plot in Figure 4.4, applicants whose occupation types are laborers or sales staff seem to be the highest both in the defaulter and non-defaulter categories.

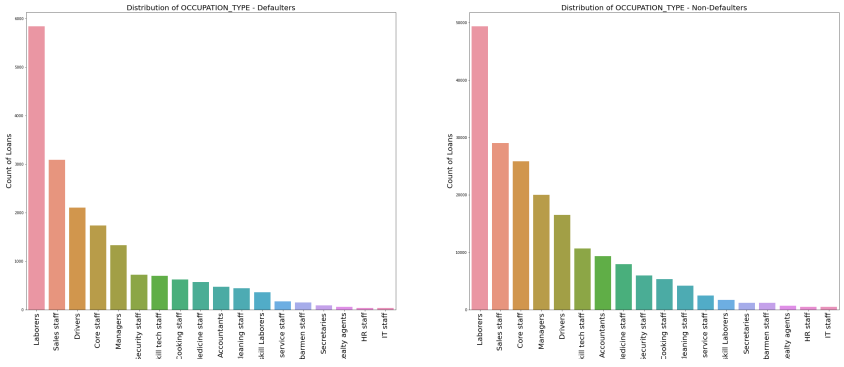


FIGURE 4.4: Distribution of the Class variable with occupation type

Figure 4.5 shows the distribution of the income type and class variable. There are four income types with no payment defaults, state servant, pensioner, commercial associate, and worker. Among all professions, working professionals have the highest number of applications with no payment defaults.

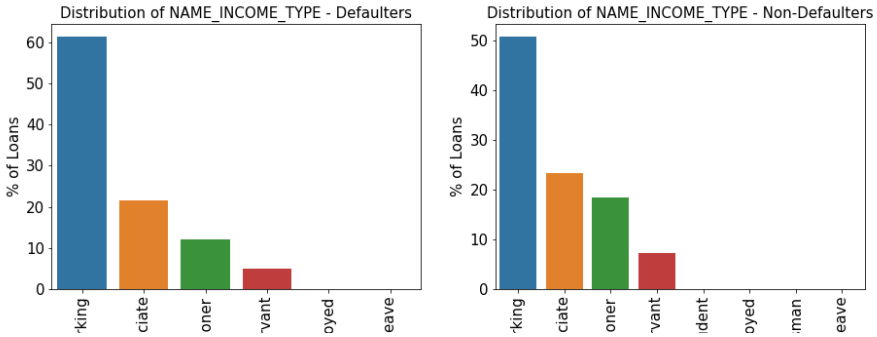


FIGURE 4.5: Distribution of the Class variable with the type of income

From the plot in Figure 4.6, it can be seen that applicants with low total income were the ones who defaulted the most as compared to those with high income.

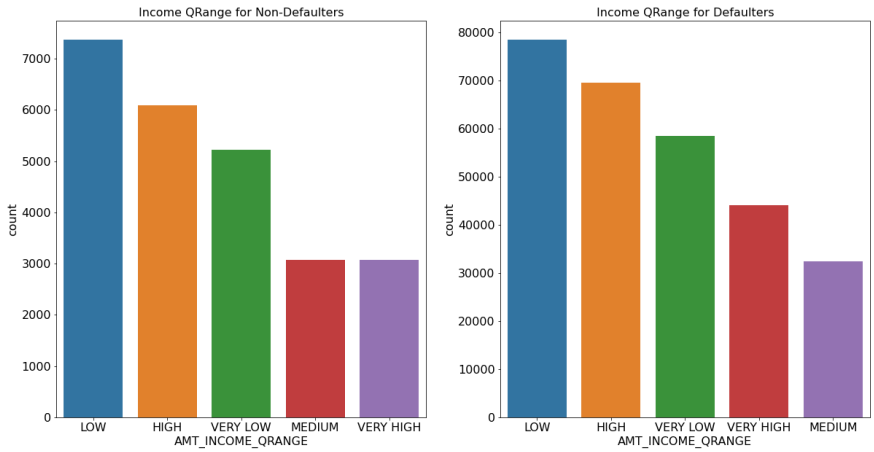


FIGURE 4.6: Distribution of the Class variable with income level

The plot in Figure 4.7 below shows the distribution of age with class. The risk of defaulters was higher in the younger age group as compared to the older group. As age increased, the risk decreased.

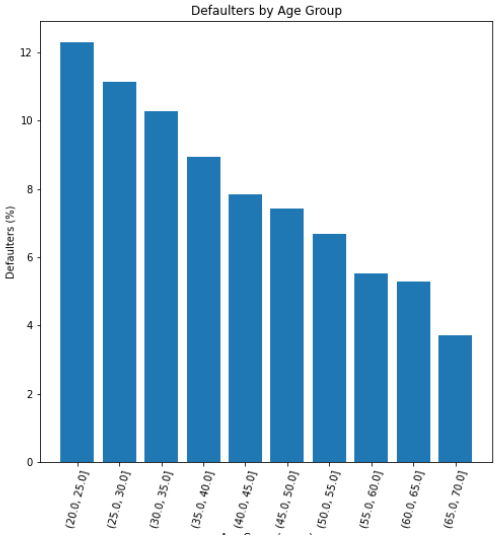


FIGURE 4.7: Distribution of the Class variable with age

4.2.3 Bivariate analysis

The plot in Figure 4.8 shows the bivariate analysis of total income and credit amount. Maximum credit amount applications are for a total income range between 0 to 2500, and applicants with higher total income levels have fewer loan applications. Hence, the total income range between 0 to 2500 has the most loan applications and defaults the most.

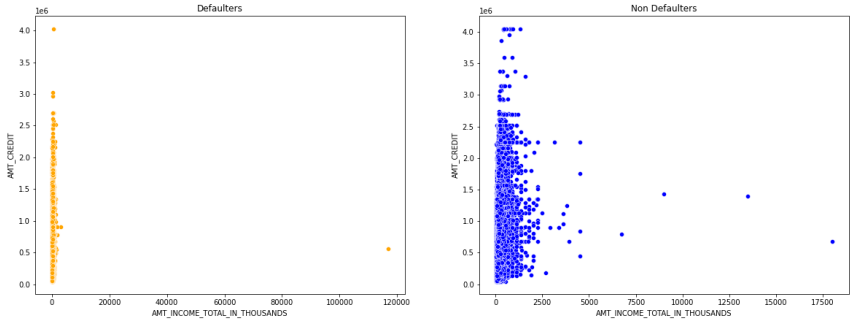


FIGURE 4.8: Bivariate analysis on total income and credit amount

4.2.4 Missing values

The original dataset had quite a lot of missing values. To understand it from a more numerical viewpoint, we decided to count missing value entries, calculate the percentage for each feature, and also sort them in descending order. Table 4.1 below shows that the commonarea-mode, commonarea-medi ,commonarea-avg, nonlivingapartments-mode and, nonlivingapartments-med have the highest percentage of missing values. Linear interpolation was used to impute missing values. It is an imputation technique that assumes a linear relationship between data points and uses non-missing values from data points that are close to compute a value for a missing data point. [1].

	Total	Percentage
COMMONAREA_MODE	214865	69.872297
COMMONAREA_MEDI	214865	69.872297
COMMONAREA_AVG	214865	69.872297
NONLIVINGAPARTMENTS_MODE	213514	69.432963
NONLIVINGAPARTMENTS_MEDI	213514	69.432963
NONLIVINGAPARTMENTS_AVG	213514	69.432963
LIVINGAPARTMENTS_MODE	210199	68.354953
LIVINGAPARTMENTS_MEDI	210199	68.354953
LIVINGAPARTMENTS_AVG	210199	68.354953
FLOORSMIN_MEDI	208642	67.848630
FLOORSMIN_AVG	208642	67.848630
FLOORSMIN_MODE	208642	67.848630
YEARS_BUILD_AVG	204488	66.497784
YEARS_BUILD_MEDI	204488	66.497784
YEARS_BUILD_MODE	204488	66.497784
OWN_CAR_AGE	202929	65.990810
LANDAREA_AVG	182590	59.376738
LANDAREA_MODE	182590	59.376738
LANDAREA_MEDI	182590	59.376738
BASEMENTAREA_MEDI	179943	58.515956
BASEMENTAREA_AVG	179943	58.515956
BASEMENTAREA_MODE	179943	58.515956
EXT_SOURCE_1	173378	56.381073
NONLIVINGAREA_MEDI	169682	55.179164
NONLIVINGAREA_MODE	169682	55.179164
NONLIVINGAREA_AVG	169682	55.179164
ELEVATORS_AVG	163891	53.295980

TABLE 4.1: Percentage of null values

Table 4.2 below shows categorical variables missing values. For categorical variables we impute missing values with mode.

4.2.5 Correlation between variables

To measure the relationship between variables, we used the Pearson correlation coefficient. The Pearson correlation measures the strength and direction of a linear

	Total	Percentage
FONDKAPREMONT_MODE	210295	68.386172
WALLSMATERIAL_MODE	156341	50.840783
HOUSETYPE_MODE	154297	50.176091
EMERGENCYSTATE_MODE	145755	47.398304
OCCUPATION_TYPE	96391	31.345545
NAME_TYPE_SUITE	1292	0.420148

TABLE 4.2: Percentage of categorical variables null values

relationship between two numeric variables. A positive correlation between two variables indicates that the variables move in the same direction, and a negative correlation between two variables indicates that the two variables move in the opposite direction. Correlations lie within the interval of -1 and 1.

Figure 4.11 below is a correlation heatmap for defaulters:

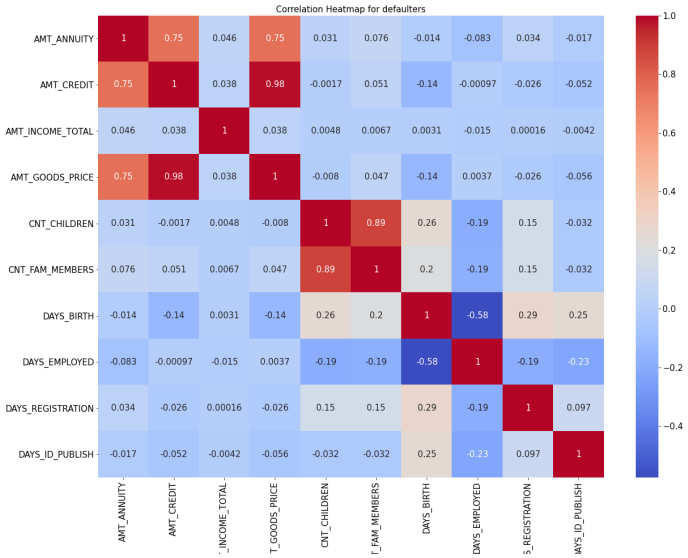


FIGURE 4.9: Defaulters heatmap

We can see a positive correlation for the following variables in descending order :

- amt-credit and amt-goods-price - highest correlated
- cnt-children and cnt-fam-members

- amt-goods-price and amt-annuity

We can see a negative correlation for the following variables in descending order :

- days-birth and amt-income-total
- days-employed and amt-goods-price

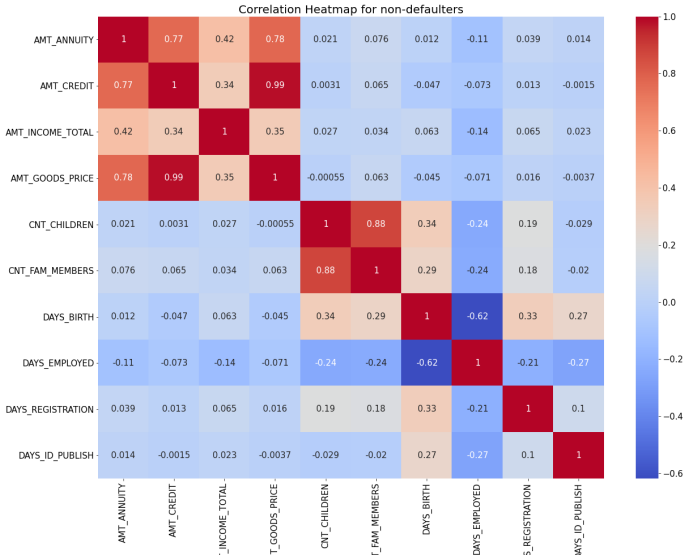


FIGURE 4.10: Non-defaulters heatmap

Looking at the correlation for non-defaulters in Figure 4.12,
We can see a positive correlation for the following variables in descending order:

- amt-credit and amt-goods-price - highest correlated
- cnt-children and cnt-fam-members
- amt-goods-price and amt-annuity
- amt-credit and amt-annuity
- days-employed and days-birth

We can see a negative correlation for the following variables in descending order :

- days-birth and amt-annuity
- days-birth and amt-income-total
- days-employed and amt-goods-price

To avoid collinearity we drop variables that are highly correlated.

4.3 Feature engineering

Feature engineering consists of creation, transformation, and selection of features.

Feature transformation is the process of manipulating features to improve the performance of the model, for example ensuring that features are in the same scale. Feature selection is the process of selecting features that contribute the most to the prediction of the target variable. It can be done either automatically or manually. Feature selection helps with identifying and removing irrelevant features that may reduce the accuracy of our model. It also helps with reducing the dimension of the dataset, thereby reducing the complexity of the model.

4.3.1 Feature selection

Label encoding was used to convert each categorical label into distinct numerical values. Label encoding can be achieved using Sklearn's effective tool for encoding levels of categorical features into numeric values. Label Encoder encodes categorical labels with a value between 0 and n-1, where n is the number of distinct labels [57]. Tables 4.3 and 4.4 show the output of label encoding.

For feature selection, we applied random forest algorithm to the training dataset to obtain feature importance estimates for each feature. Initially, we had 26 features, and the 14 most important features were selected. We then ordered these features according to their importance, as shown in Figure 4.13 below:

From Figure 4.13, it is clear that *ext-source-2* is significantly the most important feature followed by *ext-source-3*.

NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY
0	1	0	1
0	0	0	0
1	1	1	1
0	0	0	1
0	1	0	1

TABLE 4.3: Output of Label encoding

NAME_INCOME_TYPE	NAME_EDUCATION_TYPE	NAME_FAMILY_STATUS	NAME_HOUSING_TYPE
7	4	3	1
4	1	1	1
7	4	3	1
7	4	0	1
7	4	3	1

TABLE 4.4: Output of Label encoding

To train our model and balance model complicity and information loss, the 14 most important features were selected. Table 4.5 below shows the 14 selected features ordered by their feature scores.

TABLE 4.5: Top 14 selected features

Feature Name	Feature Scores
EXT-SOURCE-2	0.106803
EXT-SOURCE-3	0.100188
EXT-SOURCE-1	0.085366
DAYS-BIRTH	0.036272
DAYS-ID-PUBLISH	0.035900
DAYS-REGISTRATION	0.035246
DAYS-EMPLOYED	0.034281
SK-ID-CURR	0.034076
INGAPARTMENTS-AVG	0.031682
AMT-CREDIT	0.031626
COMMONAREA-AVG	0.031237
AST-PHONE-CHANGE	0.031166
LANDAREA-AVG	0.031216
YEARS-BUILD-AVG	0.031141

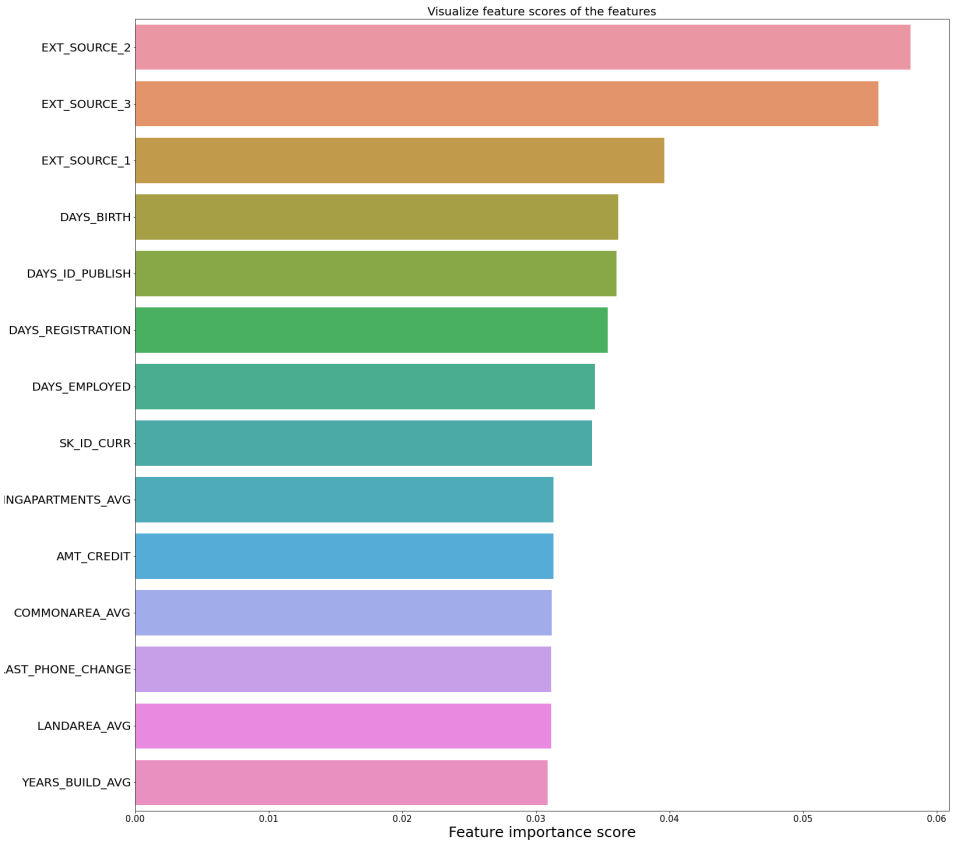


FIGURE 4.11: Overview of feature importance

4.3.2 Feature transformation

Standardisation is a scaling technique where the values are centred around a mean with one standard deviation. Standardisation is used when the dataset has varying scales. It can improve the performance of a model. Standardisation was applied to features after data splitting to improve the performance of our model. Figure 4.12 below shows the output of standardisation.

```
[[-0.81607124  0.28430771  0.11960507 ...  0.76785092  1.08097258
   0.02430965]
 [-0.12616695  0.45488243 -1.40271249 ... -0.17002049 -0.78324514
  -0.10484026]
 [ 0.67561536  1.05229782 -0.04544757 ... -1.32263512  1.08097258
  -0.32009011]
 ...
 [ 0.49243984  0.23636146 -1.07081881 ...  1.04390963 -0.45845442
   2.69340773]
 [ 0.43762096 -1.35754834  0.59132503 ...  2.56077611 -0.78512673
   0.11040958]
 [-0.55510164 -1.27375995 -0.8215685  ... -0.77686418 -0.81233736
  -0.10484026]] [[ 0.47393933 -2.75435645  0.94232909 ...  0.13364409 -0.81233736
   0.3687094 ]
 [-0.4663332   1.08439428 -0.39112497 ...  0.43730868 -1.01337761
   0.75615912]
 [ 0.09093414 -1.63537475  0.13350459 ... -1.29656511  1.08097258
   0.32565943]
 ...
 [ 0.4061325   1.24636669 -0.20117981 ... -0.17002049 -1.23294425
  -0.44924002]
 [-0.98679909  1.01163781 -0.83797364 ... -0.43508981 -0.9939828
  -0.53533995]
 [ 1.01186558  0.03326419 -0.60660422 ...  0.31555156 -1.15029919
  -0.10484026]]
```

FIGURE 4.12: Output of Standardisation.

4.4 Evaluation of Model Performance

Given the highly imbalanced dataset, datasets from the two classes are sampled to obtain training data with a balanced ratio of fraud and non-fraud cases. As stated earlier, random undersampling has been found to be better than other sampling techniques for our dataset. The dataset was then split into training and testing sets, with 80% of the dataset being the training set and 20% being the testing set.

4.4.1 First Results

We started by building four baseline classification models, random forest, Xgboost, Adaboost, and mlp, without applying any imbalanced data sampling technique to see how the models would perform. Table 4.6 below shows the parameters used for each model.

TABLE 4.6: Model Parameters

	Parameters
Random Forest	n-estimators= 100
	max-features=None
	max-depth=9
	min-samples-split=5
	min-samples-leaf=1
	bootstrap=True
Xgboost	learning-rate=0.1
	colsample-bytree = 1
	subsample = 0.8
	objective=binary:logistic
	n-estimators=100
	reg-alpha = 0.3
	max-depth=6
	gamma=5
Adaboost	n-estimators=50
	learning-rate=1
MLP	solver= adam
	learning-rate= constant
	hidden-layer-sizes= (20,)
	alpha= 0.05
	activation= relu

We used various performance measures, as stated in Chapter 3 above, to evaluate four different algorithms, as shown in Table 4.7 and Table 4.8 below. Accuracy, precision, recall, F1, and support scores arise from the classification report. AUC measures the total area under the ROC curve. The tables below show the performance of four models on both the training and testing data.

From Tables 4.7 and 4.8 below, it can be observed that all four models achieve an accuracy score of 0.92 for both the training and test datasets. However, recall and f1-score values are very low, which implies that the models are performing poorly.

TABLE 4.7: Performance analysis before undersampling Training

	Train Accuracy	precision	Recall	F1-score
Random Forest	0.92	0.99	0.02	0.04
Xgboost	0.92	0.79	0.02	0.03
Adaboost	0.92	0.49	0.02	0.03
MLP	0.92	0.62	0.00	0.00

TABLE 4.8: Performance analysis before undersampling Testing

	Test Accuracy	precision	Recall	F1-score
Random Forest	0.92	0.45	0.00	0.01
Xgboost	0.92	0.57	0.01	0.02
Adaboost	0.92	0.49	0.02	0.03
MLP	0.92	0.29	0.00	0.00

To improve the performance of these models, we then apply random undersampling. After random undersampling, we had 565372 observations. From Table 4.9 below, we see a decrease in the model's accuracy. Accuracy is not the best metric to evaluate imbalanced datasets since the majority class will show a high performance value. Valuable metrics are f1-score precision or recall score. After applying random undersampling, we observe that the precision and recall of the minority class increase. Hence, random undersampling improved the performance of our models.

TABLE 4.9: Performance Analysis for Training Dataset

	RF	MLP	XGBoost	AdaBoost
Accuracy	0.72	0.67	0.72	0.67
precision	0.73	0.66	0.72	0.67
recall	0.70	0.68	0.71	0.65
f1-score	0.71	0.67	0.72	0.66
Mattews correlation	0.43	0.35	0.43	0.34
Gini Coefficient	0.60	0.47	0.59	0.46
ROC AUC	0.80	0.74	0.79	0.73

We start by reporting the accuracy and recall of implemented models. From the test dataset Table 4.10, we can observe that random forest has the highest accuracy score of 0.68. Xgboost, Adaboost, and MLP have an accuracy score of 0.67. However,

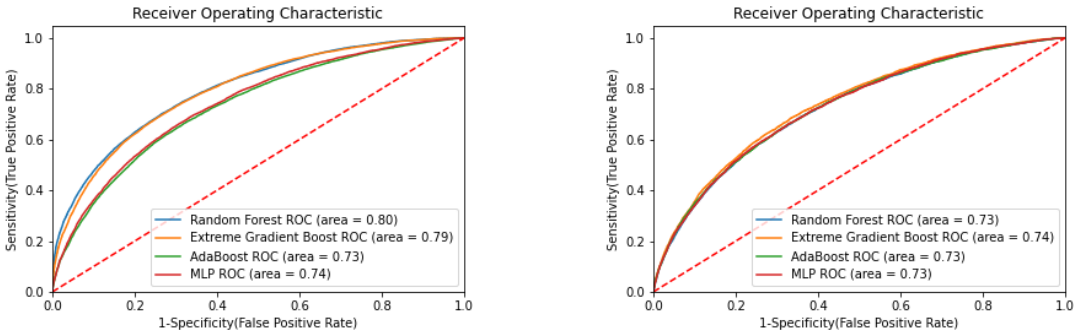
this does not mean random forest is our best model. As stated above, recall is considered a better metric compared to accuracy. Looking at the recall score, it can be observed that random forest, Xgboost, and Adaboost, classified about 0.65 of loan applications correctly.

TABLE 4.10: Performance Analysis for Test dataset

	RF	MLP	XGBoost	AdaBoost
Accuracy	0.68	0.67	0.67	0.67
precision	0.15	0.14	0.15	0.15
recall	0.65	0.64	0.65	0.65
f1-score	0.24	0.23	0.24	0.24
Mattews correlation	0.19	0.19	0.20	0.19
Gini Coefficient	0.45	0.46	0.48	0.46
ROC AUC	0.73	0.73	0.74	0.73

The Gini coefficient in fraud prediction is a ratio that represents how close our model is to being a perfect fraud prediction model and how far it is from being a random fraud prediction model. Our Gini coefficient for all our models for training lies between 0.45-0.60 and for testing lies between 0.40-0.50, which means that our models are not perfect in predicting fraud but good.

The AUC is a good metric to evaluate the performance of classifiers. It does not depend on the distribution of the class. We used class values and predicted class values obtained during the training and testing process to calculate the AUC score. To plot the AUC-ROC, we calculated the true positive rate and false positive rate and used them as our axis values. From Table 4.8 and 4.9 above and Figure 4.13 below, it can be observed that all the models have good fraud identification capabilities. However, according to Xgboost score of 0.79 for the training data and 0.74 for the test data, it is more precise and faster in identifying loan fraud.



(A) AUC-ROC Train (B) AUC-ROC Test

FIGURE 4.13: AUC-ROC Train and Test

4.4.2 Second Results

Upon reviewing the project report, we took additional steps in order to improve the performance of our models further. Initially, we did random undersampling to solve the dataset imbalance problem. In the second attempt, we used a hybrid approach of random oversampling and undersampling. We started by oversampling the dataset, and then we undersampled the oversampled dataset, and it gave an output of 565372 observations.

From Tables 4.11 and 4.12 below, we can observe that Adaboost has the highest accuracy score of 1.00 for both training and testing, followed by Xgboost with an accuracy of 0.95 for training and 0.93 for testing, and random forest with 0.89 for testing and 0.86 for testing. Adaboost, rather than being a model in and of itself, can be used on top of any classifier to learn from its mistakes and suggest a more accurate model. In our case, we applied Adaboost on top of a random forest in order to improve its performance. Looking at recall scores, it can be observed that Adaboost has the highest recall score of 1.00, followed by Xgboost with 0.99 and random forest with 0.92. This means that Adaboost was the most accurate in terms of classifying loan applications correctly. MLP performed badly compared to the previous results.

The MCC is a correlation coefficient value that lies between-1 and 1. A coefficient value of 1 represents a perfect prediction, 0 an average random prediction, and -1 an inverse prediction [35]. Hence, since for both training and testing results for all

our models, we have MCC values that lie between 0.73 and 0.99, we have a perfect prediction for fraud.

The Gini coefficient for all our models for training lies between 0.91–1.00 and for testing lies between 0.88–1.00, with Adaboost having the highest score, followed by Xgboost and then random forest. This means that our models are perfect in predicting fraud.

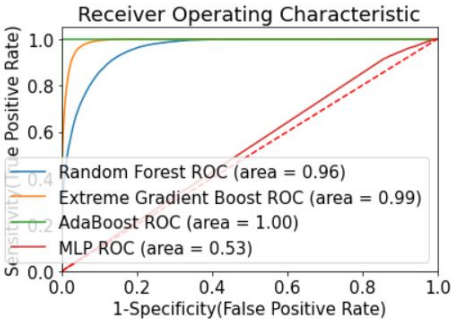
TABLE 4.11: Performance Analysis for Training Dataset

	RF	MLP	XGBoost	AdaBoost
Accuracy	0.89	0.53	0.95	1.00
precision	0.87	0.55	0.93	1.00
recall	0.92	0.35	0.99	1.00
f1-score	0.89	0.43	0.96	1.00
Mattews correlation	0.78	0.07	0.91	1.00
Gini Coefficient	0.91	0.08	0.98	1.00
ROC AUC	0.96	0.53	0.99	1.00

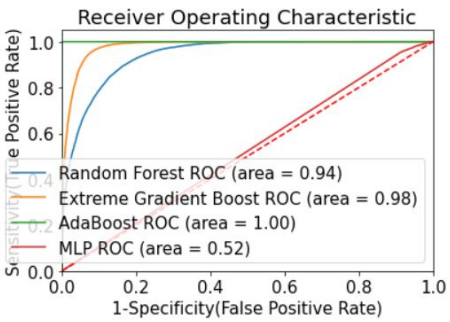
TABLE 4.12: Performance Analysis for Test Dataset

	RF	MLP	XGBoost	AdaBoost
Accuracy	0.86	0.53	0.93	1.00
precision	0.84	0.55	0.89	0.99
recall	0.90	0.35	0.98	1.00
f1-score	0.87	0.43	0.93	1.00
Mattews correlation	0.73	0.07	0.86	0.99
Gini Coefficient	0.88	0.09	0.95	1.00
ROC AUC	0.94	0.52	0.98	1.00

From Tables 4.11 and 4.12 above and Figure 4.14 below, it can be observed that random forest, Xgboost, and Adaboost have good fraud identification capabilities. However, mlp has the lowest score of 0.53 for training and 0.52 for testing.



(A) AUC-ROC Train



(B) AUC-ROC Test

FIGURE 4.14: AUC-ROC Train and Test

KS statistics are used in fraud prediction to know how many transactions to target in order to be able to predict fraud. Precisely, it measures the degree of separation between fraud and non-fraud transactions. From figures 4.15, 4.17, and 4.18 below, we can see that our ks for random forest, Xgboost, and Adaboost lie between 0.73 and 1.00. However, for mlp it is 0.07, which is bad because it means it is not doing well in separating fraud and non-fraud transactions.

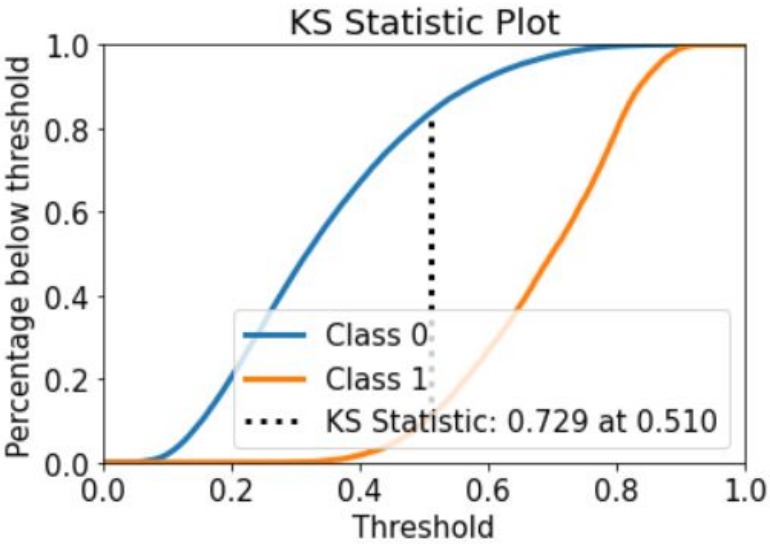


FIGURE 4.15: KS Chart Random Forest

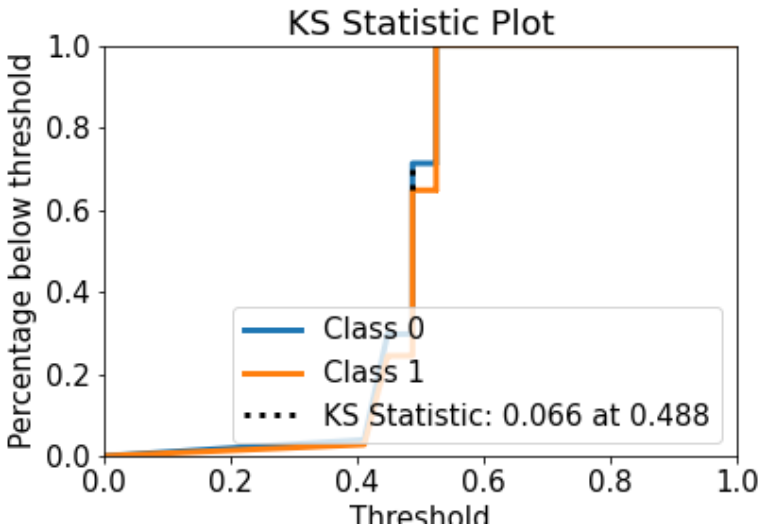


FIGURE 4.16: KS Chart MLP

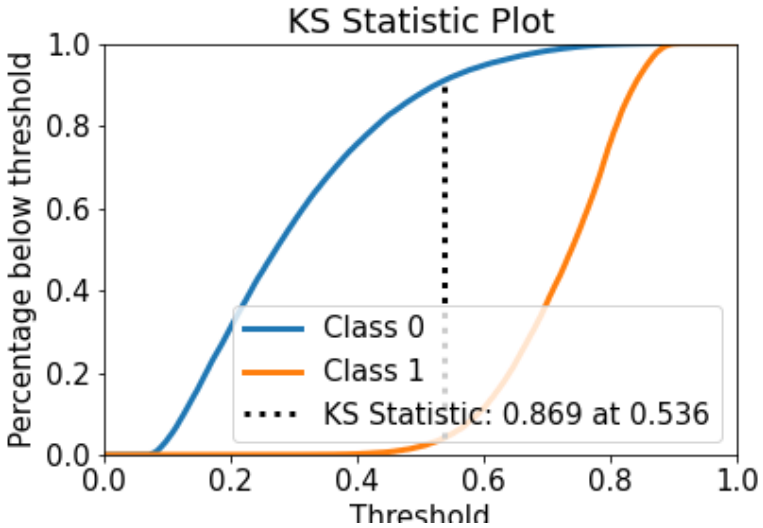


FIGURE 4.17: KS Chart Xgboost

4.5 Summary

This chapter discussed results obtained from exploratory data analysis and experimental results for the four implemented machine learning approaches. We presented data exploration results in graphs and tables to get more data insights. Exploratory data analysis helped us understand the overview of the dataset. We

performed feature selection to improve the performance of our four models. Finally, we presented experimental results obtained from four machine learning approaches in the form of tables and graphs.

Chapter 5

Summary, Conclusions and Recommendations

5.1 Summary

This research focused on predicting loan fraud using four machine learning models, namely, random forest, extreme gradient boost, Adaboost, and multilayer perceptron. The dataset used is the Kaggle fraud detection dataset for the year 2019. Exploratory data analysis helped us understand the overview of the dataset. The original dataset contained 122 features, and random forest was used to select the top 14 features. Random undersampling was used to solve the data imbalance problem.

Initially, we trained the four models on an unbalanced dataset, and the accuracy of the models was very high, which does not imply good performance since accuracy is not a good metric to evaluate imbalanced datasets, as stated in Chapter 4 above. Based on recall and f1 scores, the overall performance for all the models was bad and the models were overfitting the dataset, hence randomising. After applying undersampling to the dataset, the accuracy of the models decreased and the overall performance of all the models increased. Since accuracy is not a good enough measure for our problem, metrics such as recall, precision, f1 score, MCC, gini coefficient, AUC-ROC, and ks statistics were used to evaluate the performance further. The results obtained during the first attempt suggest that Xgboost performed best compared to the other three models, followed by random forest.

We then made a second attempt using a hybrid technique of random undersampling and random oversampling, and this improved the performance of our models. This agrees with results from [3], hybrid of undersampling and oversampling on imbalanced datasets indeed improves prediction performance drastically. To improve the performance of Adaboost further, we trained it on top of the random forest model. Adaboost performed best during the second attempt, followed by Xgboost and then random forest. In the literature presented in chapter 2 above, several papers display that Xgboost and random forest perform well in predicting credit card fraud [13] [69] [5]; the result we obtained also suggest that these models perform well in predicting loan fraud.

5.2 Conclusions

The goal of this study was to compare different machine learning algorithms for predicting fraud in loan administration in order to find the methods that give the most accurate results. Financial fraud prediction is critical because it can prevent large sums of money from being extorted. Our study is significant in that direction since it shows how machine learning may be used to address this crucial concern. Results presented in Chapter 4 above show that applying random sampling techniques increased the overall performance of the models while minimising the incorrect fraud classification. It also shows that it is possible to achieve good performance in loan fraud prediction using the proposed algorithms. However, not all machine learning models are effective in predicting loan fraud. Imbalanced data significantly affects the performance of classification models, as most models turn to maximising accuracy and reducing errors. This was found to be an issue in the reviewed literature as well.

5.3 Recommendations and future work

In the future, we will consider using different datasets to test our models. This could improve the performance of the models. Also, future work should consider using hybrid algorithms. Some recommendations include:

a. Researchers should use a more balanced dataset as it improves prediction accuracy.

b. Researchers should consider using a hybrid dataset sampling technique as it has been proven to improve the performance of a prediction model greatly.

c. More effective data processing and feature selection steps should be used since they have been proven to improve prediction performance.

Dataset Description

1. SK-IDCURR: ID of loan in our sample
2. CLASS: Target variable (1 - client with payment difficulties: he/she had late payment more than X days on at least one of the first Y installments of the loan in our sample, 0 - all other cases)
3. NAME-CONTRACT-TYPE: Identification if loan is cash or revolving
4. CODE-GENDER: Gender of the client
5. FLAG-OWN-CAR: Flag if the client owns a car
6. FLAG-OWN-REALTY: Flag if client owns a house or flat
7. CNT-CHILDREN: Number of children the client has
8. AMT-INCOME-TOTAL: Income of the client
9. AMT-CREDIT: Credit amount of the loan
10. AMT-ANNUITY: Loan annuity
11. AMT-GOODS-PRICE: For consumer loans it is the price of the goods for which the loan is given
12. NAME-TYPE-SUITE: Who was accompanying client when he was applying for the loan
13. NAME-INCOME-TYPE: Clients income type (businessman, working, maternity leave,etc)
14. NAME-EDUCATION-TYPE: Level of highest education the client achieved
15. NAME-FAMILY-STATUS:Family status of the client
16. NAME-HOUSING-TYPE: What is the housing situation of the client (renting, living with parents, etc)
17. REGION-POPULATION-RELATIVE: Normalized population of region where client lives (higher number means the client lives in more populated region)

18. DAYS-BIRTH: Client's age in days at the time of application time only relative to the application
19. DAYS-EMPLOYED: How many days before the application the person started current employment
20. DAYS-REGISTRATION: How many days before the application did client change his registration time only relative to the application
21. DAYS-ID-PUBLISH: How many days before the application did client change the identity document with which he applied for the loan
22. OWN-CAR-AGE: Age of client's car
23. FLAG-MOBIL: Did client provide mobile phone (1=YES, 0=NO)
24. FLAG-EMP-PHONE: Did client provide work phone (1=YES, 0=NO)
25. FLAG-WORK-PHONE: Did client provide home phone (1=YES, 0=NO)
26. FLAG-CONT-MOBILE: Was mobile phone reachable (1=YES, 0=NO)
27. FLAG-PHONE: Did client provide home phone (1=YES, 0=NO)
28. FLAG-EMAIL: Did client provide email (1=YES, 0=NO)
29. OCCUPATION-TYPE: What kind of occupation does the client have
30. CNT-FAM-MEMBERS: How many family members does client have
31. REGION-RATING-CLIENT: Our rating of the region where client lives (1,2,3)
32. REGION-RATING-CLIENT-W-CITY: Our rating of the region where client lives with taking city into account (1,2,3)
33. WEEKDAY-APPR-PROCESS-START: On which day of the week did the client apply for the loan
34. HOUR-APPR-PROCESS-START: Approximately at what hour did the client apply for the loan rounded

35. REG-REGION-NOT-LIVE-REGION: Flag if client's permanent address does not match contact address (1=different, 0=same, at region level)

36. REG-REGION-NOT-WORK-REGION: Flag if client's permanent address does not match work address (1=different, 0=same, at region level)

37. LIVE-REGION-NOT-WORK-REGION: Flag if client's contact address does not match work address (1=different, 0=same, at region level)

38. REG-CITY-NOT-LIVE-CITY: Flag if client's permanent address does not match contact address (1=different, 0=same, at city level)

39. REG-CITY-NOT-WORK-CITY: Flag if client's permanent address does not match work address (1=different, 0=same, at city level)

40. LIVE-CITY-NOT-WORK-CITY: Flag if client's contact address does not match work address (1=different, 0=same, at city level)

41. ORGANIZATION-TYPE: Type of organization where client works

42. EXT-SOURCE-1: Normalized score from external data source

43. EXT-SOURCE-2: Normalized score from external data source

44. EXT-SOURCE-3: Normalized score from external data source

45. APARTMENTS-AVG: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

46. BASEMENTAREA-AVG: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

47. YEARS-BEGINEXPLUATATION-AVG: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age

of building, number of elevators, number of entrances, state of the building, number of floor

48. YEARS-BUILD-AVG: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

49. COMMONAREA-AVG: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

50. ELEVATORS-AVG: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

51. ENTRANCES-AVG: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

52. FLOORSMAX-AVG: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

53. FLOORSMIN-AVG: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

54. LANDAREA-AVG: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

55. LIVINGAPARTMENTS-AVG: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

56. LIVINGAREA-AVG: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

57. NONLIVINGAPARTMENTS-AVG: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

58. NONLIVINGAREA-AVG: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

59. APARTMENTS-MODE: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

60. BASEMENTAREA-MODE: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

61. YEARS-BEGINEXPLUATATION-MODE: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age

of building, number of elevators, number of entrances, state of the building, number of floor

62. YEARS-BUILD-MODE: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

63. COMMONAREA-MODE: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

64. ELEVATORS-MODE: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

65. ENTRANCES-MODE: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

66. FLOORSMAX-MODE: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

67. FLOORSMIN-MODE: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

68. LANDAREA-MODE: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

69. LIVINGAPARTMENTS-MODE: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

70. LIVINGAREA-MODE: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

71. NONLIVINGAPARTMENTS-MODE: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

72. NONLIVINGAREA-MODE: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

73. APARTMENTS-MEDI: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

74. BASEMENTAREA-MEDI: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

75. YEARS-BEGINEXPLUATATION-MEDI: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age

of building, number of elevators, number of entrances, state of the building, number of floor

76. YEARS-BUILD-MEDI: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

77. COMMONAREA-MEDI: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

78. ELEVATORS-MEDI: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

79. ENTRANCES-MEDI Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

80. FLOORSMAX-MEDI Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

81. FLOORSMIN-MEDI Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

82. LANDAREA-MEDI: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

83. LIVINGAPARTMENTS-MEDI: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

84. LIVINGAREA-MEDI: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

85. NONLIVINGAPARTMENTS-MEDI: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

86. NONLIVINGAREA-MEDI: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

87. FONDKAPREMONT-MODE: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

88. HOUSETYPE-MODE: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

89. TOTALAREA-MODE: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

90. WALLSMATERIAL-MODE: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

91. EMERGENCYSTATE-MODE: Normalized information about building where the client lives, What is average (-AVG suffix), modus (-MODE suffix), median (-MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

92. OBS-30-CNT-SOCIAL-CIRCLE: How many observation of client's social surroundings with observable 30 DPD (days past due) default

93. DEF-30-CNT-SOCIAL-CIRCLE: How many observation of client's social surroundings defaulted on 30 DPD (days past due)

94. OBS-60-CNT-SOCIAL-CIRCLE: How many observation of client's social surroundings with observable 60 DPD (days past due) default

95. DEF-60-CNT-SOCIAL-CIRCLE: How many observation of client's social surroundings defaulted on 60 (days past due)

96. DAYS-LAST-PHONE-CHANGE: How many days before application did client change phone

97. FLAG-DOCUMENT-2: Did client provide document 2 100 application-data FLAG-DOCUMENT-3 Did client provide document 3

98. FLAG-DOCUMENT-4: Did client provide document 4 102 application-data FLAG-DOCUMENT-5 Did client provide document 5

99. FLAG-DOCUMENT-6: Did client provide document 6

100. FLAG-DOCUMENT-7 Did client provide document 7

101. FLAG-DOCUMENT-8: Did client provide document 8

102. FLAG-DOCUMENT-9: Did client provide document 9

103. FLAG-DOCUMENT-10: Did client provide document 10
104. FLAG-DOCUMENT-11: Did client provide document 11
105. FLAG-DOCUMENT-12: Did client provide document 12
106. FLAG-DOCUMENT-13: Did client provide document 13
107. FLAG-DOCUMENT-14: Did client provide document 14
108. FLAG-DOCUMENT-15: Did client provide document 15
109. FLAG-DOCUMENT-16: Did client provide document 16
110. FLAG-DOCUMENT-17: Did client provide document 17
111. FLAG-DOCUMENT-18: Did client provide document 18
112. FLAG-DOCUMENT-19: Did client provide document 19
113. FLAG-DOCUMENT-20: Did client provide document 20
114. FLAG-DOCUMENT-21: Did client provide document 21
115. AMT-REQ-CREDIT-BUREAU-HOUR: Number of enquiries to Credit Bureau about the client one hour before application
116. AMT-REQ-CREDIT-BUREAU-DAY: Number of enquiries to Credit Bureau about the client one day before application (excluding one hour before application)
117. AMT-REQ-CREDIT-BUREAU-WEEK: Number of enquiries to Credit Bureau about the client one week before application (excluding one day before application)
118. AMT-REQ-CREDIT-BUREAU-MON: Number of enquiries to Credit Bureau about the client one month before application (excluding one week before application)
119. AMT-REQ-CREDIT-BUREAU-QRT: Number of enquiries to Credit Bureau about the client 3 month before application (excluding one month before application)

120. AMT-REQ-CREDIT-BUREAU-YEAR: Number of enquiries to Credit Bureau about the client one day year (excluding last 3 months before application)

Bibliography

[1] R. Agrawal. *Interpolation – Power of Interpolation in Python to fill Missing Values*. (2021). URL: https://www.analyticsvidhya.com/blog/2021/06/power-of-interpolation-in-python-to-fill-missing-values/.

[2] B. Akkaya and N. Çolakoğlu. "Comparison of Multi-class Classification Algorithms on Early Diagnosis of Heart Diseases". In: Sept. 2019.

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

[4] R. Banerjee et al. "Comparative Analysis of Machine Learning Algorithms through Credit Card Fraud Detection". In: *2018 IEEE MIT Undergraduate Research Technology Conference (URTC)*. 2018, pp. 1–4. DOI: 10.1109/URTC45901.2018.9244782.

[5] S. Bhattacharyya et al. "Data mining for credit card fraud: A comparative study". In: *Decision support systems* 50.3 (2011), pp. 602–613.

[6] G Biau. "Analysis of a random forests model". In: *Journal of Machine Learning Research* 13.Apr (2012), pp. 1063–1095.

[7] J. Błaszczyński et al. "Auto loan fraud detection using dominance-based rough set approach versus machine learning methods". In: *Expert Systems with Applications* 163 (2021), p. 113740.

[8] R. J. Bolton, D. J. Hand, et al. "Statistical fraud detection: A review". In: *Statistical science* 17.3 (2002), pp. 235–255.

[9] J. Brownlee. *Crash Course On Multi-Layer Perceptron Neural Networks*. (2016). URL: https://machinelearningmastery.com/neural-networks-crash-course/.

[10] T. A. CHOUGULE et al. "Genetic k-means algorithm for credit card fraud detection". In: *International Journal of Computer Science and Information Technologies (IJCSIT)* 6.2 (2015), pp. 1724–1727.

[11] crifhighmark. *What Is Unsecured Loan Fraud and What Are the Ways in Which It Occurs?* (2020). URL: https://blog.crifhighmark.com/what-is-unsecured-loan-fraud-and-what-are-the-ways-in-which-it-occurs/.

[12] R Devaki, V Kathiresan, and S Gunasekaran. "Credit card fraud detection using time series analysis". In: *International Journal of Computer Applications* 3 (2014), pp. 8–10.

[13] S. Dhankhad, E. Mohammed, and B. Far. "Supervised machine learning algorithms for credit card fraudulent transaction detection: a comparative study". In: *2018 IEEE International Conference on Information Reuse and Integration (IRI)*. IEEE. 2018, pp. 122–125.

[14] K. Divakar and K Chitharanjan. "Performance evaluation of credit card fraud transactions using boosting algorithms". In: *Int. J. Electron. Commun. Comput. Eng. IJECCE* 10.6 (2019), pp. 262–270.

[15] A. Dubey. *Feature Selection Using Random forest*. (2018). URL: https://towardsdatascience.com/feature-selection-using-random-forest-26d7b747597f.

[16] I. C. Education. *Exploratory Data Analysis*. (2020). URL: https://www.ibm.com/cloud/learn/exploratory-data-analysis.

[17] I. Eweoya et al. "A Naive Bayes approach to fraud prediction in loan default". In: *Journal of Physics: Conference Series*. Vol. 1299. 1. IOP Publishing. 2019, p. 012038.

[18] I. Eweoya et al. "Fraud prediction in bank loan administration using decision tree". In: *Journal of Physics: Conference Series*. Vol. 1299. 1. IOP Publishing. 2019, p. 012037.

[19] I. Eweoya et al. "Fraud prediction in loan default using support vector machine". In: *Journal of Physics: Conference Series*. Vol. 1299. 1. IOP Publishing. 2019, p. 012039.

[20] G. Facts. *Startups worldwide*. (2014). URL: https://legaldictionary.net/fraud/.

[21] S. Falaki et al. “Probabilistic credit card fraud detection system in online transactions”. In: *International Journal of Software Engineering and Its Applications* 6.4 (2012), pp. 69–78.

[22] S. G. Fashoto et al. “Hybrid methods for credit card fraud detection using K-means clustering with hidden Markov model and multilayer perceptron algorithm”. In: *Current Journal of Applied Science and Technology* (2016), pp. 1–11.

[23] A. Folk. *Loan Fraud Lawyers*. (2020). URL: https://www.legalmatch.com/law-library/article/loan-fraud-lawyers.html.

[24] I. P. S. F. Forum. “International Public Sector Fraud Forum Guide to Understanding the Total Impact of Fraud”. In: (2020).

[25] F. Itoo, S. Singh, et al. “Comparison and analysis of logistic regression, Naïve Bayes and KNN machine learning algorithms for credit card fraud detection”. In: *International Journal of Information Technology* 13.4 (2021), pp. 1503–1511.

[26] V. Jain. *Everything you need to know about “Activation Functions” in Deep learning models*. (2019). URL: https://towardsdatascience.com/everything-you-need-to-know-about-activation-functions-in-deep-learning-models-84ba9f82c253.

[27] S. Kampakis. *Performance Measures: Cohen’s Kappa statistic*. (2020). URL: https://thedatascientist.com/performance-measures-cohens-kappa-statistic/.

[28] V. Karbhari. *What is AUC?* (2019). URL: https://medium.com/acing-ai/what-is-auc-446a71810df9.

[29] S KRISHNAN. “Credit Card Nearest Neighbor Based Outlier Detection Techniques”. In: ().

[30] N. Kumar. *Advantages of XGBoost Algorithm in Machine Learning*. (2019). URL: http://theprofessionalspoint.blogspot.com/2019/03/advantages-of-xgboost-algorithm-in.html.

[31] A. Kundu et al. “Blast-ssaha hybridization for credit card fraud detection”. In: *IEEE transactions on dependable and Secure Computing* 6.4 (2009), pp. 309–315.

[32] V. Kurama. *A Guide to AdaBoost: Boosting To Save The Day*. (2019). URL: https://blog.paperspace.com/adaboost-optimizer/.

[33] C. Kusaya and J. O'Keefe. "Insider Abuse and Fraud Prediction for US Banks: A Comparison of Machine Learning Approaches". In: *Available at SSRN 3659757* (2020).

[34] S. Lakshmi and S. Kavilla. "Machine learning for credit card fraud detection system". In: *International Journal of Applied Engineering Research* 13.24 Pt. 1 (2018), pp. 16819–16824.

[35] S. learn. *Scikit learn. Matthews correlation coefficient.* (2022). URL: https://scikit-learn.org/stable/modules/model_evaluation.html#matthews-corrcoef.

[36] A. Lindholm. *A study about fraud detection and the implementation of SUSPECT-Supervised and UnSuPervised Erlang Classifier Tool*. 2014.

[37] J. Mahanta. *introduction to neural networks advantages and applications*. (2017). URL: https://towardsdatascience.com/introduction-to-neural-netwoks-advantages-and-applications-96851bd1a207.

[38] N Malini and M Pushpa. "Analysis on credit card fraud identification techniques based on KNN and outlier detection". In: *2017 Third International Conference on Advances in Electrical, Electronics, Information, Communication and Bio-Informatics (AEEICB)*. IEEE. 2017, pp. 255–258.

[39] E. Minastireanu and G. Mesnita. "An Analysis of the Most Used Machine Learning Algorithms for Online Fraud Detection". In: Informatica Economica, 23 (Mar. 2019), pp. 5–16. DOI: 10.12948/issn14531305/23.1.2019.01.

[40] A. Mishra and C. Ghorpade. "Credit card fraud detection on the skewed data using various classification and ensemble techniques". In: *2018 IEEE International Students' Conference on Electrical, Electronics and Computer Science (SCEECS)*. IEEE. 2018, pp. 1–5.

[41] M. K. Mishra and R. Dash. "A comparative study of chebyshev functional link artificial neural network, multi-layer perceptron and decision tree for credit card fraud detection". In: *2014 International Conference on Information Technology*. IEEE. 2014, pp. 228–233.

[42] S. Narkhede. *Understanding AUC - ROC Curve*. (2018). URL: https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5.

[43] nexocode. *Loan Application Fraud Detection*. (2022). URL: https://nexocode.com/case-studies/loan-application-fraud-detection/.

[44] J. Nikulski. *The Ultimate Guide to AdaBoost, random forests and XGBoost*. (2020). URL: https://towardsdatascience.com/the-ultimate-guide-to-adaboost-random-forests-and-xgboost-7f9327061c4f.

[45] K. Nishida. *Introduction to Extreme Gradient Boosting in Exploratory*. (2017). URL: https://blog.exploratory.io/introduction-to-extreme-gradient-boosting-in-exploratory-7bbec554ac7.

[46] X. Niu, L. Wang, and X. Yang. "A comparison study of credit card fraud detection: Supervised versus unsupervised". In: *arXiv preprint arXiv:1904.10604* (2019).

[47] Y.-S. Park and S Lek. "Artificial neural networks: multilayer perceptron for ecological modeling". In: *Developments in environmental modelling*. Vol. 28. Elsevier, 2016, pp. 123–140.

[48] A Prakash and C Chandrasekar. "An optimized multiple semi-hidden Markov model for credit card fraud detection". In: *Indian Journal of Science and Technology* 8.2 (2015), p. 176.

[49] A. Pumsirirat and L. Yan. "Credit card fraud detection using deep learning based on auto-encoder and restricted boltzmann machine". In: *International Journal of advanced computer science and applications* 9.1 (2018), pp. 18–25.

[50] PWC. *PwC's Global Economic Crime and Fraud Survey 2020*. (2020). URL: https://www.pwc.com/gx/en/services/forensics/economic-crime-survey.html.

[51] P. Raghavan and N. El Gayar. "Fraud detection using machine learning and deep learning". In: *2019 international conference on computational intelligence and knowledge economy (ICCIKE)*. IEEE. 2019, pp. 334–339.

[52] G. B. Rath, D. Das, and B. Acharya. "Modern Approach for Loan Sanctioning in Banks Using Machine Learning". In: *Advances in Machine Learning and Computational Intelligence*. Springer, 2021, pp. 179–188.

[53] N. Rtayli and N. Enneya. "Selection features and support vector machine for credit card risk identification". In: *Procedia Manufacturing* 46 (2020), pp. 941–948.

[54] I Sadgali, N Sael, and F Benabbou. "Performance of machine learning techniques in the detection of financial frauds". In: *Procedia computer science* 148 (2019), pp. 45–54.

[55] Y. Sahin and E. Duman. "Detecting credit card fraud by ANN and logistic regression". In: *2011 International Symposium on Innovations in Intelligent Systems and Applications*. IEEE. 2011, pp. 315–319.

[56] A. Samudrala. *Unveiling Mathematics Behind XGBoost*. (2018). URL: https://www.kdnuggets.com/2018/08/unveiling-mathematics-behind-xgboost.html.

[57] R. Shaikh. *Choosing the right Encoding method-Label vs OneHot Encoder*. (2018). URL: https://towardsdatascience.com/choosing-the-right-encoding-method-label-vs-onehot-encoder-a4434493149b.

[58] N. Sharma. "Credit Card Fraud Detection Predictive Modeling". In: (2020).

[59] K. P. Shung. *Accuracy, Precision, Recall or F1?* (2018). URL: https://towardsdatascience.com/accuracy-precision-recall-or-f1-331fb37c5cb9.

[60] A. Singh, D. Narayan, et al. "A survey on hidden markov model for credit card fraud detection". In: *International Journal of Engineering and Advanced Technology (IJEAT)* 1.3 (2012), pp. 49–52.

[61] A. Srivastava et al. "Credit Card Fraud Detection Using Hidden Markov Model". In: *IEEE Transactions on Dependable and Secure Computing* 5.1 (2008), pp. 37–48. DOI: 10.1109/TDSC.2007.70228.

[62] T. Srivastava. *11 Important Model Evaluation Metrics for Machine Learning Everyone should know*. (2019). URL: https://www.analyticsvidhya.com/blog/2019/08/11-important-model-evaluation-error-metrics/.

[63] StackOverflow. *Adaboost Algorithm Walkthrough for non-technical*. (2017). URL: https://stackoverflow.com/questions/42604643/adaboost-algorithm-walkthrough-for-non-technical.

[64] C. Team. *Random Forest*. (2021). URL: https://corporatefinanceinstitute.com/resources/knowledge/other/random-forest/.

[65] S. how to. *Matthews Correlation Coefficient*. (2020). URL: https://www.statisticshowto.com/matthews-correlation-coefficient/.

[66] D. Varmedja et al. “Credit card fraud detection-machine learning methods”. In: *2019 18th International Symposium INFOTEH-JAHORINA (INFOTEH)*. IEEE. 2019, pp. 1–5.

[67] analytics vidhya. *Overcoming Class Imbalance using SMOTE Techniques*. (2020). URL: https://www.analyticsvidhya.com/blog/2020/10/overcoming-class-imbalance-using-smote-techniques/.

[68] L. Wei-Yin. “Classification and regression trees”. In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 1.1 (2011), pp. 14–23.

[69] H. Wen and F. Huang. “Personal loan fraud detection based on hybrid supervised and unsupervised learning”. In: *2020 5th IEEE International Conference on Big Data Analytics (ICBDA)*. IEEE. 2020, pp. 339–343.

[70] M. Widmann. *Cohen's Kappa: What It Is, When to Use It, and How to Avoid Its Pitfalls*. (2020). URL: https://thenewstack.io/cohens-kappa-what-it-is-when-to-use-it-and-how-to-avoid-its-pitfalls/.

[71] M. Wu, Y. Huang, and J. Duan. “Investigations on classification methods for loan application based on machine learning”. In: *2019 International Conference on Machine Learning and Cybernetics (ICMLC)*. IEEE. 2019, pp. 1–6.

[72] S. Xuan et al. “Random forest for credit card fraud detection”. In: *2018 IEEE 15th International Conference on Networking, Sensing and Control (ICNSC)*. 2018, pp. 1–6. DOI: 10.1109/ICNSC.2018.8361343.

[73] Q. Zhan and H. Yin. “A loan application fraud detection method based on knowledge graph and neural network”. In: *Proceedings of the 2nd International Conference on Innovation in Artificial Intelligence*. 2018, pp. 111–115.