UNIVERSITY OF NAIROBI

APPLICATION OF SUPERVISED MACHINE LEARNING ALGORITHMS FOR FRAUD DETECTION IN VEHICLE INSURANCE CLAIMS

Submitted by:

Muturi Sharlyn Wangui - I07/1500/2020
Oriedo Sharlyne Awinja - I07/1508/2020
Omenya Mercy Awili - I07/1509/2020
Odero Silas Ochieng - I07/1530/2020
Macharia Brian Mwangi - I07/1529/2020

Supervisors:

Prof. Caroline A. Ogutu
Prof. Davis N. Bundi

Submitted in Partial Fulfillment of the Requirements for the Award of Degree in BSc. Actuarial Science, Department of Mathematics, University of Nairobi.

Declaration

We, the undersigned, certify that this research project report is our original work and hasn't been submitted to the University of Nairobi or any other institution in exchange for a degree or other recognized academic achievement. Every source utilized in this research, including books, papers, and other references, has been thoroughly cited and duly acknowledged.

Signature: _____ Date: _____

Muturi Sharlyn Wangui

Registration Number: I07/1500/2020

Signature: _____ Date: _____

Oriedo Sharlyne Awinja

Registration Number: I07/1508/2020

Signature: _____ Date: _____

Omenya Mercy Awili

Registration Number: I07/1509/2020

Signature: _____ Date: _____

Odero Silas Ochieng

Registration Number: I07/1530/2020

Signature: _____ Date: _____

Macharia Brian Mwangi

Registration Number: I07/1529/2020

With my consent as a university supervisor, this research project report has been submitted for evaluation in partial fulfillment of the criteria for the University of Nairobi's Bachelor of Science in Actuarial Science degree.

Signature: _____ Date: _____

Prof. Caroline A. Ogutu

Signature: _____ Date: _____

Prof. Davis Bundi

Acknowledgement

We would like to express our sincere gratitude to all those who contributed to the successful completion of this project. First and foremost, we thank the Almighty God for seeing us through the successful completion of our research project. Profound appreciation to our supervisors, Prof. Caroline Ogutu and Prof. Davis Bundi, for their invaluable guidance, encouragement, and unwavering support throughout the duration of this research endeavor. Their expertise and insightful feedback have been instrumental in shaping the trajectory of our work. Last but not least, we express our profound appreciation to our families and friends for their unwavering love, encouragement, and understanding during this journey.

This project would not have been possible without the collective efforts of all those mentioned above, and for that, we are truly grateful. We are grateful for the opportunities that have been given to us to advance professionally and broaden our knowledge, skills, and experiences.

Abstract

Insurance fraud, especially in motor insurance has been escalating as a major concern for the reliability and longevity of insurance operations. This research aims at examining supervised machine learning algorithms as proactive ways to curb insurance fraud. The objective of this study was to determine how supervised machine learning algorithms can be used to identify false vehicle insurance claims. More specifically, the study deals with training supervised machine learning models (Logistic Regression, XGBoost and Random Forest) on historical insurance data; evaluating their performance; and comparing them in terms of accuracy between legitimate and fraudulent claims. The study utilizes the R programming environment. The model evaluation confirmed that the Random Forest algorithm was the most robust model which had higher accuracy and F1 Score. XGBoost had the highest recall. Logistic Regression had the best precision although its overall performance was inferior when compared with Random Forest. The study comprehensively showed that various algorithms are good at different things but Random Forest emerged as the best algorithm for detecting fraudulent claims based on multiple metrics.

The study points to machine learning techniques being effective in fraud identification, especially the Random Forest algorithm. This can only be made possible if such companies adopt these state-of-the-art models which will protect them from any fraudulent activities and ensure their financial stability. It means that insurers have no choice but to constantly innovate new ways of handling the dynamic nature of insurance fraud.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background of Study

Vehicle insurance is a fundamental component of the broader insurance sector, playing a pivotal role in safeguarding both individuals and businesses from financial losses resulting from unforeseen events such as accidents and theft. Insurance constitutes a contractual relationship between the insurer and policyholder, wherein the policyholder agrees to pay a premium in exchange for financial protection provided by the insurer in the event of a covered loss. This protection is activated upon the formal filing of a claim by either the policyholder or a third party, stemming from a loss resulting from future, uncertain events. Both parties are expected to uphold the principle of utmost good faith throughout their interactions (Viaene et al., 2004).

Fraud can be characterized as a criminal act perpetrated for unlawful financial gain, typically employing deceit as its primary method. Instances of insurance fraud occur when individuals deceive insurers, intermediaries, or other parties for personal gain, either during the underwriting process or when submitting a claim. Common fraudulent practices include "padding" claims, that is inflating claim amounts, providing false information on insurance applications, filing claims for injuries or damages that did not occur, and staging accidents.

In Kenya, the Insurance Regulatory Authority (IRA) is the state corporation mandated to regulate, supervise and promote development of the insurance industry in Kenya. According to IRA (2021), the entity is obliged by Kenyan legislation under Cap 486 of the Insurance Act

passed in 1988. The Insurance Fraud Investigation Unit (IFIU) was established in 2011 as a unit under criminal investigation unit to work with IRA. IFIU receives reports of suspected insurance fraud, coordinates investigations with other law enforcement state agencies and facilitates the arrest and prosecution of suspects in court.

According to IRA's report released for the first quarter in 2023 (January – March 2023), 61 insurance fraud cases were reported to IFIU. The number of fraud cases reported in each month of the first quarter of 2023 were 7 in January, 15 in February, and 39 in March. The classification of fraud cases by nature in the first quarter of 2023 is illustrated in Table 1.1 below.

Cases	**Number**
Theft / Stealing by Agents	12
Fraudulent Motor Accident (Injury) Claims	11
Forgery	11
Issuance of Fake Motor Vehicle Insurance Certificate	7
Obtaining money by False Pretenses	5
Double Registration of Motor Vehicles	4
Complaint Against an Agent	4
Fraudulent Funeral/ Death Claims	2
Theft by insurance company employees	2
Non-compliance with insurance act	2
Fraudulent medical insurance claim	1
Total	61

Table 1.1: Nature of the Insurance Fraud Cases Reported Q1 2023

Table 1.1 reveals that for the first quarter of 2023, insurance fraud cases consisted 22 originating from the motor insurance, with 11 fraudulent motor accident claims, 7 issuance of fake motor vehicle insurance certificate and 4 double registration of motor vehicles.

For the second quarter of 2023 (April – June 2023), IRA reported 51 insurance fraud cases to the Insurance Fraud Investigation Unit (IFIU). The number of fraud cases reported in each month of the second quarter of 2023 were 18 in April, 18 in May and 15 in June. The classification of fraud cases by nature in the second quarter of 2023 is illustrated in Table 1.2.

Cases	Number
Forgery	22
Obtaining money by False Pretenses	9
Double Registration	7
Fraudulent medical insurance claims	4
Fraudulent Funeral/Death Claims	3
Fraudulent Motor Accident (Injury) Claims	2
Complaint Against an Agent	1
Theft/Stealing by an Agent	1
Theft by insurance company employees	1
Non-compliance with insurance act	1
Total	51

Table 1.2: Nature of the Insurance Fraud Cases Reported Q2 2023

The findings in Table 1.2 revealed that in the second quarter of 2023, 51 fraud cases were reported out of which were 2 fraudulent motor accident claims.

During the third quarter of 2023 (July - September 2023), 38 insurance fraud cases were reported to IFIU. The number of fraud cases reported in each month of the third quarter were 14 in July, 12 in August and 12 in September. The classification of fraud cases by nature in the third quarter of 2023 is displayed in Table 1.3.

Cases	Number
Forgery	4
Obtaining money by False Pretenses	9
Issuance of fake motor vehicle insurance certificate	11
Fraudulent last expenses	2
Fraudulent Funeral/Death Claims	2
Fraudulent Motor Accident (Injury) Claims	2
Theft by servant	3
Complaint against corporate insurance companies	5
Total	38

Table 1.3: Nature of the Insurance Fraud Cases Reported Q3 2023

Table 1.3 reveals that in the third quarter of 2023, 38 fraud cases were reported out of which were 2 fraudulent motor accident (injury) claims.

While the reports by IRA suggest that reported instances of fraud have decreased throughout 2023, this decline does not solely reflect a reduction in fraudulent activity. Several factors

could contribute to this trend. For instance, economic shifts, such as tighter budgets or increased financial caution among consumers, could lead to a reduced frequency of claims being filed. Additionally, changes in market dynamics or alterations in insurance policy terms and conditions might influence individuals' decisions regarding claim submissions. Furthermore, insurance companies may opt to settle fraudulent claims rather than pursuing costly legal actions, contributing to a decrease in reported fraudulent activity. However, it's crucial to acknowledge that while reported claims have decreased, the underlying issues surrounding insurance fraud remain a concern, necessitating continued efforts to target and combat fraudulent behavior within the industry.

Insurance fraud is categorized into several dimensions, as outlined by Viaene et al. (2004): internal versus external, underwriting versus claim, and soft versus hard fraud. The classification of insurance fraud as either internal or external is based on whether the fraud is perpetrated within or outside an insurance company. Yusuf (2010) investigates two types of internal fraud occurring within insurance companies: one involves fraudulent actions perpetrated by the insurer themselves, while the other involves fraudulent activities carried out by employees of the company. Both types of fraud are committed by individuals working within the organization. According to Akomea et al. (2016), external fraud encompasses deceptive practices perpetrated by consumers or policyholders against insurers, referred to as policyholder fraud, as well as fraudulent activities carried out by independent brokers or agents against insurers, referred to as intermediary fraud. External fraud manifests in various forms, including policyholder or consumer fraud aimed at securing wrongful coverage or payment during the purchase of an insurance policy or the processing of claims (Derrig, 2002; Yusuf, 2010). Underwriting fraud entails deceitful practices occurring during the renewal of insurance contracts and coverage, while claim fraud involves the intentional submission of fictitious or false claims. Hard fraud occurs when someone deliberately plans or invents a loss, such as a collision, auto theft or fire that is covered by their insurance policy in order to claim payment for damages. Soft fraud is far more common and consists of otherwise legitimate claims that have been exaggerated or inflated, commonly referred to as "built-up" claims.

The detection of insurance fraud generally occurs in two steps. The first step is to identify suspicious claims that have a higher possibility of being fraudulent. This is often done by

referrals from claim adjusters or insurance agents. The next step is to refer these claims to investigators for further analysis. Because of the large volume of claims processed daily, it would be prohibitively costly for insurance firms to manually inspect each claim for signs of potential fraud. As a result, many companies opt to utilize computer systems and statistical methods to flag suspicious claims for closer examination. These statistical analysis tools fall into two categories: supervised and unsupervised. In both approaches, suspicious claims are pinpointed by comparing the claim data with anticipated norms.

In a supervised approach, anticipated values are determined through an analysis of records encompassing both fraudulent and non-fraudulent claims. However, according to Richard J. Bolton and David J. Hand (2002), this method comes with certain limitations. It necessitates absolute certainty regarding the classification of claims as either fraudulent or non-fraudulent, and it is constrained to detecting only those types of fraud that have been previously identified. In contrast, unsupervised statistical detection methods involve the identification of claims that deviate from the norm. Claims adjusters and computer systems can be trained to recognize "red flags" or indicators that have historically been associated with fraudulent claims. It's important to note that statistical detection does not confirm fraudulence; rather, it flags suspicious claims that warrant further investigation.

Efforts to mitigate the risk of fraudulent vehicle insurance claims have seen the implementation of various measures and technologies. Regulatory bodies have introduced stringent measures to curb fraudulent activities, including the requirement for thorough documentation and the verification of claims through standardized procedures. Use of telematics such as GPS and vehicle tracking systems have also become more prevalent to enhance the accuracy of assessing accident scenarios and validating claims. While these initiatives represent positive steps, the evolving nature of fraudulent activities necessitates continuous innovation and adaptation in the efforts to safeguard the integrity of the vehicle insurance sector in Kenya.

Machine learning algorithms have been effective in improving fraud detection across different domains. Through the development of supervised machine learning models, namely Logistic Regression, XGBoost and Random Forest, this research seeks to examine how these models utilize features extracted from datasets of vehicle insurance claims to assist in detecting fraudulent claims.

1.2 Statement of the Problem

Insurance fraud poses a significant challenge in the financial landscape, especially in the domain of vehicle insurance. The nature of vehicular claims makes this sector particularly susceptible to fraudulent activities, ranging from staged accidents and false damage claims to misrepresentations of accidents or vehicle conditions. According to the Association of Kenya Insurers (2021), fraudulent claims account for 25% of all insurance claims, with motor insurance fraud being most prevalent. It is estimated that 30% of motor insurance claims are fraudulent. With the rising importance of the automotive industry in Kenya and the corresponding increase in vehicular activities, the need for robust and effective insurance solutions becomes paramount.

The unique challenges posed by fraud within the vehicle insurance sector underscore the need for targeted, innovative solutions to ensure the integrity and sustainability of this critical aspect of the insurance industry. This study focuses on exploring how supervised machine learning algorithms (Logistic Regression, XGBoost and Random Forest) work in identifying fraudulent activity in the field of vehicle insurance, where traditional methods may need support from machine learning.

1.3 Objectives of the Study

1.3.1 General Objective

- To investigate how supervised machine learning algorithms can be used to identify fraudulent claims by utilizing features taken from datasets of auto insurance claims.

1.3.2 Specific Objectives

- To implement supervised machine learning models (Logistic Regression, XGBoost and Random Forest) to identify potentially fraudulent vehicle insurance claims.
- To evaluate performance of the models and identify the best performing model to ensure effectiveness in distinguishing between legitimate and fraudulent claims.
- To identify and investigate the key features influencing the model's decision-making process, providing insights into the factors that contribute to fraudulent claims.

1.4 Significance of the Study

This study delves into the implementation of advanced machine learning models (Logistic Regression, XGBoost and Random Forest) to strengthen fraud detection mechanisms within the insurance sector. By effectively identifying and mitigating false or fraudulent claims, insurers can curtail financial losses, paving the way for more sustainable and cost-effective operations. Heightened accuracy in detecting fraudulent activities not only fosters trust among genuine policyholders but also positions insurance companies at the forefront of technological innovation, boosting their competitiveness and attractiveness in the market. Efficient fraud detection also enables insurers to streamline resource allocation, redirecting focus towards legitimate claims processing and customer service enhancement. Through its contributions to the existing body of knowledge, this study provides valuable insights and methodologies for future research and practical applications in the realm of insurance fraud detection.

1.5 Scope and Limitation of the Study

Scope

This research focuses on developing and implementing supervised machine learning models, particularly Logistic Regression, XGBoost and Random Forest models, for fraud detection specifically in the context of vehicle insurance claims. The study will encompass data collection, preprocessing, model training, and evaluation phases, with an emphasis on enhancing the accuracy and efficiency of fraud detection processes within this domain.

Limitation

Constraints stemming from data availability and quality pose significant hurdles, impacting the efficacy of model implementation. The dynamic nature of fraud patterns presents additional complexities, necessitating adaptable strategies over time. Furthermore, external factors beyond the research scope, such as regulatory shifts and evolving fraudulent tactics, exert influence on model effectiveness.

Chapter 2

Literature Review

Cummins et al. (1996) investigated customers' attitudes toward fraudulent behavior to comprehend the root causes of insurance fraud. According to his study, ethical attitudes toward insurance fraud among individuals are influenced by their social and cultural surroundings, with people exhibiting greater tolerance when harboring negative perceptions of insurance institutions. Tennyson (2008) asserts that social attitudes toward insurance fraud shape individuals' beliefs and behavior regarding fraudulent activities, ultimately leading to mistrust in relationships within organizations. Piquero et al. (2005) in his study states that insurance fraud falls under the category of white-collar crimes, which are financially motivated offenses.

The Fraud triangle offers insights into the factors driving individuals to engage in fraud. It states that "individuals are motivated to commit fraud when three elements come together: some kind of perceived pressure, some perceived opportunity, and some way to rationalize the fraud as not being inconsistent with one's values. Akomea et al. (2016) suggests that fraud disrupts business operations, wastes time, tarnishes reputation, and leads to financial losses. According to, Soteriou et al. (1999) a common performance metric used in the insurance industry is financial performance, often assessed through profitability levels. Anjani et al. (2013) states that profitability is crucial for insurers' continuity and advancement serving as the primary source of funds. Anjani et al. (2013) furthers found that the escalation of insurance fraud results in higher premiums and strains the financial stability of insurance companies.

Kyung (2003) in his study to validate the effectiveness of data mining tools in detecting real-world insurance fraud underscored the importance of IT in fraud detection. Results indicated that tools like Analyst's Notebook not only identified fraudulent insurance transactions within numerous claims but also revealed connections among them, even exposing organized crime groups. In his work, Ngai et al. (2011) states the crucial role of machine learning techniques to extract valuable information and derive knowledge from extensive databases. He categorized six data mining application classes supported by various algorithmic approaches, including Classification, Clustering, Prediction, Outlier detection, Regression, and visualization. Ngai et al. (2011) investigated the application of data mining techniques in financial fraud detection across various sectors. The findings revealed a higher prevalence of financial fraud detection in the insurance sector compared to others.

Ahramovich (2023) says, "Machine learning-based fraud detection systems rely on ML algorithms that can be trained with historical data on past fraudulent or legitimate activities to autonomously identify the characteristic patterns of these events and recognize them once they recur." A study conducted by Blessie et al. (2011) explored the impact of feature selection techniques on improving classification performance. Imbalance between genuine and fraudulent claims poses a significant challenge in fraud detection using machine learning. Mazumder (2023) delved into techniques such as Synthetic Minority Over-sampling Technique (SMOTE) to address this issue and improve model performance. Halder (2023) explored various techniques for handling categorical variables in predictive modelling, their advantages and disadvantages, and best practices for using them.

Saini (2024) explored the logistic regression approach to predictive modelling, breaking down the basics and showcasing practical uses. Mathur (n.d.) demonstrated effective methods for logistic regression model optimization and parameter tuning in R using the caret package. Wilson (n.d.), has investigated the role of logistic regression in predicting the likelihood of fraudulent claims, offering insights into the significance of key features. Sruthi (2024) provided valuable insights into the workings of random forest algorithms. The avcontentteam (2021) provided practical guidance on implementing machine learning techniques in R using the mlr package. An article by Gondalia et al. (2020) extensively highlights Random Forest Classifier as a technique for modelling fraud detection in vehicle insurance Claims. Business

Science (2024) outlined a two-step process for tuning XGBoost hyperparameters, offering valuable insights into optimizing model performance.

Research by Zheng et al. (2023) presented a comprehensive understanding on predicting customer car insurance claims using Gradient Boosting Decision Tree (GBDT) and XGBoost models. Pandian (2022) has investigated hyperparameter tuning and its techniques, emphasizing the need of fine-tuning to achieve robust models for accurate results. Shin (2023) elaborated feature importance in machine learning, why it's so useful, its implementation and visualization. Brownlee (2019) illustrated how to finalize a machine learning model in R, including making predictions on unseen data, re-building the model from scratch and saving the model for later use.

By implementing machine learning models, this research aims to contribute to the enhancement of the insurance sector's ability to detect and deter vehicle insurance fraud, through exploration of the interpretability of Logistic Regression, XGBoost and Random Forest models and their predictive mechanisms.

Chapter 3

Research Methodology

3.1 Data Collection

In the initial stages of the project, efforts were made to obtain local datasets relevant to the domain of vehicle insurance in Kenya. However, due to challenges in finding comprehensive and suitable local data sources, an alternative approach was adopted. The decision was made to leverage publicly available datasets from Kaggle.com, a platform known for hosting a diverse range of datasets.

3.2 Data Description

The dataset utilized for this research encompasses details on policyholders, accident occurrences, and variables relevant to fraudulent claims, collected over a period of one year. The target is to detect if a claim application is fraudulent or not. It comprises of 15420 data points each containing 33 variables points.

The dataset underwent various preprocessing steps aimed at optimizing it for training of machine learning models. These include feature selection to identify pertinent variables, data splitting to ensure unbiased model evaluation, balancing to address class distribution disparities and handling missing values to enhance data quality.

Table 3.1 provides a description of the variables in the dataset.

Variable Name	Description
Month	Month in which an event related to insurance occurred.
WeekOfMonth	Week of the month in which an insurance-related event took place.
DayOfWeek	Day of the week when an event related to insurance occurred.
Make	Brand or manufacturer of the vehicle involved in the insurance event.
AccidentArea	Geographical area or location where the accident happened.
DayOfWeekClaimed	Day of the week when the insurance claim was reported.
MonthClaimed	Month in which the insurance claim was reported.
WeekOfMonthClaimed	Week of the month when the insurance claim was reported.
Sex	Gender of the policyholder.
MaritalStatus	Marital status of the policyholder.
Age	Age of the policyholder.
Fault	Indicates whether the policyholder was at fault in the insurance event.
PolicyType	Type or category of insurance policy.
VehicleCategory	Category to which the vehicle belongs.
VehiclePrice	Price range or value of the vehicle.
FraudFoundP	Binary variable indicating whether fraud was found (1) or not (0).
PolicyNumber	Unique identifier for each insurance policy.
RepNumber	Unique identifier for each insurance representative.
Deductible	Amount of money paid by the policyholder before an insurance claim is paid.
DriverRating	Rating of the driver's behavior or risk.
DaysPolicyAccident	Number of days between policy issuance and the accident.
DaysPolicyClaim	Number of days between policy issuance and reporting a claim.
PastNumberOfClaims	Number of previous insurance claims made by the policyholder.
AgeOfVehicle	Age of the insured vehicle.
AgeOfPolicyHolder	Age of the policyholder.
PoliceReportFiled	Whether a police report was filed (Yes/No).
WitnessPresent	Whether a witness was present during the insurance event (Yes/No).
AgentType	Type or category of the insurance agent.
NumberOfSuppliments	Number of supplemental claims filed.
AddressChangeClaim	Indicates whether the policyholder changed address after the claim (Yes/No).
NumberOfCars	Number of cars covered by the insurance policy.
Year	Year in which the data was recorded.
BasePolicy	Fundamental type of insurance policy.

Table 3.1: Description of the Vehicle Insurance Dataset

3.3 Model Development

Different types of machine learning models utilized for fraud detection include supervised and unsupervised algorithms. Supervised machine learning algorithms are trained on labeled data, where each data point is associated with a corresponding label or outcome. These algorithms learn patterns from the input data and their corresponding labels to make predictions on new, unseen data. Examples include logistic regression, decision trees, random forests, XGBoost and support vector machines. These methods are categorized into classification and regression algorithms. Classification aims to forecast categorical outcomes within a fixed set of values, while regression predicts real-valued numerical outputs. The task of detecting fraud is classified as a classification task as it involves predicting categorical outcomes, that is identifying whether a claim is fraudulent or legitimate, based on the input data and patterns learned from labeled examples.

Unsupervised machine learning algorithms operate on unlabeled data, aiming to identify hidden patterns or structures within the dataset without explicit guidance. These algorithms are particularly useful in detecting anomalies or outliers in the data, which could indicate potential fraudulent activity. Examples of unsupervised machine learning algorithms include K-Means Clustering, Hierarchical Clustering, Principal Component Analysis, Association Rule Learning, t-distributed Stochastic Neighbor Embedding(t-SNE) and Anomaly Detection.

The machine learning algorithms selected for this study were logistic regression, Random Forest, and XGBoost, aiming to leverage the interpretability of logistic regression, the ensemble learning capabilities of Random Forest, and the boosting algorithm's capacity to handle complex patterns, thus ensuring a multifaceted analysis with complementary strengths.

3.3.1 Logistic Regression

Logistic regression is a popular machine learning algorithm used in predicting a categorical dependent variable from a set of independent variables.

It utilizes a hypothesis function, cost function, and optimization method to make predictions. In logistic regression, the y-axis is confined to probabilities between 0 and 1.

The output of the linear combination of input features is transformed into probabilities using the logistic function that maps any real-valued number z to the range $[0, 1]$. The logistic function is defined as:

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{3.1}$$

The hypothesis function represents the probability that a given input example belongs to a certain class. It combines the linear combination of input features with the logistic function (sigmoid function) to output probabilities. It is formulated as the function:

$$h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}} \tag{3.2}$$

where:

- $h_\theta(x)$ represents the predicted probability that x belongs to the positive class,
- θ represents the parameters (coefficients) of the model,
- x represents the input features.

Figure 3.1 depicts the logistic regression function. Any output value that is greater than 0.5 is considered as 1, meaning there is a fraudulent claim, otherwise is 0, meaning no fraudulent claim.

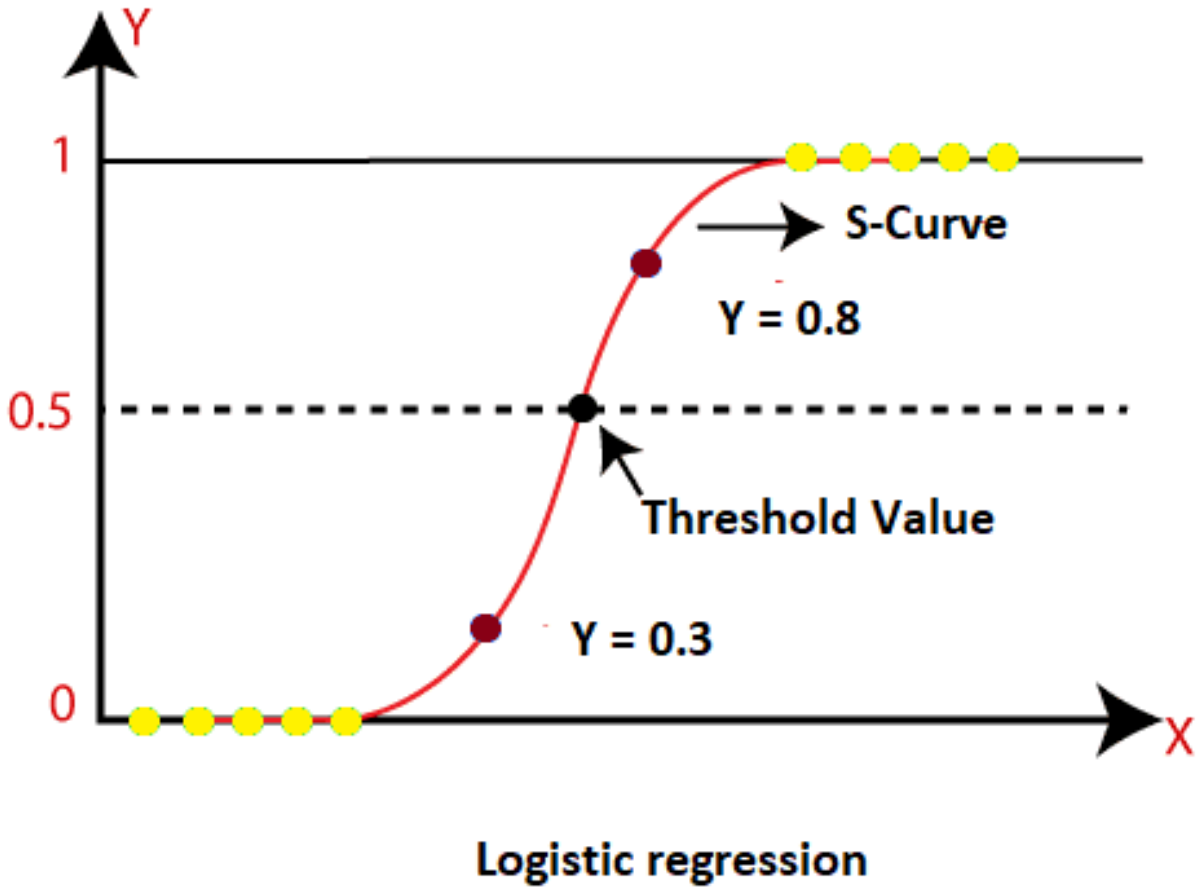

Figure 3.1: Logistic Regression Function

A cost function is used to train parameters so that the curve best fits a dataset for accurate predictions. The cost function measures the error between the predicted probabilities and the actual class labels. In logistic regression, the cost function is the logistic loss:

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[y^{(i)}\log(h_\theta(x^{(i)})) + (1-y^{(i)})\log(1-h_\theta(x^{(i)}))] \tag{3.3}$$

where:

- m is the number of training examples,
- $x^{(i)}$ represents the features of the i-th example,
- $y^{(i)}$ is the corresponding class label (0 or 1),
- $h_\theta(x^{(i)})$ is the predicted probability of the positive class for the i-th example.

The goal is to minimize the cost function by finding the optimal parameters θ. This is typically achieved using optimization algorithms like gradient descent, where the parameters are iteratively updated:

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta) \tag{3.4}$$

where:

- α is the learning rate,
- $\frac{\partial}{\partial \theta_j} J(\theta)$ is the partial derivative of the cost function with respect to θ_j.

3.3.2 Random Forest

Random Forest is an ensemble learning method that leverages the power of multiple decision trees to improve predictive performance and reduce overfitting. It employs bootstrap sampling to create multiple subsets of the training data, each subset of the same size as the original training dataset. A sample can be chosen more than once when creating the subsets.

Each bootstrap sample is then used to train an individual decision tree. This introduces diversity among the trees and helps reduce overfitting. At each node of the decision tree, Random Forest randomly selects a subset of features to consider for splitting. This technique helps decorrelate the trees and improve the generalization performance of the model. For classification tasks, Random Forest combines the predictions of individual trees through majority voting. Since duplicate entries are allowed in the subsets, some entries do not end up in the bootstrap samples and would therefore not be used in creating the trees they are not included in. These are called the Out-Of-Bag set.

Random Forest calculates the Out-Of-Bag error, which is the proportion of Out-Of-Bag samples that were incorrectly classified. The error serves as an estimate of the model's performance without the need for a separate validation set.

The formula for Out-of-Bag (OOB) error is generally represented as:

$$\text{OOB Error} = \frac{1}{n} \sum_{i=1}^{n} L(y_i, \hat{y}_i) \tag{3.5}$$

where:

- n is the total number of observations.
- y_i is the true response value for observation i.
- $\hat{y}_i$ is the predicted response value for observation i.
- L is the loss function used for measuring prediction error, which could be, for example, cross-entropy loss for classification problems.

However, it's important to note that the exact implementation may vary based on the specific software or library used for training the Random Forest model.

For classification in Random Forests, one commonly used loss function is the cross-entropy loss, also known as the log loss. The formula for the cross-entropy loss is:

$$L(y_i, \hat{y}_i) = -\sum_{j} y_{ij} \log(\hat{y}_{ij}) \tag{3.6}$$

where:

- y_{ij} is 1 if observation i belongs to class j, and 0 otherwise.
- $\hat{y}_{ij}$ is the predicted probability of observation i belonging to class j.

This loss function quantifies the difference between predicted probability distributions and true class labels, penalizing highly confident incorrect predictions more heavily.

Figure 3.2 depicts the Random Forest algorithm.

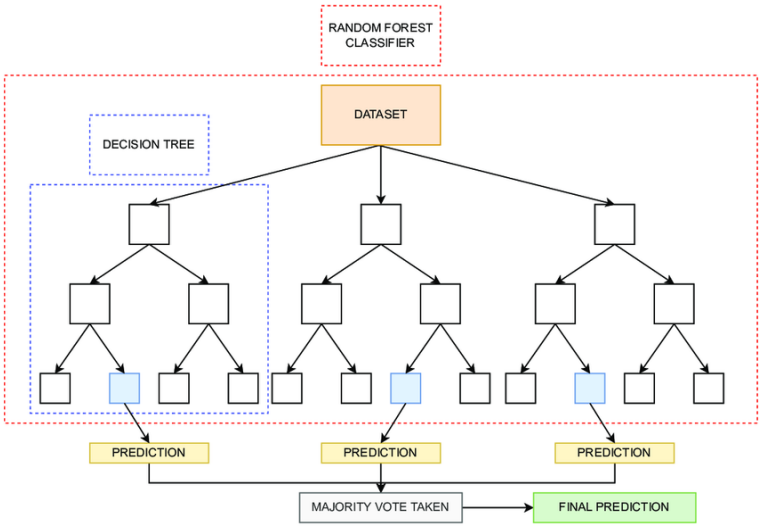

Figure 3.2: Random Forest Algorithm

3.3.3 XGBoost

XGBoost, also known as Extreme Gradient Boosting, is a machine learning algorithm that builds upon the gradient boosting framework, where weak learners, typically decision trees, are combined to form a robust learner. XGBoost starts with a simple model, usually a single leaf node, which serves as the initial prediction for all data points. This initial prediction is typically the mean of the target variable.

The initial residuals are calculated by subtracting the initial predictions from the true target values. These residuals represent the errors or discrepancies between the initial model's predictions and the actual target values. XGBoost iteratively builds decision trees to capture and correct the residuals of the previous iteration.

During tree building, the algorithm selects the best split point for each feature to minimize the objective function. The objective function is defined as:

$$\text{Objective} = \sum_{i=1}^{n} L(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k) \tag{3.7}$$

where:

- L is the loss function that measures the difference between the predicted and true target values for each data point.
- y_i is the true target value for the ith data point.
- $\hat{y}_i$ is the predicted value for the ith data point.
- $\Omega(f_k)$ represents the regularization term for the kth tree in the ensemble that penalizes complex models to prevent overfitting.

For binary classifications, XGBoost uses binary logistic loss function defined as:

$$\text{logloss} = -\frac{1}{n}\sum_{i=1}^{n}(y_i \log(p_i) + (1 - y_i)\log(1 - p_i)) \tag{3.8}$$

where:

- n is the number of samples,
- y_i is the actual label (0 or 1) for sample i,
- p_i is the predicted probability that sample i belongs to the positive class.

At each iteration, XGBoost explores potential splits for each feature and evaluates them using a metric such as gain. Gain measures the improvement in the objective function obtained by splitting the data at a particular point. It utilizes the first and second-order derivatives of the loss function with respect to the predicted values. These are called Gradients (g) and Hessians (h) respectively.

$$\text{Gradient} = g_i = \frac{\partial L(y_i, \hat{y}_i)}{\partial \hat{y}_i} \tag{3.9}$$

$$\text{Hessian} = h_i = \frac{\partial^2 L(y_i, \hat{y}_i)}{\partial \hat{y}_i^2} \tag{3.10}$$

Gain is calculated as:

$$\text{Gain} = \frac{1}{2}\left[\frac{(\sum g)^2}{\sum h + \lambda} + \gamma\right] - \frac{1}{2}\left[\frac{(\sum_{\text{left}} g)^2}{\sum_{\text{left}} h + \lambda} + \frac{(\sum_{\text{right}} g)^2}{\sum_{\text{right}} h + \lambda}\right] \tag{3.11}$$

where:

- $\sum g$ and $\sum h$ are the sum of gradients and Hessians respectively for all data points.
- $\sum_{\text{left}} g$, $\sum_{\text{left}} h$, $\sum_{\text{right}} g$, and $\sum_{\text{right}} h$ are the sums of gradients and Hessians for the left and right child nodes after the split.
- λ and γ are regularization parameters.

XGBoost continues building trees until a predefined stopping criterion is met, such as reaching the maximum number of trees or when further splits no longer lead to significant improvements in the objective function on a validation dataset. The final prediction is made by summing the predictions from all trees in the ensemble, optionally scaled by regularization parameters applied during training.

Figure 3.3 below showcases the XGBoost algorithm.

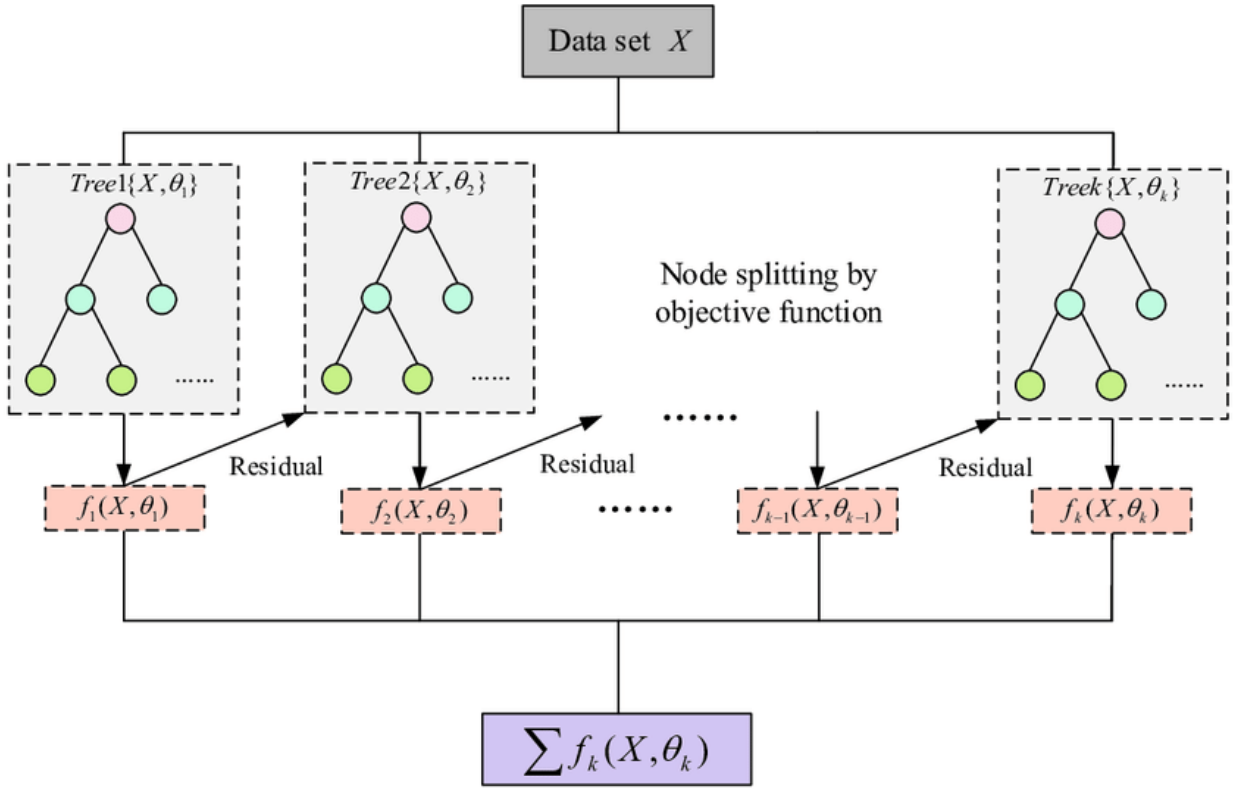

Figure 3.3: Flow Chart of XGBoost

3.3.4 Hyperparameter Tuning

Hyperparameters are configuration settings for machine learning algorithms that require tuning to optimize model performance. They are not learned from data but need to be set before training. Hyperparameter tuning involves finding the optimal values for these settings to improve the performance of the model.

In the training processes of logistic regression, hyperparameters were fine-tuned using cross-validation. This method involved partitioning the training dataset into 10 equal parts. Each subset is used as a validation set while the model is trained on the remaining data. This process repeats 10 times, with each subset used once as the validation set. This process aids in the selection of optimal hyperparameters and assessment of the model's generalization performance. The binomial family was also used to handle the binary classification task.

The hyperparameter tuning process for the random forest model was carried out to find the optimal number of trees and the number of variables considered at each split. We explored different parameter combinations to identify the best parameters to train the model.

The hyperparameters that control various aspects of the XGBoost model's learning process, such as the step size, tree complexity, regularization, and randomness, were tuned through techniques like grid search or randomized search to find the combination that yields the best model performance.

3.4 Evaluating model performance

Each of these models was trained on tha dataset to predict fraud occurrence. The outcomes were projected on test data and the models' performance evaluated by comparing the predicted values with the actual outcomes. This study employed a set of metrics to assess each model's effectiveness in predicting fraudulent claims within the dataset.

These metrics are:

Confusion Matrix

Confusion matrix is a table that summarizes the performance of a classification model by displaying counts of:

- **True Positive(TP)** - Number of instances correctly predicted as positive, that is fraudulent claims correctly predicted as fraudulent.
- **True Negative(TN)** - Number of instances correctly predicted as negative, that is non fraudulent claims correctly predicted as non fraudulent.
- **False Positive(FP)** - Number of instances incorrectly predicted as positive, that is non fraudulent claims incorrectly predicted as fraudulent.
- **False Negative(FN)** - Number of instances incorrectly predicted as negative, that is fraudulent claims incorrectly predicted as non fraudulent.

It is illustrated as Figure 3.4 below.

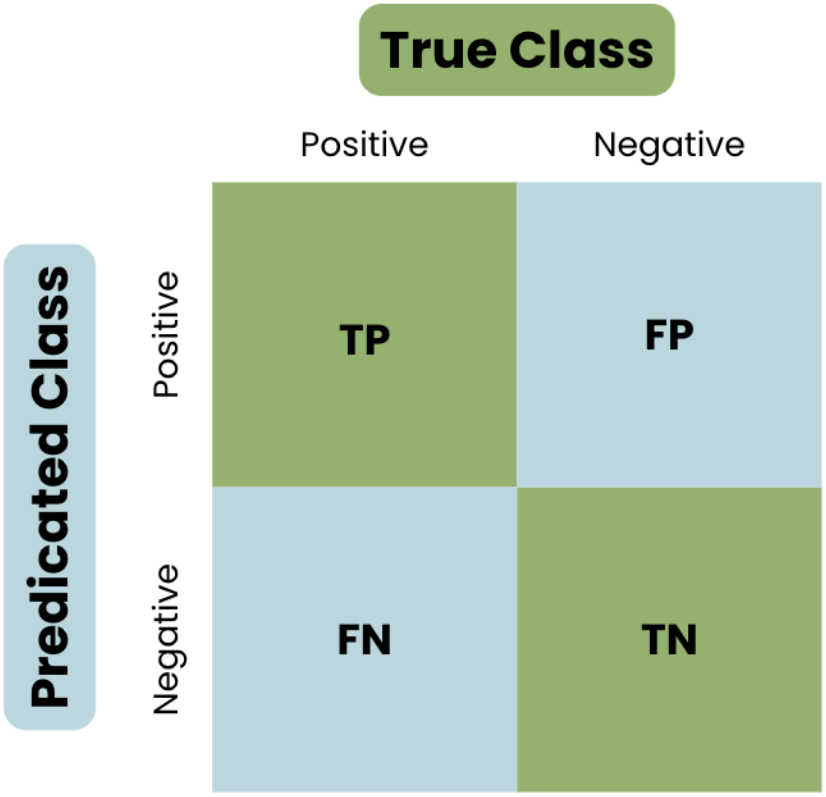

Figure 3.4: Confusion Matrix

Accuracy

Accuracy is defined as the proportion of correctly predicted instances to the proportion of total instances. Accuracy is at its best for a model when it is at 1.0 and at its worst at 0.0.

$$\text{Accuracy} = \frac{\text{True Positive} + \text{True Negative}}{\text{True Positive} + \text{True Negative} + \text{False Positive} + \text{False Negative}} \tag{3.12}$$

Precision

Precision quantifies the accuracy of positive predictions. A higher precision score indicates a lower rate of false positives, highlighting the model's effectiveness in making accurate positive predictions while minimizing incorrect positive classification.

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}} \tag{3.13}$$

Recall

Recall evaluates the model's ability to capture all actual positives. A higher recall indicates a model that is effective in identifying a larger proportion of true positive cases often at the expense of a potentially higher number of false positives.

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}} \tag{3.14}$$

F1 Score

F1 Score evaluates model's ability to achieve high accuracy. A higher F1 Score indicates a better balance between precision and recall.

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3.15}$$

The area under the ROC curve (AUC-ROC)

The ROC curve is a graphical representation of the trade-off between True Positive Rate (Sensitivity) and False Positive Rate (Specificity). True Positive Rate denotes the ratio of positive cases correctly identified by the model during predictions, divided by the total number of positive cases within the dataset. This metric is the same as recall. False Positive Rate represents the proportion of negative cases incorrectly identified as positive by the model during predictions, divided by the total number of negative cases within the dataset.

Figure 3.5 is an illustration of the ROC Curve.

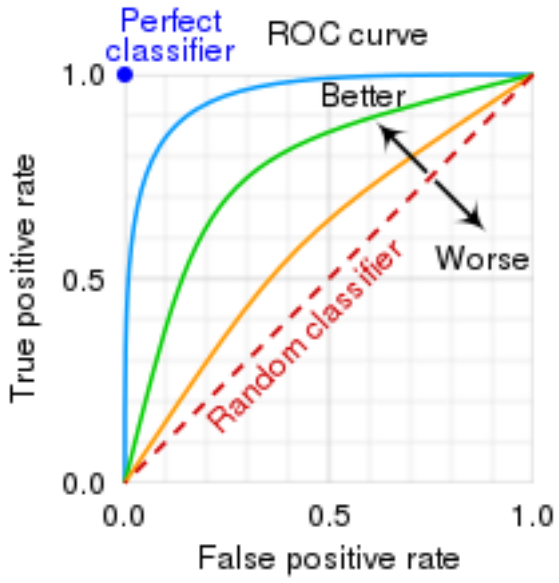

Figure 3.5: AUC-ROC Curve

The area under the ROC curve (AUC) measures performance of a model in terms of how much are correct and incorrect classifications. A higher AUC indicates better ability to distinguish between positive and negative instances.

3.5 Computational Framework

The research leveraged the R programming language, utilizing its extensive libraries and powerful statistical capabilities for data analysis and model development.

Chapter 4

Data Analysis and Model Training

4.1 Data Structure

Table 4.1 below showcases the characteristics of the dataset under analysis.

Structure	Count
Total Claims	15420
Variables	33
Categorical Variables	24
Integer Variables	9

Table 4.1: Dataset Structure

4.2 Data Transformation

It is the process of selecting, transforming or creating features that will be used as inputs to the machine learning algorithms.

4.2.1 Feature Selection

Feature selection is a critical process in machine learning and data analysis, involving the identification and selection of the most pertinent variables from a dataset to enhance model performance and mitigate the risk of overfitting. A correlation heatmap was employed as a visual aid during feature selection. A correlation heatmap displays the correlation coefficients between pairs of variables, typically represented as a grid of colored cells.

Figure 4.1 below is a correlation heatmap of the data variables. Positive correlations between variables are depicted using shades of blue, while negative correlations are represented by shades of red. The intensity of the color corresponds to the strength of the correlation; darker shades indicate stronger correlations, while lighter shades suggest weaker correlations.

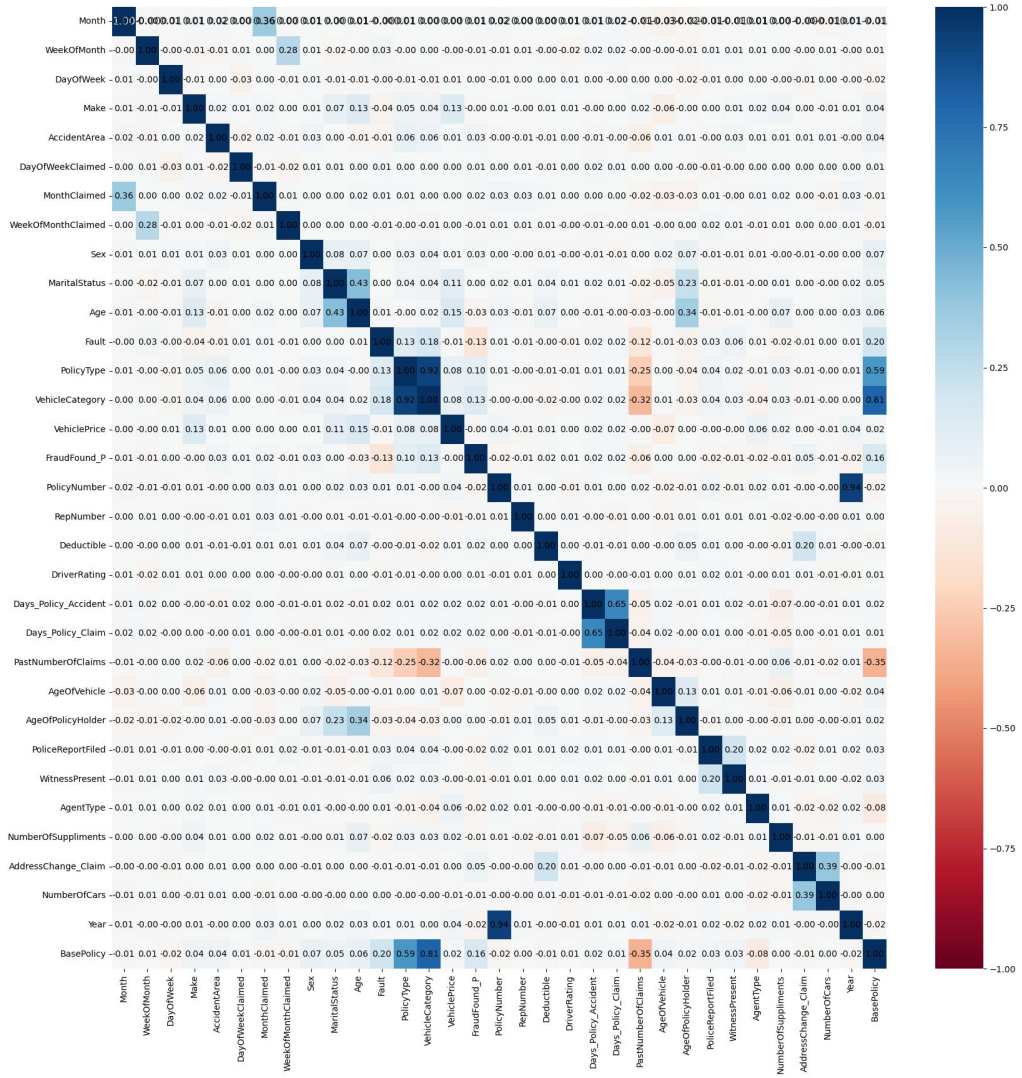

Figure 4.1: Correlation Heatmap of Data Variables

From analysis of Figure 4.1, highly correlated variables were identified and excluded from the study. A correlation threshold of 0.5 was chosen to strike a balance between eliminating highly correlated variables to mitigate multicollinearity while still retaining enough predictive power in the model. The following relationships were identified. Policy number and year are highly correlated (0.94) due to the temporal nature of policy issuance. Similary, base policy and vehicle category are highly correlated (0.81) because the type of vehicle typically determines the type of insurance coverage offered by the base policy. Days since policy accident and days since policy claim are highly correlated (0.65) due to their mutual dependency on the occurrence of an accident or claim within a policy period. Policy type and base policy are highly correlated (0.59) since the base policy often dictates the structure and coverage of policy types. The variable age contains the specific age of the policyholder. We opted to use the variable age of policyholder containing the age of the policyholder classified in groups. Age groups help generalize the model and reduce overfitting by grouping similar individuals together. Additionally, we eliminated the variables day of the week claimed and week of the month claimed, opting to use the month claimed, to further generalize the model. We eliminated the following variables:

```
##  [1] "WeekOfMonth"        "DayOfWeek"          "DayOfWeekClaimed"
##  [4] "WeekOfMonthClaimed" "Age"                "PolicyType"
##  [7] "VehicleCategory"    "PolicyNumber"       "RepNumber"
## [10] "Days_Policy_Claim"
```

23 out of the 33 features in the dataset were selected for this study. These were:

```
##  [1] "FraudFound_P"          "AgeOfPolicyHolder"    "Fault"
##  [4] "VehiclePrice"          "Days_Policy_Accident" "PastNumberOfClaims"
##  [7] "AgeOfVehicle"          "PoliceReportFiled"    "AgentType"
## [10] "NumberOfSuppliments"   "AddressChange_Claim"  "WitnessPresent"
## [13] "Sex"                   "AccidentArea"         "BasePolicy"
## [16] "MaritalStatus"         "DriverRating"         "NumberOfCars"
## [19] "Deductible"            "MonthClaimed"         "Make"
## [22] "Month"                 "Year"
```

4.2.2 Exploratory Data Analysis

EDA involved examining the distribution of fraudulent claims across the various features by employing visualizations to provide a quantitative overview.

Exploratory Analysis of the Dependent Variable

The dataset comprises of 14,497 non fraudulent claims and 923 fraudulent claims. Fraudulent claims comprised about 6% of the dataset, with genuine claims accounting 94%. Figure 4.2 illustrates the distribution of fraud within the dataset.

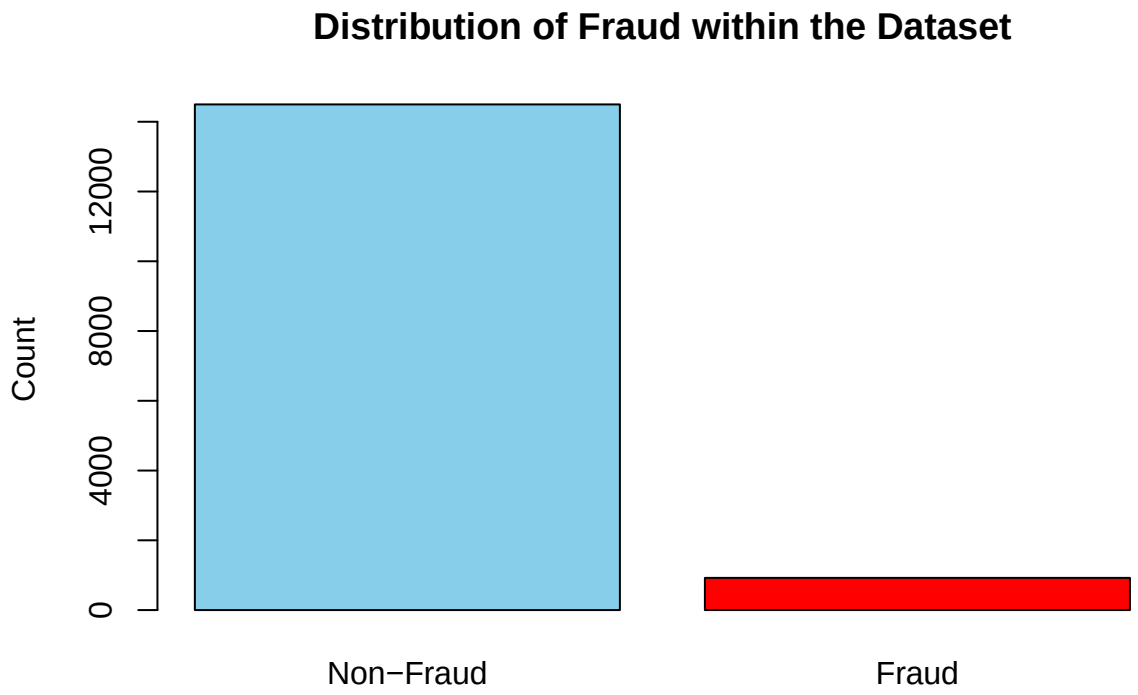

Figure 4.2: Distribution of Fraud within the Dataset

Exploratory Analysis of Explanatory Variables

Table 4.2 below provides a comprehensive summary of the counts of fraud for each unique feature within various variables in the dataset.

Table 4.2: Counts of Fraud by Variable Features

Variable	**Feature**	**Fraud Count**
WitnessPresent	No	920
	Yes	3
PoliceReportFiled	No	907
	Yes	16
AgentType	External	919
	Internal	4
AccidentArea	Rural	133
	Urban	790
Sex	Female	105
	Male	818
Fault	Policy Holder	886
	Third Party	37
BasePolicy	All Perils	452
	Collision	435
	Liability	36
PastNumberOfClaims	1	222
	2 to 4	294
	more than 4	68
	none	339
NumberOfSuppliments	1 to 2	159
	3 to 5	97
	more than 5	195
	none	472
MaritalStatus	Divorced	3
	Married	639
	Single	278
	Widow	3

Continued on next page

Table 4.2 – continued from previous page

Variable	**Feature**	**Fraud Count**
NumberOfCars	1 vehicle	850
	2 vehicles	43
	3 to 4	29
	5 to 8	1
	more than 8	0
Make	Accura	59
	BMW	1
	Chevrolet	94
	Dodge	2
	Ferrari	0
	Ford	33
	Honda	179
	Jaguar	0
	Lexus	0
	Mazda	123
	Mecedes	1
	Mercury	6
	Nisson	1
	Pontiac	213
	Porche	0
	Saab	11
	Saturn	6
	Toyota	186
	VW	8
AgeOfPolicyHolder	16 to 17	31
	18 to 20	2
	21 to 25	16
	26 to 30	33

Continued on next page

Table 4.2 – continued from previous page

Variable	**Feature**	**Fraud Count**
	31 to 35	360
	36 to 40	237
	41 to 50	144
	51 to 65	70
	over 65	30
DaysPolicyAccident	1 to 7	1
	8 to 15	5
	15 to 30	3
	more than 30	905
	none	9
VehiclePrice	20000 to 29000	421
	30000 to 39000	175
	40000 to 59000	31
	60000 to 69000	4
	less than 20000	103
	more than 69000	189
AddressChangeClaim	1 year	11
	2 to 3 years	51
	4 to 8 years	33
	no change	825
	under 6 months	3

From the findings summarized in Table 4.2, "No" overwhelmingly surpasses "Yes" in Witness Present and Police Report Filed, indicating a scarcity of witnesses and police involvement in fraudulent events. Similarly, external agents significantly outnumber internal ones, suggesting a potential vulnerability in external partnerships to fraudulent activities. Urban accidents surpass rural ones, possibly due to higher population density. Male policyholders significantly outnumber females, hinting at potential gender-specific fraud patterns. Policyholder fault prevails over third-party fault, indicating common instances of policyholder-initiated fraudulent claims. "All Perils" and "Collision" policies are most frequently associated with fraudulent claims. Additionally, policyholders with 2 to 4 past claims are more prone to fraudulent behavior, while most fraudulent claims avoid supplemental filings. Married individuals represent the largest marital status group among fraudsters, and single-vehicle policies are the most common targets for fraudulent activities. Chevrolet, Pontiac, and Toyota have notably higher counts compared to others, possibly indicating that the models are susceptible to fraudulent activities due to their popularity or market demand. Age brackets 31 to 35 and 36 to 40 exhibit the highest fraudulent claim counts among policyholders, suggesting a potential correlation between age demographics and fraudulent behavior. The majority of fraudulent claims occur when the accident is reported more than 30 days after the policy issuance, implying a delay tactic often employed by fraudsters to obfuscate the connection between the policy and the claimed incident. The analysis of fraudulent claims reveals intriguing patterns across different vehicle price ranges, with the 20000 to 29000 category exhibiting the highest count. Policyholders with no change in address after the claim show the highest count, suggesting a potential reluctance among fraudsters to alter personal information post-claim.

4.2.3 Data Splitting

Partitioning the dataset into training and testing sets enables a comprehensive evaluation of the model's performance on data it has not been exposed to during training, ensuring robustness and reliability in real-world applications. The data underwent a strategic partitioning process with 70% allocated for training the models and the remaining 30% reserved for testing.

10,794 records we selected for training the models while the remaining 4,626 records were reserved for testing the performance of the models.

4.2.4 Balancing Techniques

Balancing techniques in machine learning involve adjusting the distribution of classes in a dataset to prevent bias toward the majority class, ensuring fair model training and better performance on minority classes. This study utilized ROSE (Random Over-Sampling Examples), a method that uses Synthetic Minority Over-sampling Technique (SMOTE) to generate synthetic instances of the minority class by interpolating between existing minority class samples in feature space. Additionally, it randomly selects examples from the majority class. This increases the representation of the minority class and reduces that of the majority class until they reach a balanced proportion. Notice that there are more non-fraudulent claims than fraudulent ones in the dataset. See Figure 4.2. ROSE technique increases fraudulent claims in the dataset and reduces the dominance of genuine claims, creating a more balanced dataset, as illustrated by Figure 4.3. After balancing, the dataset comprised of 5,462 non fraudulent claims and 5,332 fraudulent claims.

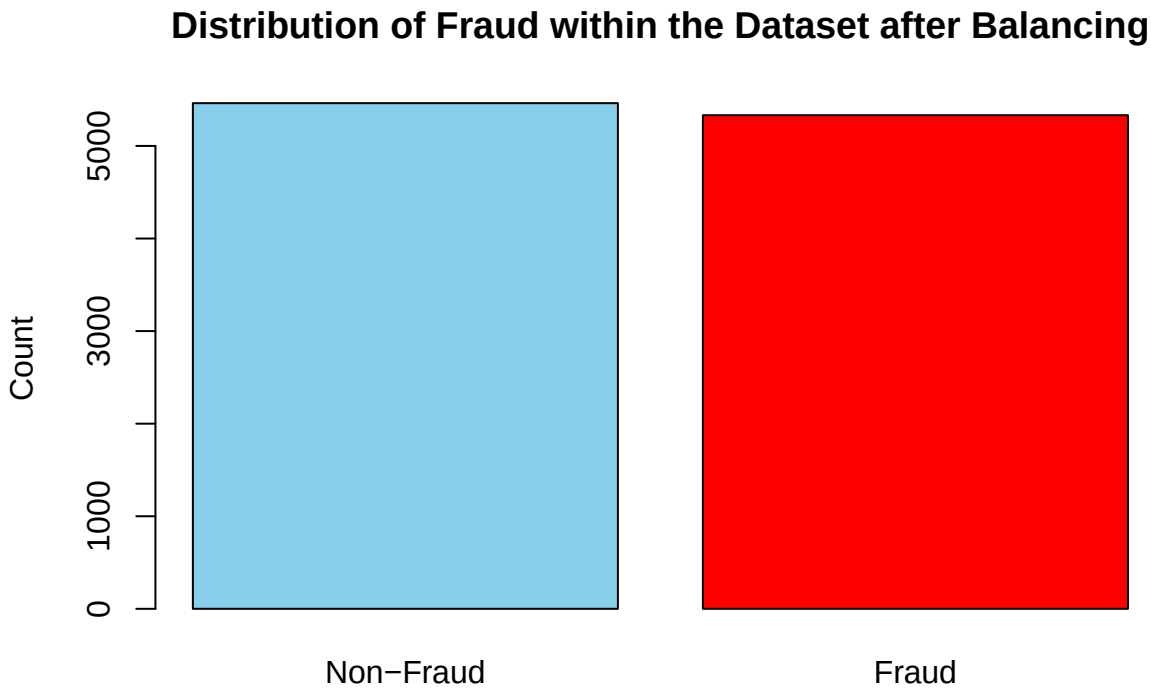

Figure 4.3: Distribution of Fraud within the Dataset after Balancing

4.2.5 Data Pre-processing

Missing and duplicated values were identified to ensure integrity of the dataset. Exploratory analysis revealed that there were no missing or duplicated values in the dataset.

Categorical variables often need to be encoded, transforming them into binary vectors to improve machine learning algorithms' understanding of categorical data. We identified the categorical columns as:

```
##  [1] "AgeOfPolicyHolder"    "Fault"                "VehiclePrice"
##  [4] "Days_Policy_Accident" "PastNumberOfClaims"   "AgeOfVehicle"
##  [7] "PoliceReportFiled"    "AgentType"            "NumberOfSuppliments"
## [10] "AddressChange_Claim"  "WitnessPresent"       "Sex"
## [13] "AccidentArea"         "BasePolicy"           "MaritalStatus"
## [16] "NumberOfCars"         "MonthClaimed"         "Make"
## [19] "Month"
```

We performed one-hot encoding to the categorical columns in both the training set and the testing set.

```
## Dimension of the Training Data after Encoding: 10794 94
```

```
## Dimension of the Testing Data after Encoding: 4626 94
```

Notice an increase in the number of columns. The encoding process introduces dummy variables as each category becomes its own separate feature.

4.3 Model Training and Performance Evaluation

The three machine learning algorithms were trained on the transformed data and their performance evaluated. Refer to Section 3.4. Table 4.3 summarizes the performance metrics of the models.

Model	Accuracy	Precision	Recall	F1 Score	TP	FP	FN	TN	AUC
Logistic Regression	65.26%	98.79%	63.83%	77.55%	2776	34	1573	243	0.76
XGBoost	76.16%	76.50%	97.62%	85.78%	3327	1022	81	196	0.82
Random Forest	87.89%	91.10%	95.82%	93.40%	3962	387	173	104	0.83

Table 4.3: Model Performance Metrics

Figure 4.4 displays the ROC Curves of the three trained models. See Figure 3.5.

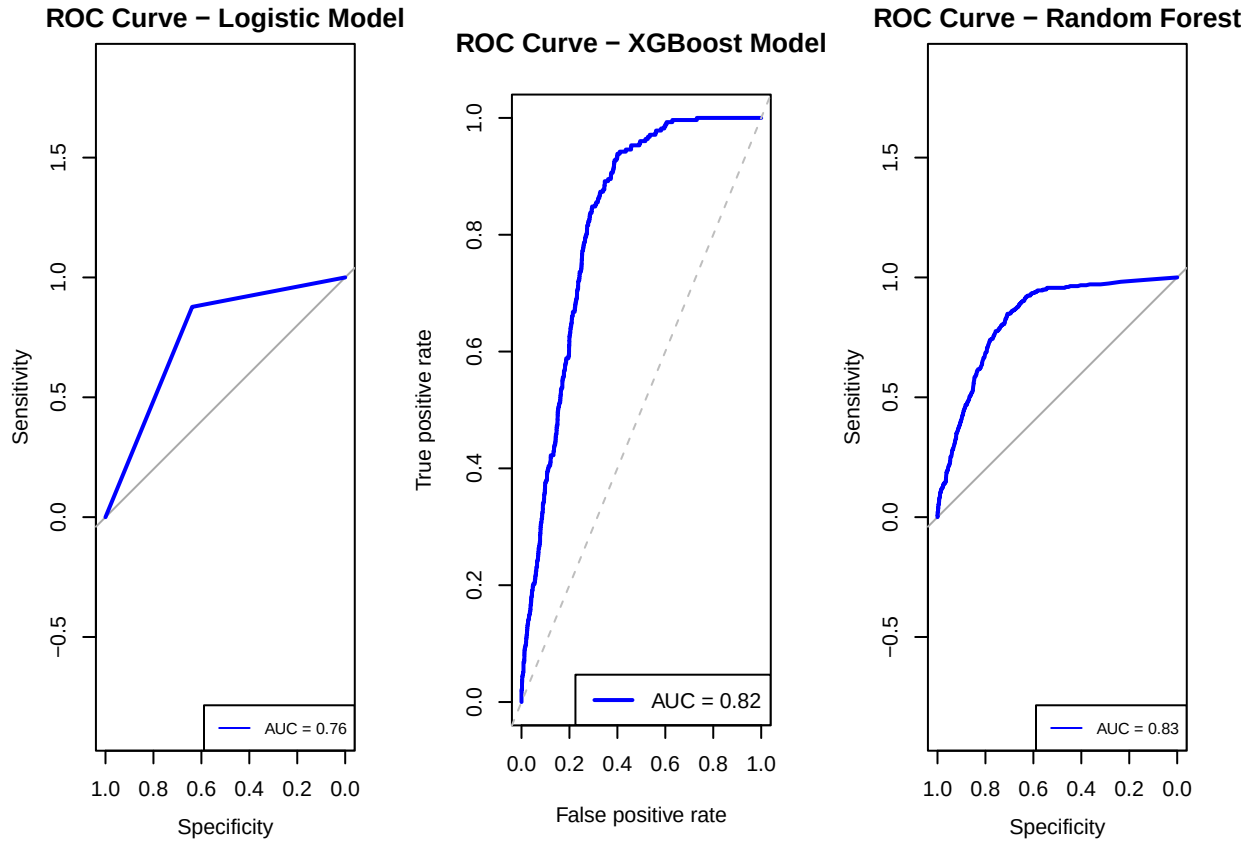

Figure 4.4: ROC Curves of the Trained Models

Among the models, Random Forest achieved the highest accuracy (87.89%) and F1 Score (93.40%), while XGBoost achieved the highest recall (97.62%). On the other hand, Logistic Regression had highest precision (98.79%), with the lowest accuracy (65.26%), recall (63.83%), and F1 Score (77.55%). With the highest AUC (0.83), the Random Forest Models exhibit the best ability to distinguish between positive and negative instances.

Considering the overall performance across multiple metrics, our research demonstrated that the Random Forest model exhibited superior performance, with an accuracy of 87.89%, precision of 91.10%, recall of 95.82%, F1 Score of 93.40% and AUC of 0.83.

4.4 Feature Importance

Feature importance quantifies the contribution of each input variable to the output of a machine learning model, aiding in understanding which features are most influential in making predictions. Feature importance provides insights into the factors that contribute to fraudulent claims. The top 10 features identified by the Random Forest model were:

```
##             BasePolicyLiability                 FaultThird.Party
##                       855.65788                        700.67518
##                      Deductible                             Year
##                       620.18243                        424.92118
##                    DriverRating AddressChange_Claim2.to.3.years
##                       330.05157                        120.78773
##             BasePolicyCollision          NumberOfSupplimentsnone
##                       67.05474                         61.49570
##                     MakeToyota         PastNumberOfClaims2.to.4
##                       61.44944                         59.50020
```

The numbers under each of the features represent their respective importance scores, quantifying their contribution to the Random Forest model's predictions regarding the probability of a claim being classified as fraudulent. It is important to note that since categorical variables were encoded, the features represent specific aspects of the insurance claims data. Higher values indicate greater influence on the model's decisions. For instance, features such as Base Policy - Liability and Fault - Third Party exhibit substantial importance scores, indicating their significant impact on prediction outcomes. Importance scores are utilized in feature selection during model training to identify the most influential features for predicting the target variable.

4.5 Deployment of the Model

Since the Random Forest model meets expectations and performs effectively, it was saved to disk for deployment for real-world applications. The model can be used to make predictions on fresh data.

Before making predictions, ensure that the new data undergoes the same preprocessing steps as the data used to train the model. Say there are 5 new records. You can leverage the Random Forest model to generate the probabilities of each of these records indicating fraudulent or non-fraudulent behavior. 0 signifies a genuine claim and 1 denotes a fraudulent claim. Specifically, the model assigns a probability score ranging from 0 to 1 for each record as illustrated below.

```
##           0           1
## 1 1.0000000 0.000000000
## 2 0.9960630 0.003937008
## 3 0.8149606 0.185039370
## 4 0.9803150 0.019685039
## 5 0.9724409 0.027559055
## attr(,"class")
## [1] "matrix" "array"  "votes"
```

Chapter 5

Conclusion and Recommendations

5.1 Introduction

The role of machine learning models becomes increasingly pivotal as practitioners delve deeper into detection of insurance fraud. Our study explores how these models can effectively combat fraudulent insurance claims within the auto insurance sector. By investigating the utilization of machine learning algorithms to identify fraudulent claims, we aim to shed light on their practical applications and implications for the insurance industry.

5.2 Summary of Findings

The modified data was used to train the three machine learning systems and their performances were assessed. Out of all the models, Random Forest achieved the highest accuracy (87.89%) and F1 Score (93.40%), while XGBoost had the best recall with a value of 97.62%. On the contrary, Logistic Regression had the highest precision at 98.79%, but recorded least scores on accuracy (65.26%), recall (63.83%), and F1 Score (77.55%). Random Forest also scored highly on Area Under the Curve (AUC) with a value of 0.83. In summary, our study established that Random Forest model outperformed other models as it attained an accuracy of 87.89%, precision of 91.10%, recall of 95.82%, F1 Score of 93.40% and an AUC of 0.83. These findings imply that, this model meets all necessary standards in terms of efficiency thus, can be relied upon for prediction in new datasets.

5.3 Study Conclusion

In conclusion, our study underscores the transformative potential of machine learning algorithms in combating fraudulent insurance claims within the auto insurance sector. Through analysis and experimentation, we have demonstrated the efficacy of machine learning technologies in identifying fraudulent patterns and mitigating financial losses for insurance companies. By harnessing the power of machine learning, insurers can fortify their defenses against the persistent threat of insurance fraud, preserve financial stability, and uphold the trust of policyholders in the integrity of the insurance industry.

5.4 Study Achievements

In the course of investigating how machine learning algorithms can be applied in the detection of fraud in auto insurance claims, this research has reached a number of significant milestones. We implemented and optimized models designed to identify fraudulent claims by using complex features from datasets to build strong mechanisms for detecting fraud. The study also provided the best model that could separate false from real claims having scrutinized the performance metrics such as accuracy, precision, recall and F1 score. As a result, it offers practical solutions to improve fraud detection efforts within the industry. Besides, through careful analysis and interpretation of feature importance the study identified important characteristics and patterns related to fraudulent behavior which provide indispensable insights into future researches as well as practice applications.

5.5 Study Limitations

Constraints in data availability and quality challenge our model implementation, potentially affecting robustness. The evolving nature of fraud necessitates adaptable strategies, as new tactics can undermine model effectiveness. Continuous monitoring and refinement are crucial to maintain relevance. Despite these limitations, our study provides valuable insights into applying machine learning algorithms for fraud detection in auto insurance, emphasizing the need for robust and adaptable models.

5.6 Study Recommendations

From the results of our study, several recommendations have been deduced to improve auto insurance fraud detection through machine learning. The insurance companies should focus on obtaining comprehensive datasets which are of high quality that would include detailed information about the insured, claim attributes and historical fraudulent cases. It is important to note that advanced feature engineering techniques as well as using different types of machine learning algorithms such as logistic regression, decision trees and neural networks are necessary. It is important that we continually evaluate and tune hyperparameters while monitoring model's performance. In addition, feedback from experts in a specific field and users will increase the transparency, interpretability and efficacy of fraud detection systems.

5.7 Future Work Suggestions

Future study should attain better data quality and availability, build flexible models for monitoring changing fraud techniques in real-time and incorporate external influences including regulatory modification into the area of car insurance. Improving the process of collecting data and ensuring that there is adequate information will make crime detection more effective. To address some complex problems and consider ethical and legal consequences linked to fairness and transparency in model development, interdisciplinary work needs to be promoted through cooperation between researchers, industry practitioners and regulators. This will stem from an intense effort towards examining fraudulent dealings that are expected to form a safer insurance environment with integrity.

References

[1] Viaene, S., et al. (2004). *Insurance Fraud: Issues and Challenges*. Geneva Pap Risk Insur Issues Pract 29, 313–333 (2004). See Link

[2] Derrig, R.A. (2002). *Insurance fraud*. Journal of Risk and Insurance, 69(3), 271–287. See Link

[3] Akomea, F., et al. (2016). *Causes, effects and deterrence of insurance fraud: evidence from Ghana*. Journal of Financial Crime, 23(4), 678–699. See Link

[4] Yusuf, T.O. (2010). *Brokers and the control of post contractual opportunism in the Nigerian Insurance market*. Journal of Financial Crime, 17 (2), 223-239. See Link

[5] Cummins, J.D., et al. (1996). *Moral hazard in insurance claiming: Evidence from automobile insurance*. Journal of Risk and Uncertainty, 12(1), 29–50. See Link

[6] Piquero, N.L. et al. (2005). *Integrating the desire for control and rational choice in a corporate crime context*. Justice Quarterly, 22, 252–280. See Link

[7] Soteriou, A., et al. (1999). *Operations, Quality, and Profitability in the Provision of Banking Services*. Management Science, 45, 1221–1238. See Link

[8] Anjani K., et al. (2013). *Life Insurance Fraud – Risk Management and Fraud Prevention*. International Journal of Marketing, Financial Services & Management Research, 2(5), 100–109. See Link

[9] Kyung S. (2003). *Detection of Insurance Fraud using Visualization Data Mining Tool*. Information Systems Review, 5(1), 49–60.

[10] Ngai E., et al. (2011). *The application of data mining techniques in financial fraud detection: A classification framework and an academic review of literature.* See Link

[11] Bolton, R.J., et al. (2002). *Statistical fraud detection: A review.* Statistical Science, 17(3), 235-249. See Link

[12] IRA. (2023). *Insurance industry report for the period January - March 2023: First quarter release.* See Link

[13] IRA. (2023). *Insurance industry report for the period April - June 2023: Second quarter release.* See Link

[14] IRA. (2023). *Insurance industry report for the period July - September 2023: Third quarter release.* See Link

[15] Ahramovich, A. (2023). *Machine learning for fraud detection: essentials, use cases, and guidelines.* See Link

[16] Blessie, E. et al. (2011). *Sigmis: A Feature Selection Algorithm Using Correlation Based Method.* See Link

[17] Mazumder, S. (2023). *5 Techniques to Handle Imbalanced Data For a Classification Problem.* See Link

[18] Halder, N. (2023). *Mastering categorical variables: Techniques and best practices for predictive modeling.* See Link

[19] Saini, A. (2024). *A Beginner's Guide to Logistic Regression.* See Link

[20] Mathur, S. (n.d.). *Tuning logistic regression using caret package.* See Link

[21] Holton, W. (n.d.). *An Analytical Approach To Detecting Insurance Fraud Using Logistic Regression.* See Link

[22] Sruthi, E. R. (2024). *Understand Random Forest Algorithms With Examples.* See Link

[23] avcontentteam (2021). *Practicing machine learning techniques in R with mlr package.* See Link

[24] Gondalia, D. et al. (2020). *Automobile Insurance Claim Fraud Detection using Random Forest and ADASYN.* See Link

[25] Business Science (2024). *XGBoost tuning: The hyperparameters, my secret 2-step process in R.* See Link

[26] Zheng, H. et al. (2023). *Insurance Fraud Detection Based on XGBoost.* See Link

[27] Pandian, S. (2022). *A Comprehensive Guide on Hyperparameter Tuning and its Techniques.* See Link

[28] Shin, T. (2023). *Understanding Feature Importance in Machine Learning.* See Link

[29] Brownlee, J. (2019). *Save And Finalize Your Machine Learning Model in R.* See Link

Appendix

```r
# MODEL TRAINING SOURCE CODE

data<-read.csv("fraud_oracle.csv", header = TRUE, sep = ',',
               stringsAsFactors = TRUE)
# Feature Selection
library(dplyr)
df<-select(data, FraudFound_P, AgeOfPolicyHolder, Fault, VehiclePrice,
           Days_Policy_Accident, PastNumberOfClaims, AgeOfVehicle,
           PoliceReportFiled, AgentType, NumberOfSuppliments,
           AddressChange_Claim, WitnessPresent, Sex, AccidentArea,
           BasePolicy, MaritalStatus, DriverRating, NumberOfCars,
           Deductible, MonthClaimed, Make, Month, Year)
# Data Splitting
library(caret)
library(caTools)
set.seed(123)
sample = sample.split(df$FraudFound_P, SplitRatio = .70)
train_data = subset(df, sample == TRUE)
test = subset(df, sample == FALSE)
# Data Balancing
library(ROSE)
traindata <- ROSE(FraudFound_P ~ ., data = train_data, p = 0.5, seed = 123)
train <- traindata$data
# Encoding Categorical Variables
categorical_columns <- sapply(train, is.factor)
categorical_column_names <- names(categorical_columns[categorical_columns])
non_categorical_column_names <- names(train)[!categorical_columns]
non_categorical_vars <- train[, non_categorical_column_names]
encoded_data <- model.matrix(~ . - 1, data = train[, categorical_columns])
train <- cbind(encoded_data, non_categorical_vars)
categorical_columns <- sapply(test, is.factor)
categorical_column_names <- names(categorical_columns[categorical_columns])
non_categorical_column_names <- names(test)[!categorical_columns]
non_categorical_vars <- test[, non_categorical_column_names]
encoded_data <- model.matrix(~ . - 1, data = test[, categorical_columns])
```

```r
test <- cbind(encoded_data, non_categorical_vars)
# Make Variables Names valid R Names
column_names <- names(train)
column_names <- make.names(column_names, unique = TRUE)
names(train) <- column_names
column_names <- names(test)
column_names <- make.names(column_names, unique = TRUE)
names(test) <- column_names
# Convert Target Variable to Factor Type for Modelling
train$FraudFound_P<-factor(train$FraudFound_P)
test$FraudFound_P<-factor(test$FraudFound_P)

#Train a Logistic Regression Model with Hyperparameter Tuning
ctrl <- trainControl(method = "cv", number = 10)
logistic_tuned <- train(
  FraudFound_P ~ .,
  data = train,
  method = "glm",
  trControl = ctrl,
  family = "binomial")
# Evaluate the Logistic Regression Model
logistic_predictions <- predict(logistic_tuned, newdata = test)
predicted_labels <- ifelse(logistic_predictions > 0.5, 1, 0)
glm_confusion_matrix <- table(logistic_predictions, test$FraudFound_P)
glm_confusion_matrix
predictions<-as.numeric(logistic_predictions)
glm_confusion_matrix <- table(logistic_predictions, test$FraudFound_P)
accuracy <- sum(diag(glm_confusion_matrix)) / sum(glm_confusion_matrix)
precision <- glm_confusion_matrix[1, 1] / sum(glm_confusion_matrix[1, ])
recall <- glm_confusion_matrix[1, 1] / sum(glm_confusion_matrix[, 1])
f1_score <- 2 * (precision * recall) / (precision + recall)
# Print Evaluation Metrics
cat("Accuracy:", accuracy, "\n")
cat("Precision:", precision, "\n")
cat("Recall:", recall, "\n")
cat("F1 Score:", f1_score, "\n")
# Plotting ROC Curve
library(pROC)
logistic_predictions<-as.numeric(logistic_predictions)
roc_curve <- roc(test$FraudFound_P, logistic_predictions)
auc_value <- auc(roc_curve)
plot(roc_curve, main = "ROC Curve - Logistic Model", col = "blue", lwd = 2)
auc_value <- auc(roc_curve)
legend("bottomright", legend = paste("AUC =", round(auc_value, 2)),
       col = "blue", lty = 1, cex = 0.8)
```

```r
# Training the xgboost
library(xgboost)
library(mltools)
library(Matrix)
library(caret)
library(dplyr)
trainm <- sparse.model.matrix(FraudFound_P ~ . - 1, data = train)
testm <- sparse.model.matrix(FraudFound_P ~ . - 1, data = test)
train_label <- as.integer(train[, "FraudFound_P"]) - 1
test_label <- as.integer(test[, "FraudFound_P"]) - 1
# Define the Parameter Grid for Hyperparameter Tuning
param_grid <- list(
  eta = c(0.01, 0.05, 0.1), # Learning rate
  max_depth = c(3, 6, 9),   # Maximum tree depth
  min_child_weight = c(1, 3, 5), # Minimum sum of instance weight needed
  gamma = c(0, 0.1, 0.3), # Minimum loss reduction required
  subsample = c(0.6, 0.8, 1), # Subsample ratio of the training instances
  colsample_bytree = c(0.6, 0.8, 1))# Subsample ratio of columns
tuned_model <- xgboost(
  data = trainm,
  label = train_label,
  nrounds = 1000, # Increase the number of boosting rounds
  objective = "binary:logistic", # Binary classification objective function
  eval_metric = "logloss",  # Evaluation metric
  verbose = 0, # Silent mode
  early_stopping_rounds = 10, # Early stopping rounds
  maximize = FALSE, # For logloss, we want to minimize
  params = param_grid) # Provide the parameter grid
# Evaluating XGBoost
predictions <- predict(tuned_model, testm)
predicted_labels <- ifelse(predictions > 0.5, 1, 0)
xg_confusion_matrix <- table(Actual = test_label, Predicted = predicted_labels)
accuracy <- sum(diag(xg_confusion_matrix)) / sum(xg_confusion_matrix)
precision <- xg_confusion_matrix[1, 1] / sum(xg_confusion_matrix[1, ])
recall <- xg_confusion_matrix[1, 1] / sum(xg_confusion_matrix[, 1])
f1_score <- 2 * (precision * recall) / (precision + recall)
# Print Evaluation Metrics
print(xg_confusion_matrix)
cat("Accuracy:", accuracy, "\n")
cat("Precision:", precision, "\n")
cat("Recall:", recall, "\n")
cat("F1 Score:", f1_score, "\n")
# Plotting ROC Curve
library(ROCR)
pred_obj <- prediction(predictions, test_label)
roc_xg <- performance(pred_obj, "tpr", "fpr")
```

```
plot(roc_xg, main = "ROC Curve - XGBoost Model", col = "blue", lwd = 2)
abline(a = 0, b = 1, col = 'gray', lty = 2)
library(pROC)
auc_value <- as.numeric(performance(pred_obj, "auc")@y.values)
legend("bottomright", legend = paste("AUC =", round(auc_value, 2)),
       col = 'blue', lwd = 2)

#Train the Random Forest Model
library(mlr)
library(randomForest)
train$FraudFound_P<-as.factor(as.character(train$FraudFound_P))
trainTask<-makeClassifTask(data = train, target = "FraudFound_P")
rf<-makeLearner("classif.randomForest",
                predict.type="response",
                par.vals=list(ntree = floor(0.1 * nrow(train)),
                              mtry = floor((ncol(train)-1)/3)))
rf$par.vals<-list(importance=TRUE)
rf_param<-makeParamSet(
  makeIntegerParam("ntree", lower=10, upper = floor(0.1*nrow(train))),
  makeIntegerParam("mtry", lower = floor((ncol(train)-1)/3),
                   upper = ncol(train)))
rancontrol<-makeTuneControlRandom(maxit = 10L)
set_cv<-makeResampleDesc("CV", iters=3L)
rf_tune<-tuneParams(learner = rf,
                    resampling=set_cv,
                    task=trainTask,
                    par.set=rf_param,
                    control = rancontrol,
                    measures = acc)
# Best Parameters for training
rf_tune$x$ntree
rf_tune$x$mtry
RF_Model<-randomForest(formula=FraudFound_P~.,
                       data = train,
                       ntree=rf_tune$x$ntree,
                       mtry=rf_tune$x$mtry,
                       replace=TRUE)
# Evaluating RF Model
Predict_RF_Test<-predict(object = RF_Model,
                         newdata = test,
                         type = "class")
CM_RF<-table(test$FraudFound_P, Predict_RF_Test)
accuracy <- sum(diag(CM_RF)) / sum(CM_RF)
precision <- CM_RF[1, 1] / sum(CM_RF[1, ])
recall <- CM_RF[1, 1] / sum(CM_RF[, 1])
f1_score <- 2 * (precision * recall) / (precision + recall)
```

```r
cat("Accuracy:", accuracy, "\n")
cat("Precision:", precision, "\n")
cat("Recall:", recall, "\n")
cat("F1 Score:", f1_score, "\n")
# Plotting ROC Curve
library(pROC)
rf_predicted_probs <- as.numeric(predict(RF_Model, newdata = test,
                                         type = 'prob')[,"1"])
roc_curve_rf <- roc(test$FraudFound_P, rf_predicted_probs)
auc_value_rf <- auc(roc_curve_rf)
plot(roc_curve_rf, main = "ROC Curve - Random Forest Model", col = "blue")
auc_value_rf <- auc(roc_curve_rf)
legend("bottomright", legend = paste("AUC =", round(auc_value_rf, 2)),
       col = "blue", lty = 1, cex = 0.8)
# Sort Feature Importance by Decreasing Importance
importance <- importance(RF_Model)
importance_sorted <- importance[order(-importance[, "MeanDecreaseGini"]),
                                drop = FALSE]
top_features <- importance_sorted[1:10, ]
print(top_features)
# Saving the Trained RF Model Object to a File
saveRDS(RF_Model, "Random_Forest_Model.rds")

# Making predictions on new records
new_records<-read.csv("newrecords.csv", header = TRUE, sep = ',',
                      stringsAsFactors = TRUE)
# Ensure new records undergo same pre-processing steps as training data
library(dplyr)
new_records<-select(new_records, FraudFound_P, AgeOfPolicyHolder, Fault,
                    VehiclePrice, Days_Policy_Accident, PastNumberOfClaims,
                    AgeOfVehicle, PoliceReportFiled, AgentType,
                    NumberOfSuppliments, AddressChange_Claim,
                    WitnessPresent, Sex, AccidentArea, BasePolicy,
                    MaritalStatus, DriverRating, NumberOfCars, Deductible,
                    MonthClaimed, Make, Month, Year)
new_records$FraudFound_P<-factor(new_records$FraudFound_P)
# Ensure categorical columns have same levels as the training data
categorical_columns <- sapply(new_records, is.factor)
categorical_column_names <- names(categorical_columns[categorical_columns])
for (col in categorical_column_names) {
  new_records[[col]] <- factor(new_records[[col]], levels = levels(train[[col]]))}
loaded_model <- readRDS("Random_Forest_Model.rds")
# Use the loaded model to make predictions on the new data
fraud_probabilities <- predict(loaded_model, newdata = new_records, type = "prob")
print(fraud_probabilities)
```

```r
# Training and Predicting on Datasets of Vehicle Insurance Claims using RF

# 1. PREPARE THE DATA ON VEHICLE INSURANCE CLAIMS

data2<-read.csv("insurance_claims.csv", header = TRUE, sep = ',',
                stringsAsFactors = TRUE)

# Clean, preprocess
# Checking for missing values
missing_count<-colSums(is.na(data2) | data2 == "")
# Removing variable with missing records
data2 <- data2[, colSums(is.na(data2)) == 0]
# Checking for duplicates
duplicated_count <- sum(duplicated(data2) | duplicated(data2,
                                                       fromLast = TRUE))

# Performing Feature Selection
# Dropping variables deemed redundant
drop_columns <- c('policy_state', 'policy_csl', 'incident_date',
                  'incident_state', 'incident_city',
                  'incident_location', 'policy_bind_date')
# Saving the dataframe with selected features to train model on
data2 <- data2[, !names(data2) %in% drop_columns]
# Ensure your response variable is binary (0/1)
# Replacing 'Y' with '1' and 'N' with '0' in the 'fraud_reported' column
data2$fraud_reported <- gsub('Y', '1', data2$fraud_reported)
data2$fraud_reported <- gsub('N', '0', data2$fraud_reported)
# Converting the 'fraud_reported' column from integer to factor
data2$fraud_reported <- as.factor(data2$fraud_reported)

# Performing Data Balancing
library(ROSE)
dataset <- ROSE(`fraud_reported` ~ `.`, data = data2, p = 0.5, seed = 123)

# Retrieving the balanced dataset for model training
# Saving the data in a variable called traindata
traindata <- dataset$data
```

```r
# 2. TRAIN THE RANDOM FOREST MODEL

set.seed(123)
library(mlr)
library(randomForest)
# Creating a classification task object using the train data.
TaskObj<-makeClassifTask(data = traindata, target = "fraud_reported")
# Creating learner for classification using classif.randomForest algorithm.
```

```
# Setting initial parameters for the number of trees (ntree)
# And the number of features to consider for each split (mtry).
# Importance computes variable importance
RF_learner<-makeLearner("classif.randomForest",
                predict.type="response",
                par.vals=list(ntree = floor(0.1 * nrow(traindata)),
                              mtry = floor((ncol(traindata)-1)/3),
                              importance = TRUE))
# Specifying range of values for tuning parameters ntree and mtry.
RF_param_set<-makeParamSet(
  makeIntegerParam("ntree", lower=10, upper = floor(0.1*nrow(traindata))),
  makeIntegerParam("mtry", lower = floor((ncol(traindata)-1)/3),
                   upper = ncol(traindata)))
# Creating a control object for tuning parameters
# Specifying random search with a maximum of 10 iterations.
controlObj<-makeTuneControlRandom(maxit = 10L)
# Creating a cross-validation resampling description with 3-fold cv.
cv_description<-makeResampleDesc("CV", iters=3L)
# Tuning the parameters of the rf learner using the specified settings.
# Optimizing accuracy (acc) as the performance measure.
RF_tuned<-tuneParams(learner = RF_learner,
    resampling=cv_description,
    task=TaskObj,
    par.set=RF_param_set,
    control = controlObj,
    measures = acc)

# Training the model, setting ntree and mtry based on the tuned parameters
# Allowing replacement of samples during bootstrap
rf_Model<-randomForest(formula=fraud_reported~.,
                       data = traindata,
                       ntree=RF_tuned$x$ntree,
                       mtry=RF_tuned$x$mtry,
                       replace=TRUE)

# Determining and sorting feature importance
importance <- importance(rf_Model)
importance_sorted <- importance[order(-importance[, "MeanDecreaseGini"]),
                                drop = FALSE]
#top_features <- importance_sorted[1:10, ]
#print(top_features)

# Save the trained model object to a file
saveRDS(rf_Model, "random_forest_model.rds")
```

```r
# 3. PREPARE NEW CLAIM APPLICATION RECORDS FOR PREDICTION

# Prepare the records in the same format as the data used to train the model.
# Records should contain values for all predictor variables used in the model.
# In the new data, we have 4 new records.
# Use the model to predict the likelihood of each record being fraudulent.

new_data<-read.csv("newdata.csv", header = TRUE, sep = ',',
                   stringsAsFactors = TRUE)
missing_count<-colSums(is.na(new_data) | new_data == "")
new_data <- new_data[, colSums(is.na(new_data)) == 0]
drop_columns <- c('policy_state', 'policy_csl','incident_date',
                  'incident_state', 'incident_city',
                  'incident_location', 'policy_bind_date')
new_data <- new_data[, !names(new_data) %in% drop_columns]
new_data$fraud_reported <- gsub('Y', '1', new_data$fraud_reported)
new_data$fraud_reported <- gsub('N', '0', new_data$fraud_reported)
new_data$fraud_reported <- as.factor(new_data$fraud_reported)
# Ensure categorical columns have same levels as data used in training
categorical_columns <- sapply(new_data, is.factor)
categorical_column_names <- names(categorical_columns[categorical_columns])
for (col in categorical_column_names) {
  new_data[[col]] <- factor(new_data[[col]], levels = levels(traindata[[col]]))
}

# New records are now in the same format as data used during model training
```

```r
# 4. MAKE FRAUD DETECTIONS

# Load the saved model into memory
loaded_model <- readRDS("random_forest_model.rds")
# Use the loaded model to make predictions on the new data
fraud_probabilities <- predict(loaded_model, newdata = new_data, type = "prob")
# Output the predicted probabilities of fraud associated with each record
# 0 for genuine claim and 1 for fraudulent
print(fraud_probabilities)
```

```
##           0         1
## 1 0.33333333 0.66666667
## 2 0.96666667 0.03333333
## 3 0.93333333 0.06666667
## 4 0.03333333 0.96666667
## attr(,"class")
## [1] "matrix" "array"  "votes"
```