



Analysing fuel transactions of government vehicles in the Eastern Cape, South Africa

by

Jared Tavares

A research project
presented to the University of Cape Town
in partial fulfillment of the
requirements for the Masters of Science
Statistics and Data Science degree

Under the supervision of
Prof Sheetal Silal

December 1, 2024

The copyright of this thesis vests in the author. No quotation from it or information derived from it is to be published without full acknowledgement of the source. The thesis is to be used for private study or non-commercial research purposes only.

Published by the University of Cape Town (UCT) in terms of the non-exclusive license granted to UCT by the author.

Authors' Declaration

I, Jared Tavares, confirm that this project was written solely by myself. Additionally, I confirm that:

- Any contributions or references used in this project from other people's work were correctly acknowledged, cited and referenced.
- The APA referencing system was used for any and all citations and referencing.
- This project is my own work.
- This project was completed at the University of Cape Town as part of the requirements for a Statistical Sciences Masters degree in Statistics and Data Science.

Name: Jared Tavares

Date: December 1, 2024

Signed by candidate

Abstract

Fuel management and fraud detection in government fleets are critical issues that have far-reaching financial and operational implications. To address these challenges, an investigation of fuel usage patterns and anomalies in the Eastern Cape Province government fleet in South Africa from April 2021 to January 2022 was conducted. Through the application of exploratory data analysis, clustering techniques, and predictive modelling, the research uncovers valuable insights that can be used to optimise fuel consumption and detect fraudulent activities within the fleet.

Univariate and bivariate analyses reveal distinct patterns in fleet composition, transaction volumes, and fuel efficiency across various vehicle makes, model derivatives, and departments. The use of clustering techniques enables the identification of distinct vehicle segments and transaction patterns, emphasising the importance of considering contextual factors when analysing fuel usage.

To detect potential fraud, three key indicators are developed: abnormally large transactions, frequent transactions, and fuel price differences. Predictive models, including XGBoost, Multi-layer Perceptron, and Random Forest, are employed to automate the classification of transactions based on these fraud indicators. The Multi-layer Perceptron demonstrates the best performance, achieving an accuracy of 87% on the test set.

The dissertation acknowledges limitations due to the scope of the data and missing information for certain geographic variables such as district and site. Future research could expand the geographical and temporal range, incorporate qualitative data, explore real-time monitoring systems, and investigate vehicle maintenance and fuel efficiency.

The present research makes a noteworthy contribution to the knowledge of fuel management and fraud detection in government fleets by offering a data-driven approach to expose inefficiencies and anomalies. The insights and methodologies presented serve as a foundation for future research and practical applications, ultimately leading to more efficient, cost-effective, and transparent fleet operations.

Acknowledgments

I would like to express my deepest gratitude to God for granting me this incredible opportunity as well as several individuals who have substantially contributed to the completion of this dissertation.

First and foremost, I extend my sincere thanks to my supervisor, Prof Sheetal Silal, for her invaluable guidance, support, and insights throughout this research journey. Her expertise and encouragement have been instrumental in shaping this dissertation.

I am also grateful to the Statistical Sciences Department at the University of Cape Town for providing me with the necessary resources and a conducive environment to pursue my research. My heartfelt appreciation goes to my family and friends for their unwavering support and understanding during this demanding period. Their constant encouragement and belief in me have been a source of strength and motivation.

Finally, I would like to acknowledge Nikhil Khanna from the Clinton Health Access Initiative for granting access to the fuel transaction data, without which this study would not have been possible. I am thankful for their cooperation and trust in this research endeavor.

Contents

Authors' Declaration	ii
Abstract	iii
Acknowledgments	iv
List of Figures	ix
List of Tables	xii
1 Introduction	1
1.1 Research Aims and Objectives	2
1.2 Dissertation Structure	2
1.3 Code Repository	2
2 Literature Review	3
2.1 Fuel Theft: Prevalent Mechanisms and Their Dynamics	3
2.1.1 Categorising Fuel Theft Strategies	3
2.2 Technological Tools to Combat Fuel Fraud	4
2.3 Model-Based Solutions for Fuel Theft Detection	5
2.3.1 Characteristics of an Effective Fraud Detection System	5
2.3.2 Approaches to Addressing Challenges in Fraud Detection Modelling	5
2.3.3 A Review of Transaction Fraud Detection Models	6
2.4 The Context of the Eastern Cape	8
3 Methods	9
3.1 k-means++ Clustering	9
3.1.1 Introduction	9
3.1.2 Underlying Assumptions	9
3.1.3 Algorithm and Mechanism	10
3.1.4 Possible Implications for Future Research	11
3.1.5 Python Implementation of k-means++	11
3.2 Agglomerative Hierarchical Clustering	12
3.2.1 Introduction	12

3.2.2	Underlying Assumptions	12
3.2.3	Algorithm and Mechanism	13
3.2.4	Python Implementation of Agglomerative Clustering	13
3.3	Guassian Mixture Models (GMM) Clustering	14
3.3.1	Introduction	14
3.3.2	Underlying Assumptions	14
3.3.3	Algorithm and Mechanism	14
3.3.4	Python Implementation GMM Clustering	16
3.4	BIRCH Clustering	17
3.4.1	Introduction	17
3.4.2	Underlying Assumptions	17
3.4.3	Algorithm and Mechanism	17
3.4.4	CF-Tree	18
3.4.5	Python Implementation of BIRCH Clustering	19
3.5	Modelling Methods Overview	20
3.6	XGBoost Model	21
3.6.1	Introduction	21
3.6.2	Underlying Assumptions	21
3.6.3	Algorithm and Mechanism	21
3.6.4	Python Implementation of XGBoost Model	23
3.7	Random Forest	24
3.7.1	Introduction	24
3.7.2	Underlying Assumptions	24
3.7.3	Algorithm and Mechanism	24
3.7.4	Python Implementation of Random Forest Model	25
3.8	Multi-layer Perceptron Model	26
3.8.1	Introduction	26
3.8.2	Underlying Assumptions	26
3.8.3	Algorithm and Mechanism	26
3.8.4	Python Implementation of MLP Model	27
3.9	Model Evaluation Methods	28
3.9.1	Precision	28
3.9.2	Recall	28
3.9.3	F1-score	28
3.9.4	Accuracy	29
3.9.5	Support	29
4	Data Preparation	30
4.1	Combining the Raw Datasets	31
4.2	Data Cleaning	33
4.3	Adding Transaction Coordinates	33
4.4	Adding Government-Mandated Fuel Prices	34
4.5	Creating a Vehicle-Level Dataset	34
4.6	Breakdown of the Final Transaction Dataset	36
4.7	Breakdown of the Final Vehicle Dataset	38

5	Results: Data Exploration	40
5.1	Exploratory Data Analysis (EDA)	40
5.1.1	Univariate EDA	41
5.1.2	Bivariate EDA	51
5.2	Clustering	61
5.2.1	Vehicle Clustering	61
5.2.2	Clustering Transactions	70
6	Results: Predictive Modelling	79
6.1	Indicators	79
6.1.1	Abnormally Large Transactions Indicator	79
6.1.2	Frequent Transactions Indicator	80
6.1.3	Fuel Price Difference Flag	80
6.1.4	Combining the Flags	81
6.2	Predictive Models	85
6.2.1	Overall Results	85
6.2.2	XGBoost	87
6.2.3	Multi-layer Perceptron (MLP)	89
6.2.4	Random Forest	91
7	Discussion and Conclusion	93
	References	98
	APPENDICES	102
A	Other Modelling Methods	103
A.1	Naive Bayes Model	103
A.1.1	Introduction	103
A.1.2	Underlying Assumptions	103
A.1.3	Algorithm and Mechanism	104
A.1.4	Python Implementation	105
A.2	Linear Support Vector Machine Model	106
A.2.1	Introduction	106
A.2.2	Underlying Assumptions	106
A.2.3	Algorithm and Mechanism	106
A.2.4	Python Implementation	107
A.3	Stochastic Gradient Descent Classifier	108
A.3.1	Introduction	108
A.3.2	Underlying Assumptions	108
A.3.3	Algorithm and Mechanism	108
A.3.4	Python Implementation	109
B	Other Predictive Modelling Results	110
B.1	Linear SVM	110
B.2	Naive Bayes	114

B.3 SGD Classifier 117

List of Figures

4.1	A diagram of the high-level data preparation process.	31
4.2	A screenshot of the Google Cloud console displaying the activated Google Maps API key to convert site and merchant names to coordinates.	33
4.3	A screenshot of the Google BigQuery dataset size report for the full GPS tracker dataset.	34
4.4	A screenshot of the Google BigQuery dataset preview for the GPS tracker dataset.	35
4.5	A screenshot of the Google BigQuery query used to get the final odometer reading at the end of each day for each vehicle.	35
5.1	Bar plots of the top counts per make for the a) fleet and b) transactions.	41
5.2	Bar plots of the top counts per vehicle derivative for the a) fleet and b) transactions.	42
5.3	Bar plots of the top counts per rate card category for the a) fleet and b) transactions.	43
5.4	Bar plots of the top counts per department for the a) fleet and b) transactions.	44
5.5	Bar plots of the top transaction counts over the a) weekdays and b) months.	45
5.6	Bar plots of the number of vehicles per district and site.	46
5.7	Bar plots of the top transaction counts over the a) merchants and b) fuel types.	47
5.8	Histograms of the a) days between transactions grouped by vehicles for the transaction dataset and b) average days between transactions for each vehicle.	48
5.9	Histograms of the a) transaction amounts, b) number of litres for all transactions and c) kilometers per litre for all vehicles.	50
5.10	Bar plots of the average transaction amount and average number of litres for each vehicle make.	51
5.11	Bar plots of the average transaction amount and average number of litres for each model derivative.	52
5.12	Bar plots of the average transaction amount and average number of litres for each rate card category.	53
5.13	Bar plots of the average transaction amount and average number of litres over different subsets of days between refuelling with the number of transactions in each category above each bar.	54
5.14	Box plots displaying the distribution of the 2-day rolling average number of litres (the average number of litres purchased for every two consecutive purchases per vehicle) over different subsets of days between refuelling.	55
5.15	Bar plots of the average transaction amount and average number of litres for each month.	56
5.16	Bar plots of the average transaction amount and average number of litres for each weekday.	56

5.17	Box plots of the kilometres per litres for the top a) model derivatives and b) rate card categories.	57
5.18	Map of transaction locations within the Eastern Cape.	59
5.19	Map of transaction locations within South Africa.	60
5.20	(Left) A line plot of within-cluster sum of squares over clusters to determine the optimal number of clusters for the k-means++ algorithm. (Right) A dendrogram to determine the number of clusters for agglomerative clustering.	61
5.21	Bar plots for the number of vehicles in each cluster for a) k-means++ and b) agglomerative clustering.	62
5.22	Box plots of the average transaction amount across clusters for a) k-means++ and b) agglomerative clustering.	63
5.23	Box plots of the average number of litres across clusters for a) k-means++ and b) agglomerative clustering.	64
5.24	Stacked bar plots for the compositions of the k-means++ clusters across: a) model derivative, b) department, c) district and d) rate card category.	66
5.25	Stacked bar plots for the compositions of the agglomerative clusters across: a) model derivative, b) department, c) district and d) rate card category.	68
5.26	(Left) A line plot of the within cluster sum of squares (WCSS) over number of clusters to determine the number of clusters to use for the k-means++ algorithm. (Right) A line plot of the silhouette score over the number of clusters to determine the optimal number for the BIRCH algorithm.	70
5.27	Bar plots of the transactions per cluster for a) k-means++, b) BIRCH and c) GMM.	71
5.28	Box plots of the transaction amount per cluster for a) k-means++, b) BIRCH and c) GMM.	72
5.29	Box plots of the number of litres per cluster for a) k-means++, b) BIRCH and c) GMM.	73
5.30	Stacked bar plots for the compositions of the k-means++ clusters across: a) model derivative, b) department, c) district and d) rate card category.	74
5.31	Stacked bar plots for the compositions of the BIRCH clusters across: a) model derivative, b) department, c) district and d) rate card category.	75
5.32	Stacked bar plots for the compositions of the GMM clusters across: a) model derivative, b) department, c) district and d) rate card category.	77
6.1	Barplot of the counts across number of indicators (flags).	81
6.2	Box plots of the a) transaction amounts and b) number of litres across the number of indicators (flags).	82
6.3	Stacked bar plots for the compositions of the transactions with different numbers of flags across: a) model derivative, b) department, c) district and d) rate card category.	84
6.4	(Left) The confusion matrix with percentages of total observations for the XGBoost model on the test set. (Right) The ROC curves for the XGBoost model for each class on the test set using a one-vs-all approach, where the class of interest is treated as the positive class, and all other classes are combined to form the negative class.	87

6.5	(Left) The confusion matrix with percentages of total observations for the Multi-layer Perceptron model on the test set. (Right) The ROC curves for the Multi-layer Perceptron model for each class on the test set using a one-vs-all approach, where the class of interest is treated as the positive class, and all other classes are combined to form the negative class.	89
6.6	(Left) The confusion matrix with percentages of total observations for the Random Forest model on the test set. (Right) The ROC curves for the Random Forest model for each class on the test set using a one-vs-all approach, where the class of interest is treated as the positive class, and all other classes are combined to form the negative class.	91
B.1	The confusion matrix with percentages of total observations for the Linear SVM model on the test set.	111
B.2	The ROC curves for the Linear SVM model for each class on the test set using a one-vs-all approach, where the class of interest is treated as the positive class, and all other classes are combined to form the negative class.	112
B.3	The confusion matrix with percentages of total observations for the Naive Bayes model on the test set.	114
B.4	The ROC curves for the Naive Bayes model for each class on the test set using a one-vs-all approach, where the class of interest is treated as the positive class, and all other classes are combined to form the negative class.	115
B.5	The confusion matrix with percentages of total observations for the SGD Classifier model on the test set.	118
B.6	The ROC curves for SGD Classifier for each class on the test set using a one-vs-all approach, where the class of interest is treated as the positive class, and all other classes are combined to form the negative class.	119

List of Tables

3.1	Comparison of Modelling Methods	20
4.1	The first 5 transactions from the raw transactions dataset (with anonymised registration numbers).	32
4.2	The first five entries from the raw fleet register dataset (with anonymised registration numbers).	32
4.3	The first 5 entries from the raw EMS and White fleet register dataset (with anonymised registration numbers).	32
4.4	The first 5 rows from the anonymised final fuel transaction dataset.	36
4.5	The first 5 rows from the final anonymised vehicle-level aggregated dataset.	38
6.1	Diesel prices from various sources on 1 March 2024	80
6.2	Count of transactions flagged per each indicator	81
6.3	Comparison of classification reports for different models on the test set (20% of the observations).	85

Chapter 1

Introduction

Fleet management includes many complex tasks, such as vehicle allocation, rental management, and usage planning. Among these, optimising fuel efficiency and identifying fuel transaction fraud presents a considerable challenge for all fleets, including the government fleet in the Eastern Cape Province, South Africa. This dissertation focuses on analysing fuel usage patterns and identifying anomalies in fuel transactions within this fleet during the period from April 2021 to January 2022.

The Eastern Cape Province, characterised by its rural landscape and economic challenges, provides a critical backdrop to this study. High unemployment rates, widespread crime, income inequality, poverty, and health crises, notably HIV/AIDS, mark the socio-economic fabric of the region (Oyelana & Fiseha, 2014). The province's infrastructure, particularly in terms of roads and public health, is underdeveloped, hampered by limited funding, inadequate capacity, and prevalent corruption (Mazele & Amoah, 2022). Such conditions exacerbate the difficulty in managing fleet operations efficiently, particularly in rural areas with poor road connectivity (Ngcobo, 2014).

Given the challenges faced by the Eastern Cape government fleet, it is important to understand fuel usage patterns and identify any potential anomalies in fuel transactions. This knowledge will contribute to improved fuel efficiency and help detect and prevent fraudulent activities, leading to better resource management and cost savings.

1.1 Research Aims and Objectives

The primary goal of this analysis is twofold:

- To conduct a thorough investigation into the fuel usage of the Eastern Cape government fleet, a subject not previously explored in depth.
- To develop simple, yet effective indicators for identifying anomalies in fuel transactions, facilitating the creation of predictive models that can achieve a high degree of accuracy with minimal information.

Specific objectives include:

- Employing exploratory data analysis and clustering techniques to reveal fuel usage patterns across different districts, vehicle makes, models and categories.
- Establishing simple yet robust indicators for flagging suspicious transactions and usage patterns, laying the groundwork for subsequent model development.
- Designing and training predictive models to efficiently predict anomalous transactions based on the established indicators, accounting for possible data limitations.

1.2 Dissertation Structure

The dissertation is structured as follows:

- **Chapter 2:** Discusses literature on various forms of fuel theft and transaction fraud analysis.
- **Chapter 3:** Describes the methodologies employed, including clustering techniques and predictive modeling.
- **Chapter 4:** Details the dataset and preparation process for analysis.
- **Chapter 5:** Presents findings from the exploratory data analysis, clustering, indicator development, and model performance.
- **Chapter 6:** Synthesises the findings and concludes the dissertation.

1.3 Code Repository

The code repository associated with this dissertation can be accessed via the following link:

<https://github.com/Jared-T/masters-dissertation>

Chapter 2

Literature Review

2.1 Fuel Theft: Prevalent Mechanisms and Their Dynamics

Fuel theft poses a serious threat for any vehicle fleet and warrants in-depth analysis. Unfortunately, academic research—particularly within the South African setting—remains limited. To gain a better understanding of this issue within the research context, alternative sources such as media reports and law enforcement records were used. These offer up-to-date insights into the diverse tactics of fuel theft throughout the region.

2.1.1 Categorising Fuel Theft Strategies

According to an article by Avis (2023) on fuel fraud in South Africa, thieves employ several strategies to commit fuel theft. While the article does not provide specific data for the Eastern Cape province, it can be reasonably assumed that these strategies are also prevalent in this region, given their widespread use across the country. These strategies include tank syphoning, card manipulation, card cloning, side fueling, multiple station consolidation, and continuous pumping.

- **Tank Syphoning:** This tactic involves the mechanical extraction of fuel from vehicle tanks, known for its stealth.
- **Card Manipulation:** Schemes involving fraudulent inflation of fuel card transactions are highly prevalent. Drivers may collude with service personnel to record higher fuel volumes than actually dispensed, allowing them to share the profits.
- **Card Cloning:** Unauthorised duplication of fuel cards enables criminals to purchase fuel for vehicles unrelated to the original card. While such fraud may be easier to detect, recovering financial losses remains a challenge.
- **Side Fueling:** This method involves simultaneously filling a vehicle and an external container under a single transaction. Identifying this often depends on detecting discrepancies between recorded and actual fuel volumes.

-
- **Multiple Station Consolidation:** Fraudsters may consolidate charges from various service stations into a single transaction, illicitly pocketing the accumulated cash. This scheme calls for close examination of transaction records.
 - **Continuous Pumping:** In this method, the fuel pump is left running after a transaction, allowing subsequent customers to be charged for fuel they didn't receive. Vigilant monitoring of transaction sequences can help reveal this type of fraud.

2.2 Technological Tools to Combat Fuel Fraud

In addition to understanding the various fuel theft mechanisms, physical tools can deter fuel theft before or as it occurs, with innovations designed to protect both stationary fuel supplies and vehicles. These tools target specific vulnerabilities, providing a multi-layered defense against theft.

- **Fuel Level Sensors:** These sensors provide real-time monitoring of fuel levels, detecting sudden drops that could indicate theft. Their alert systems play a vital role in early detection and potential prevention of unauthorised fuel withdrawals (SmartFuel, 2023).
- **Anti-Syphoning Devices:** Installed inside the fuel tank's filler neck, these devices make it impossible to insert siphoning equipment, providing a simple yet effective physical barrier against this common theft method (iFleet, 2023; Madke & Jaybhaye, 2016).
- **Lockable Fuel Caps:** These caps create a direct barrier to the fuel tank, requiring a specific key or code to open. They offer an affordable security measure with proven effectiveness (ReviewNitro, 2023).
- **GPS Tracking Systems:** Integrated GPS tracking provides real-time location data on vehicles. This feature is mainly used to prevent vehicle theft but can also be used get usage information and fuel tank levels if the correct equipment is installed (Tracking, 2023).
- **GSM-Based Fuel Theft Alerts:** These systems combine fuel level monitoring with Global System for Mobile Communications (GSM) technology to send immediate alerts when discrepancies suggest theft. The combination of monitoring and location tracking enables quicker response times, potentially minimising losses (Madke & Jaybhaye, 2016; Saini & Khan, 2021).

In South Africa, anti-syphoning devices are the the most common, due to their low cost and accessibility (Diesel Guard, 2024; TankSafe South Africa, 2024). However, no information was available about which devices are currently being employed by the Eastern Cape government fleet. Utilising these tools can create a more robust defense against the various methods of fuel theft, ultimately improving fuel security.

2.3 Model-Based Solutions for Fuel Theft Detection

In contrast to the physical devices described above, model-based solutions leverage historical data, including odometer readings, fuel tank levels, transaction data, distances travelled, and other relevant features, to ascertain the likelihood of fraudulent activities. Although there is a scarcity of literature explicitly focused on fuel transaction fraud within South Africa, the principles underlying these models align closely with those used in transaction fraud detection from other fields.

2.3.1 Characteristics of an Effective Fraud Detection System

According to Maes et al. (2002a), an effective fraud detection system should possess certain characteristics to identify and manage fuel fraud. These characteristics include:

1. **Unbalanced Data:** The system must be able to work with unbalanced data distributions, a common scenario in fraud detection where legitimate transactions usually vastly outnumber fraudulent ones.
2. **Robustness to Noise:** Given that datasets often contain corrupted or erroneous data, the system should maintain accuracy and effectiveness despite such noise.
3. **Handling Overlapping Data:** The ability to differentiate between fraudulent transactions that closely resemble legitimate ones is important too.
4. **Dynamic Adaptation:** As fraud techniques evolve over time (concept drift), the system needs to be dynamic and adaptable to these changing patterns.
5. **Effective Classification Metrics:** The selection of appropriate classification metrics is critical for the accurate evaluation of fraud detection methodologies.

2.3.2 Approaches to Addressing Challenges in Fraud Detection Modelling

Most transaction fraud modelling techniques are based on anomaly detection methods. This approach is based on the premise that fraudulent transactions are likely to be different from normal transactions in some way. Anomaly-based detection can be unsupervised, supervised, or semi-supervised (Abdallah et al., 2016).

- **Supervised Detection:** Supervised anomaly-based detection requires labeled data, where the algorithm is trained on a dataset containing both normal and fraudulent transactions. For instance, a supervised learning model for fuel transaction fraud detection could be trained on a dataset where each transaction is labeled as either legitimate or fraudulent, allowing the model to learn the characteristics of fraudulent behavior. However, it requires labeled data, which can be scarce or expensive to obtain.
- **Unsupervised Detection:** Unsupervised anomaly-based detection does not require labeled data, making it the opposite of supervised learning. This type of detection is often used when labeled data is scarce or expensive to obtain, but can be less accurate than supervised anomaly-based detection, as it does not have known examples of fraud to learn from. For example, in

fuel transaction fraud detection, an unsupervised learning algorithm could be used to identify unusual patterns in fuel consumption data without the need for labeled examples of fraudulent transactions. However, unsupervised anomaly-based detection can be less accurate than supervised anomaly-based detection.

- **Semi-Supervised Detection:** Semi-supervised anomaly-based detection uses a small amount of labeled data and a large amount of unlabeled data, combining aspects of both supervised and unsupervised learning. This type of detection can be more accurate than unsupervised anomaly-based detection and less expensive than supervised anomaly-based detection. In the context of fuel transaction fraud detection, a semi-supervised approach could involve training a model on a small set of labeled fraudulent and legitimate transactions, while also leveraging a larger dataset of unlabeled transactions to improve the model’s ability to identify anomalies. This approach can strike a balance between accuracy and the cost of obtaining labeled data.

2.3.3 A Review of Transaction Fraud Detection Models

Building on the approaches above are the specific implementations of various models to tackle the task of transaction fraud detection. Based on a survey by Adewumi and Akinyelu (2016), this section explores prevalent models within these categories, emphasising their strengths in combating fraud.

Hidden Markov Models (HMM) have been effectively used for credit card fraud detection. Khan, Pathan, and Ahmed (2014) combined HMMs with K-clustering to analyse transaction sequences, comparing incoming transactions against cardholder history and known patterns for dynamic risk assessment. The model’s accuracy in identifying unusual spending patterns was further demonstrated in Khan et al. (2013) and Bhusari and Patil (2011). Mhamane and Lobo (2012a); Mhamane and Lobo (2012b) also applied HMMs specifically to online banking fraud detection.

Additionally, Support Vector Machines (SVM) have shown strong performance in fraud detection. In a study comparing the performance of decision tree methods and SVMs on a real-world credit card fraud detection dataset, Sahin and Duman (2011) created three sample datasets with different ratios of fraudulent to normal transactions (1:1, 1:4, and 1:9) to evaluate the models under varying conditions. The first dataset contained an equal number of fraudulent and normal transactions, the second had four normal transactions for each fraudulent one, and the third had nine normal transactions for each fraudulent transaction. While the decision tree models, especially Classification and Regression Trees (C&RT), outperformed the SVM models in terms of accuracy and the number of fraudulent transactions correctly identified, the performance gap decreased as the training dataset size increased. Lu and Ju (2011) enhanced SVMs using techniques such as class weighting to improve accuracy and handle data imbalance.

Furthermore, meta-learning models introduce layered sophistication by combining the outputs of multiple base classifiers using a higher-level learning algorithm to improve overall performance. Pun (2011) applied this meta-learning strategy to enhance credit card fraud detection. The meta-classifier consisted of three base classifiers constructed using the k-nearest neighbour, decision tree (C4.5), and naïve Bayesian algorithms. These base algorithms were selected based on diversity calculations to ensure the classifiers had varied learning biases and could identify different patterns in the dataset. The outputs of the base classifiers were then used as input features, along with the original dataset attributes, to

train a meta-level naïve Bayesian classifier. This meta-classifier combined the base predictions to make the final classification. The naïve Bayesian algorithm was chosen as the meta-level algorithm because prior research had demonstrated its effectiveness in combining base classifiers for credit card fraud detection. By employing this meta-learning approach, Pun achieved a 24% to 34% improvement in fraud detection performance compared to the bank’s existing neural network-based fraud detection system, resulting in substantial cost savings. Earlier work by Stolfo et al. (1997) and Sen and Dash (2013) had also explored meta-learning algorithms for credit card fraud detection, demonstrating increased accuracy and efficiency through the combination of multiple learning algorithms.

Ensemble-based techniques have also shown promise in fraud detection. Zareapoor and Shamsolmoali (2005) used a bagging ensemble classifier for credit card fraud detection, showing its ability to handle data imbalance while achieving high detection rates. Ensemble models improve performance by combining multiple classifiers.

In addition, Artificial Immune System (AIS) based approaches model the biological immune system’s adaptability for fraud detection. Wong et al. (2012) developed an AIS architecture and prototype. Soltani et al. (2012) and Soltani Halvaie and Akbari (2014) proposed novel user-based AIS models that leverage evolutionary algorithms, which are optimisation techniques inspired by biological evolution processes such as reproduction, mutation, and selection. Genetic algorithms also use evolutionary strategies to optimise fraud detection. Patel and Singh (2013), Duman and Ozcelik (2011), and RamaKalyani and UmaDevi (2012) all applied GAs to credit card fraud, showing improved classification efficiency, especially when hybridised with other algorithms.

Frequent Itemset Mining has also been applied to fraud detection. Seeja and Zareapoor (2014) proposed the FraudMiner model based on frequent itemset mining. This technique excels at extracting patterns from transaction data to distinguish fraudulent activity, and is particularly effective on imbalanced datasets.

Lastly, Artificial Neural Networks (ANN) are at the forefront of fraud detection research. Studies have combined ANNs with techniques such as simulated annealing (Khan, Akhtar, & Qureshi, 2014), Bayesian networks (Maes et al., 2002b), and web mining (Modi et al., 2013) to optimise accuracy and speed. Novel ANN architectures (Ganesh & Sena, 2012; Van Vlasselaer et al., 2015) continue to advance the field.

In summary, the literature shows a wide range of sophisticated models being applied to transaction fraud detection, each with unique strengths. HMMs, SVMs, meta-learning, frequent itemset mining, ensemble methods, AIS, GAs, and ANNs have all demonstrated capabilities for identifying fraudulent patterns in complex real-world financial data. These models excel at handling imbalanced datasets, adapting to evolving fraud strategies, and optimising classification accuracy and efficiency. Given the similarities between financial transaction fraud and fuel transaction fraud, these techniques should be applicable to fuel transaction fraud detection as well, providing a solid foundation for developing effective fraud detection systems in the fuel industry.

2.4 The Context of the Eastern Cape

The Eastern Cape Province faces considerable socioeconomic challenges stemming from its historical legacy and ongoing disparities (Ross, 2013). The region experiences high unemployment, crime, poverty, and income inequality, further exacerbated by a persistent HIV/AIDS burden (Oyelana & Fiseha, 2014). These factors hinder development and social progress considerably.

According to the latest Census 2022 results, the total population of Eastern Cape increased by 10.2% from 6.5 million in Census 2011 to 7.2 million (StatsSA, 2023). Over 85% of the population in the province is black African, 7.6% is coloured, 5.6% is white, and Indian/Asian comprises less than half a percentage of the population. The province also experiences high out-migration, with a substantial number of persons migrating to Western Cape and Gauteng (StatsSA, 2023).

Infrastructure inadequacies pose a major obstacle. Despite government efforts to improve the road network (Ngcobo, 2014), progress is hampered by corruption, funding shortages, and insufficient technical capacity (Mazele & Amoah, 2022). This disproportionately impacts rural communities, limiting access to essential services, economic opportunities, and agricultural markets (Ngcobo, 2014). Additionally, the province's complex political landscape, including factional divisions, also influences the efficacy of policies and programs (Ngqula, 2022).

Regional inequity persists, with rural areas historically marginalised in infrastructure investment (Ngcobo, 2014). This results in limited access to basic necessities such as water and sanitation. The healthcare system faces challenges including language barriers and vulnerability to infectious diseases such as SARS-CoV-2 (Mhlauli, 2017), Moyo and Osunkunle (2021).

The COVID-19 pandemic has further strained the province's growth trajectory, with the unemployment rate standing at 47% for the 3rd quarter of 2021 compared to 45.8% for the same period last year (Hendricks, 2021). Joblessness in Buffalo City and Nelson Mandela Bay stands at 40%. The stagnant economic outlook continues to have dire effects on residents (Hendricks, 2021).

To advance development in the Eastern Cape, focused policy initiatives and sufficient resource allocation are critical. Prioritising support for small businesses, improving healthcare delivery, and investing in sustainable infrastructure—particularly in underserved rural areas—are essential for overcoming these socioeconomic hurdles (Ngiam, 2005; Oyediran-Tidings et al., 2021).

Chapter 3

Methods

This chapter presents the methods that will be employed for investigating fuel usage in the Eastern Cape government fleet. Clustering techniques (k-means++, Agglomerative, Gaussian Mixture Model and BIRCH) will be used to reveal patterns and predictive models (Naive Bayes, Linear Support Vector Machine, XGBoost, Random Forest, Multi-layer Perceptron and Stochastic Gradient Descent Classifier) will be used to predict the level of fuel transaction fraud.

3.1 k-means++ Clustering

This section is based on a paper by Arthur and Vassilvitskii (2007).

3.1.1 Introduction

The classic k-means algorithm aims to partition a dataset into k clusters while minimising the total squared distance between each data point and its assigned cluster center. While fast and simple, traditional k-means suffers from sensitivity to the initial choice of centers, potentially leading to suboptimal clustering results.

The k-means++ algorithm addresses this by strategically choosing initial cluster centers. The key idea is to select centers with a probability proportional to their squared distance from the nearest existing center. This improves both the speed and accuracy of k-means (Arthur & Vassilvitskii, 2007). The k-means++ algorithm provides a better initialisation strategy that leads to faster convergence and improved clustering results compared to the standard k-means algorithm.

3.1.2 Underlying Assumptions

- **Cluster Structure:** The dataset is assumed to contain distinct clusters of data points, meaning these clusters should have relatively clear boundaries.
- **Spherical Clusters:** k-means++ is most effective when clusters are roughly spherical or hyperspherical in shape. This is because it primarily uses Euclidean distance to measure similarity

between points. Due to the use of Euclidean distance, outliers may cause issues with the clustering, unless outliers have their own cluster, which would require tuning the number of clusters beforehand. When the clusters have non-spherical shapes or varying densities, k-means++ may not perform optimally. In such cases, other clustering algorithms such as DBSCAN or hierarchical clustering might be more suitable (Ester et al., 1996; Jain et al., 1999).

- **Number of Clusters (k):** The ideal number of clusters is assumed to be known or estimable in advance. k-means++ does not inherently determine the optimal value of k .

3.1.3 Algorithm and Mechanism

For the k-means problem, we are given k clusters and a set of n data points $\mathcal{X} \subset \mathbb{R}^d$. We wish to choose k centers $\mathcal{C}$ so as to minimise the potential function,

$$\phi = \sum_{x \in \mathcal{X}} \min_{c \in \mathcal{C}} |x - c|^2. \quad (3.1)$$

Choosing these centers implicitly defines a clustering - for each cluster, we set one cluster to be the set of data points that are closer to that center than to any other. Finding an exact solution to the k-means problem is NP-hard.

Let $\mathcal{C}_{OPT}$ denote the optimal clustering for a given instance of the k-means problem and ϕ_{OPT} the corresponding potential. Given a clustering $\mathcal{C}$ with potential ϕ , we also let $\phi(\mathcal{A})$ denote the contribution of $\mathcal{A} \subset \mathcal{X}$ to the potential as follows:

$$\phi(\mathcal{A}) = \sum_{x \in \mathcal{A}} \min_{c \in \mathcal{C}} |x - c|^2 \quad (3.2)$$

k-means algorithm

1. Arbitrarily choose k initial centers $\mathcal{C} = \{c_1, \dots, c_k\}$.
2. For each $i \in 1, \dots, k$, set the cluster C_i to be the set of points in $\mathcal{X}$ that are closer to c_i than they are to c_j for all $j \neq i$.
3. For each $i \in 1, \dots, k$, set c_i to be the center of mass of all points in C_i : $c_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$.
4. Repeat Steps 2 and 3 until $\mathcal{C}$ no longer changes.

It is standard practice to choose initial centers uniformly at random from $\mathcal{X}$. For step 2, ties may be arbitrarily broken as long as the method is consistent. Steps 2 and 3 are guaranteed to decrease ϕ until it is no longer possible to do so.

k-means++ algorithm

Instead of starting with an arbitrary set of cluster centers, k-means++ chooses them more carefully. Let $D(x)$ denote the shortest distance from a data point x to the closest center we have already chosen. Then, we define the following algorithm:

1. Choose an initial center c_1 uniformly at random from $\mathcal{X}$.
2. Choose the next center c_i , selecting $c_i = x' \in \mathcal{X}$ with probability $\frac{D(x')^2}{\sum_{x \in \mathcal{X}} D(x)^2}$.
3. Repeat Step 2 until we have chosen a total of k centers.
4. Proceed as with the standard k -means algorithm.

The weighting used in Step 2 is simply called " D^2 weighting". This weighting gives higher probability to points that are farther away from the already chosen centers, ensuring a more balanced initial distribution of centers (Arthur & Vassilvitskii, 2007).

The initialisation step of k-means++ has a time complexity of $\mathcal{O}(nk)$, where n is the number of data points and k is the number of clusters. The subsequent k-means iterations have the same complexity as the standard k-means algorithm (Arthur & Vassilvitskii, 2007).

3.1.4 Possible Implications for Future Research

The k-means++ algorithm is probabilistic, which may have implications for future research that relies on the cluster definitions due to stacking/inflating probabilities. Stacking probabilities refers to the potential issue of compounding uncertainties when using the results of one probabilistic model as input for another. In the current context, this could occur if the cluster assignments are used as definitive labels in subsequent analyses without accounting for the inherent uncertainty in the clustering process (ie predictive models).

However, keep in mind that in this analysis, clustering is employed as an exploratory tool to better understand patterns in fuel usage and transactions, rather than as a definitive fraud indicator. The insights gained from the clustering process are used to guide the development of simple, robust indicators and predictive models. As such, the probabilistic nature of k-means++ does not pose a problem for the current analysis.

Nonetheless, future research that builds upon the cluster definitions from this study should be mindful of the potential for stacking or inflating probabilities. When using cluster assignments as input for predictive models or other downstream tasks, it is necessary to consider the uncertainty in the clustering process. Techniques such as consensus clustering or ensemble clustering can help mitigate the effect of stacking probabilities (Ghosh et al., 2002; Strehl & Ghosh, 2002).

3.1.5 Python Implementation of k-means++

For the Python implementation, the **KMeans** function in the **sklearn** package was used (Pedregosa et al., 2011). More details on the exact implementation can be found on the dissertation GitHub repository.

3.2 Agglomerative Hierarchical Clustering

This section is based on a paper by Ward Jr (1963).

3.2.1 Introduction

Agglomerative hierarchical clustering (AHC) is a bottom-up method within the family of hierarchical clustering algorithms. It begins with each data point considered its own cluster and iteratively merges the most similar clusters based on a predefined distance metric and linkage criterion. This process continues until all data points are merged into a single cluster, forming a complete hierarchical structure (dendrogram). If a desired number of clusters is specified, the dendrogram is cut at an appropriate level to obtain the requested number of clusters.

Agglomerative hierarchical clustering has a long history and has been widely used in various domains (Sneath & Sokal, 1973). One of the advantages of this method is the interpretability of the resulting dendrogram, which provides a visual representation of the clustering hierarchy (Rokach & Maimon, 2005).

3.2.2 Underlying Assumptions

- **Data Structure:** Agglomerative clustering generally assumes that meaningful clusters exist within the data. Different linkage methods implicitly favor different cluster shapes (e.g., Ward’s method assumes more spherical clusters).
- **Similarity Metric:** The chosen distance metric strongly influences how similarity between data points and clusters is defined. Listed below are some common metrics.
 - Euclidean distance: Suitable for data with natural geometric interpretation (e.g., clustering geographic locations).
 - Manhattan distance: Appropriate for high-dimensional data with equally important dimensions (e.g., clustering text documents).
 - Jaccard distance: Computationally expensive and compares the size of set intersection to the size of union (e.g., clustering sets of items, such as customer purchases).
 - Mahalanobis distance: Computationally expensive and accounts for covariance structure and scales distance based on variability (e.g., clustering multivariate data with correlated variables).
- **Non-Reversibility:** Merges performed in earlier steps of agglomerative clustering cannot be undone. This requires careful consideration of linkage methods and stopping criteria. Once a merge is performed, it cannot be undone, and this can lead to suboptimal clustering results if the initial merges are not appropriate (Kaufman & Rousseeuw, 2009). This emphasises the importance of carefully selecting the linkage method and distance metric.

3.2.3 Algorithm and Mechanism

Denote the dataset as $\mathcal{X} = x_1, x_2, \dots, x_n$ where each data point x_i is a vector in $\mathbb{R}^d$. Agglomerative clustering with Ward’s method aims to minimise the increase in within-cluster variance as clusters are formed. Within-cluster variance measures the compactness and homogeneity of clusters, and minimising it aims to create compact and homogeneous clusters (Ward Jr, 1963). The core mechanism is shown below:

1. **Initialisation:** Begin with each data point as its own cluster. This means there are initially n clusters.
2. **Cluster SSE:** For each existing cluster $\mathcal{C}_i$, calculate the Sum of Squared Errors (SSE):

$$SSE(\mathcal{C}_i) = \sum_{x \in \mathcal{C}_i} |x - \mu_{\mathcal{C}_i}|^2 \quad (3.3)$$

where $\mu_{\mathcal{C}_i}$ is the centroid (mean) of cluster $\mathcal{C}_i$.

3. **Pairwise Merge Evaluation:** Consider all possible pairs of clusters to merge. For each potential pair $(\mathcal{C}_i, \mathcal{C}_j)$: Calculate the SSE of the hypothetical merged cluster, $\mathcal{C}_{merged}$. The increase in variance due to the merge is:

$$\Delta(\mathcal{C}_i, \mathcal{C}_j) = SSE(\mathcal{C}_{merged}) - [SSE(\mathcal{C}_i) + SSE(\mathcal{C}_j)] \quad (3.4)$$

4. **Minimal Variance Merge:** Identify the cluster pair $(\mathcal{C}_i, \mathcal{C}_j)$ with the smallest $\Delta(\mathcal{C}_i, \mathcal{C}_j)$. Execute the merge.
5. **Recompute Distances:** Update the pairwise distances between the newly formed cluster and the remaining clusters.

The time complexity of agglomerative hierarchical clustering is typically $\mathcal{O}(n^3)$ for the standard algorithm, where n is the number of data points (Manning et al., 2008). However, there are more efficient implementations that can reduce the time complexity to $\mathcal{O}(n^2)$ by using priority queues or other data structures (Müllner, 2011).

Agglomerative hierarchical clustering has several advantages, including the ability to handle any distance metric, the interpretability of the dendrogram, and the flexibility to obtain different numbers of clusters by cutting the dendrogram at different levels (Gan et al., 2007). However, it also has some limitations, such as sensitivity to noise and outliers, the inability to undo merges, and high computational complexity (Xu & Wunsch, 2005).

3.2.4 Python Implementation of Agglomerative Clustering

For the Python implementation, the **AgglomerativeClustering** function in the **sklearn** package was used (Pedregosa et al., 2011). More details on the exact implementation can be found on the dissertation GitHub repository.

3.3 Gaussian Mixture Models (GMM) Clustering

This section is based on a paper by Xu and Jordan (1996).

3.3.1 Introduction

Gaussian Mixture Models provide a probabilistic clustering method that assumes the data is generated from a mixture of underlying Gaussian distributions. Each Gaussian represents a potential cluster, and GMMs estimate distribution parameters to describe the data's cluster structure. GMMs provide a probabilistic framework that allows for soft clustering (i.e., each data point has a probability of belonging to each cluster) and can handle clusters with different sizes and shapes (Bishop, 2006).

3.3.2 Underlying Assumptions

- **Mixture Distribution:** GMMs assume that the data is drawn from a finite mixture of Gaussian distributions. While GMMs are flexible and can approximate many different distributions, they may not be suitable for datasets with highly non-Gaussian clusters or complex cluster shapes (Fraley & Raftery, 2002). For example, in a fuel fraud transaction detection dataset, fraudulent transactions may exhibit intricate, non-Gaussian patterns, such as occurring in specific geographic regions, at unusual times, or with abnormally large or small fuel quantities. These fraud patterns may not be well-approximated by Gaussian distributions. Consequently, the Gaussian components of the GMM may not be flexible enough to accurately represent the irregular shapes and distributions of these clusters, potentially leading to suboptimal fraud detection performance.
- **Cluster Representation:** Each Gaussian component in the mixture is believed to represent a cluster within the data. This makes them probabilistic and prone to the same issues as the k-means++ cluster definitions if used directly for the predictive models. As stated before, this was not the case and the clusters were used merely for exploratory purposes.
- **Parameters:** The clusters are often parameterised by their means, covariances, and the mixing probability of each Gaussian component.

3.3.3 Algorithm and Mechanism

Notation:

- Dataset: $\mathcal{X} = x_1, x_2, \dots, x_n$, with each data point $x_i \in \mathbb{R}^d$
- Number of Gaussian components: K (selecting the appropriate value of K is crucial for obtaining meaningful clustering results; common approaches include the Bayesian Information Criterion (BIC) and the Akaike Information Criterion (AIC) (Fraley & Raftery, 1998))
- Mean of component k : μ_k
- Covariance matrix of component k : Σ_k
- Mixing weight of component k : π_k (where $\sum_{k=1}^K \pi_k = 1$)

Probability Density of a GMM

The probability density function of a GMM over a data point x is:

$$p(x) = \sum_{k=1}^K \pi_k \mathcal{N}(x|\mu_k, \Sigma_k) \quad (3.5)$$

where $\mathcal{N}(x|\mu_k, \Sigma_k)$ is the multivariate Gaussian probability density function.

Robust EM Algorithm

Let z_{ik} denote the latent variable indicating whether data point x_i belongs to component k (with $z_{ik} = 1$ if it does, 0 otherwise). The Expectation-Maximisation (EM) algorithm is the standard algorithm for learning the parameters of a GMM. It alternates between the E-step (expectation step), where the posterior probabilities of the latent variables are computed, and the M-step (maximisation step), where the model parameters are updated based on the current responsibilities (Dempster et al., 1977).

1. **Initialisation:** Choose initial values for μ_k , Σ_k , and π_k for $k = 1, 2, \dots, K$.

2. **E-Step:**

- Calculate responsibilities (posterior probabilities):

$$\gamma_{ik} = \frac{\pi_k \mathcal{N}(x_i|\mu_k, \Sigma_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(x_i|\mu_j, \Sigma_j)} \quad (3.6)$$

- Outlier Weighting (Robustness): Calculate Mahalanobis distance for each point to each component:

$$d_{ik}^2 = (x_i - \mu_k)^T \Sigma_k^{-1} (x_i - \mu_k) \quad (3.7)$$

- Define an outlier threshold (e.g., based on a percentile of the chi-squared distribution).
- Down-weight responsibilities for points where d_{ik}^2 exceeds the threshold (there are different weighting functions proposed in the literature).

3. **M-Step:** Update the parameters, incorporating down-weighting from E-step:

$$\begin{aligned} \mu_k^{new} &= \frac{\sum_{i=1}^n w_i \gamma_{ik} x_i}{\sum_{i=1}^n w_i \gamma_{ik}} \\ \Sigma_k^{new} &= \frac{\sum_{i=1}^n w_i \gamma_{ik} (x_i - \mu_k^{new})(x_i - \mu_k^{new})^T}{\sum_{i=1}^n w_i \gamma_{ik}} \\ \pi_k^{new} &= \frac{\sum_{i=1}^n w_i \gamma_{ik}}{n} \end{aligned} \quad (3.8)$$

where w_i is the weight associated with data point x_i (potentially down-weighted in the E-step for outliers).

4. **Iteration:** Repeat E and M steps until convergence.

Key Idea of Robust Modification: The core change lies in the outlier weighting within the E-step, which reduces the influence of points that are distant from any Gaussian component, increasing the algorithm’s robustness.

One shouldn’t overlook the fact that GMMs have some limitations and challenges. GMMs can be sensitive to the initialisation of the parameters, and the EM algorithm may converge to a local optimum (Biernacki et al., 2003). Additionally, GMMs assume that each cluster follows a Gaussian distribution, which may not always be appropriate for real-world datasets (McLachlan & Peel, 2004).

3.3.4 Python Implementation GMM Clustering

For the Python implementation, the **GaussianMixture** function in the **sklearn** package was used (Pedregosa et al., 2011). More details on the exact implementation can be found on the dissertation GitHub repository.

3.4 BIRCH Clustering

This section is based on a paper by Zhang et al. (1996).

3.4.1 Introduction

Traditional clustering algorithms often struggle when confronted with very large datasets due to scalability limitations and memory constraints. BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) addresses these challenges through a hierarchical, incremental clustering approach. The core of BIRCH lies in its use of Clustering Features (CFs) and the CF-Tree structure, which provide a compact summary of the dataset. BIRCH has several advantages compared to other clustering algorithms, particularly in terms of its ability to handle large datasets efficiently. It is an incremental algorithm that uses a compact summary of the data through CFs, allowing it to process data in a single pass (Zhang et al., 1996).

3.4.2 Underlying Assumptions

- **Data Types:** BIRCH is designed primarily for numerical data that can be represented in Euclidean space (metric attributes). It has limitations in directly handling categorical data.
- **Cluster Shape:** BIRCH exhibits optimal performance when clusters tend to have spherical or convex shapes. It might be less effective with datasets containing arbitrarily shaped clusters.
- **Noise:** BIRCH possesses a degree of robustness to noise within the data, as outliers can be identified to some extent. The threshold parameter T in the CF-Tree construction process determines the maximum radius allowed for a sub-cluster represented by a leaf node. By setting an appropriate value for T , BIRCH can isolate outliers and prevent them from greatly influencing the clustering results. However, extreme outliers or a high proportion of noisy data points may still affect the clustering outcome.

3.4.3 Algorithm and Mechanism

Clustering Feature (CF)

A Clustering Feature is defined as a concise representation of a cluster. It is represented by the tuple (N, LS, SS) , where:

- N : The number of data points within the cluster.
- LS : The linear sum of the data points (a vector sum).
- SS : The square sum of the data points.

Importantly, Clustering Features have the property of additivity. This property is particularly beneficial in the context of fuel fraud transaction detection. It allows BIRCH to efficiently summarise and compress large-scale datasets by combining CFs of subclusters. Although BIRCH was only used for exploratory purposes in this analysis, this property also facilitates incremental updates to the clustering structure as new data arrives, making BIRCH suitable for real-time fraud detection.

3.4.4 CF-Tree

The CF-Tree is a height-balanced tree data structure that forms the backbone of the BIRCH algorithm. It allows for efficient insertion and retrieval of data points, as well as the ability to dynamically adjust the tree structure based on the data distribution (Zhang et al., 1996). Key parameters include:

- **Branching Factor (B):** The maximum number of children a non-leaf node can have.
- **Threshold (T):** A maximum radius within which subclusters of a node must fit.

Each leaf node contains CFs representing micro-clusters. The tree is built dynamically as data is inserted.

Insertion Process

1. Start at the root of the CF-Tree.
2. Identify the closest CF node at the current level, based on an appropriate distance metric.
3. Descend to the closest child node.
4. If the closest CF in the child node can absorb the new data point without exceeding the threshold T , update the CF. Otherwise, create a new CF entry.
5. If adding a new CF entry would overflow the node (exceeding the branching factor B), split the node. Information about the split is propagated upwards to adjust the tree structure.

BIRCH Algorithm Overview

1. **Phase 1 (Tree Building):** Scan the dataset and construct an initial CF-Tree. This phase builds the initial CF-Tree by incrementally inserting data points and adjusting the tree structure based on the threshold and branching factor.
2. **Phase 2 (Optional):** Further condense the CF-Tree for size reduction. This optional phase aims to further reduce the size of the CF-Tree by merging or removing CFs based on certain criteria.
3. **Phase 3 (Global Clustering):** A traditional clustering algorithm (such as k-means) is applied to the CFs, treating them as pseudo-data points. This phase performs global clustering on the CFs to obtain the final clusters.

The time complexity of building the CF-Tree is $\mathcal{O}(n)$, where n is the number of data points, and the space complexity is $\mathcal{O}(B \cdot L)$, where B is the branching factor and L is the maximum depth of the tree (Zhang et al., 1996).

Mathematical Notation

- Let N be the number of data points in a cluster: $N = \sum_{i=1}^N 1$
- Let $\vec{LS}$ be the linear sum of the N data points: $\vec{LS} = \sum_{i=1}^N \vec{X}_i$, where $\vec{X}_i$ is the i -th data point.
- Let SS be the square sum of the data points: $SS = \sum_{i=1}^N \vec{X}_i^2$
- The centroid $\vec{X}_0$ of the cluster is defined as: $\vec{X}_0 = \frac{\vec{LS}}{N}$
- The radius R of the cluster, which is the average distance from the data points to the centroid, is defined as:

$$R = \sqrt{\frac{SS}{N} - \left(\frac{\vec{LS}}{N}\right)^2} = \sqrt{\frac{N \cdot SS - \vec{LS}^2}{N \cdot (N - 1)}} \quad (3.9)$$

- The diameter D of the cluster, which is the average pairwise distance between data points, is defined as:

$$D = \sqrt{\frac{2 \cdot N \cdot SS - 2 \cdot \vec{LS}^2}{N \cdot (N - 1)}} \quad (3.10)$$

- If $CF1 = (N_1, \vec{LS}_1, SS_1)$ and $CF2 = (N_2, \vec{LS}_2, SS_2)$ are the Clustering Features for two disjoint clusters, the Clustering Feature $CF1 + CF2 = (N_1 + N_2, \vec{LS}_1 + \vec{LS}_2, SS_1 + SS_2)$ represents the union of the two clusters.
- The distance d between two clusters $CF1$ and $CF2$ is defined as the Euclidean distance between their centroids:

$$d(CF1, CF2) = \sqrt{\frac{N_1 \cdot N_2}{N_1 + N_2}} \cdot \left| \frac{\vec{LS}_1}{N_1} - \frac{\vec{LS}_2}{N_2} \right| \quad (3.11)$$

It is worth mentioning that BIRCH may not perform well when the clusters have highly irregular shapes or when the data contains a large number of outliers (Zaki & Meira Jr, 2014). Additionally, the quality of the clustering results depends on the choice of the threshold parameter T and the branching factor B (Zhang et al., 1997).

3.4.5 Python Implementation of BIRCH Clustering

For the Python implementation, the **Birch** function in the **sklearn** package was used (Pedregosa et al., 2011). More details on the exact implementation can be found on the dissertation GitHub repository.

3.5 Modelling Methods Overview

This section provides an overview of the modelling methods employed. Table 3.1 below presents a comparison of the key characteristics, advantages, and disadvantages of all the modelling methods considered in the analysis. To maintain the conciseness of the main methodology chapter, detailed explanations of the Random Forest, MLP, and XGBoost algorithms will be provided in the following sections. For more information on the Naive Bayes, Linear Support Vector Machine (SVM), and Stochastic Gradient Descent (SGD) Classifier, including their mathematical formulations and implementation details, please refer to Appendix A.

Table 3.1: Comparison of Modelling Methods

Method	Key Characteristics	Advantages	Disadvantages
Naive Bayes	• Probabilistic model • Assumes conditional independence between features (Domingos & Elkan, 1997)	• Simple and fast • Works well with high dimensional data • Requires little training data	• Assumes feature independence (often violated) • Sensitive to feature scaling
Linear SVM	• Finds optimal hyperplane to separate classes • Maximises margin between hyperplane and closest data points (Burges, 1998)	• Effective in high dimensional spaces • Memory efficient • Robust to outliers	• Assumes linear separability • Sensitive to feature scaling • Ineffective for nonlinear problems
XGBoost	• Gradient boosted decision trees • Uses regularisation to control complexity • Handles missing data (Chen & Guestrin, 2016)	• High performance and scalability • Effective for structured data • Has built-in regularisation	• Sensitive to hyperparameter tuning • Prone to overfitting • Less interpretable than simpler models
Random Forest	• Ensemble of decision trees • Uses bagging and feature randomness • Averages predictions of individual trees (Breiman, 2001)	• Reduces overfitting • Handles high dimensional data • Provides feature importance	• Computationally expensive • Less interpretable than single trees • Prone to overfitting with noisy data
MLP	• Feedforward neural network • Learns complex nonlinear relationships • Uses backpropagation for training (Hinton et al., 2006)	• Can learn complex patterns • Flexible architecture design • Effective for unstructured data	• Computationally expensive to train • Requires large amount of training data • Prone to overfitting, sensitive to initialization
SGD Classifier	• Linear classifier • Uses stochastic gradient descent optimization • Efficient for large-scale learning (Bottou, 2010)	• Computationally efficient • Scalable to large datasets • Effective for online learning	• Sensitive to feature scaling • Requires careful learning rate tuning • May converge slowly with noisy data

3.6 XGBoost Model

This section is based on a paper by Chen and Guestrin (2016).

3.6.1 Introduction

XGBoost (Extreme Gradient Boosting) is a highly effective and widely adopted implementation of gradient boosted decision trees (Friedman, 2001; Friedman, 2002). It has gained popularity due to its exceptional performance in machine learning competitions and real-world applications (Chen & Guestrin, 2016). XGBoost excels in structured data tasks, such as classification and regression, and is renowned for its speed, performance, and scalability (Chen & Guestrin, 2016; Mitchell, 2017). The algorithm builds upon the principles of gradient boosting Friedman (2001) and introduces several optimisations and enhancements, making it a powerful tool for tackling problems with tabular data (Chen & Guestrin, 2016; Nielsen, 2016).

Explanation of "Extreme" in XGBoost:

XGBoost is considered "extreme" compared to traditional gradient boosting methods due to its optimised implementation and additional features that contribute to its speed and performance Chen and Guestrin (2016). XGBoost employs parallelisation techniques, such as parallel tree construction and distributed computing, which allow it to efficiently handle large-scale datasets (Chen & Guestrin, 2016; Mitchell, 2017). Moreover, it incorporates algorithmic optimisations, including a novel tree pruning algorithm and a sparsity-aware split finding method, which further enhance its computational efficiency (Chen & Guestrin, 2016).

3.6.2 Underlying Assumptions

1. **Differentiable Loss Function:** XGBoost presumes the use of a differentiable loss function. Common choices include mean squared error (regression) and log loss (classification).
2. **Additive Training:** The assumption is that the model is built iteratively, with new trees added to improve upon the errors of previous ones.

3.6.3 Algorithm and Mechanism

Objective Function

At its core, XGBoost seeks to minimise a carefully designed objective function. For the k -th tree, the function is expressed as:

$$Obj^{(k)} = \sum_{i=1}^n l(y_i, \hat{y}_i^{(k-1)} + f_k(x_i)) + \Omega(f_k) \quad (3.12)$$

where each of the components are:

-
- i : Index of a data point in the dataset.
 - n : Total number of data points.
 - y_i : True label for data point i .
 - $\hat{y}_i^{(k-1)}$: Prediction from the previous ensemble of trees (up to tree $k - 1$).
 - $f_k(x_i)$: The output from tree k for data point i .
 - l : A differentiable loss function (e.g., squared error for regression, log loss for classification).
 - $\Omega(f_k)$: Regularisation term controlling the complexity of tree k .

Gradient Boosting and Second-Order Approximation

XGBoost builds its model iteratively. Denote the gradient and Hessian of the loss function at iteration k as:

- $g_i = \partial_{\hat{y}_i^{(k-1)}} l(y_i, \hat{y}_i^{(k-1)})$
- $h_i = \partial_{\hat{y}_i^{(k-1)}}^2 l(y_i, \hat{y}_i^{(k-1)})$

Using a second-order Taylor expansion, XGBoost approximates the objective function. After simplification, the objective at iteration k becomes dependent on g_i and h_i .

Tree Structure

To determine the optimal structure of a new tree, XGBoost calculates a score to assess potential splits. This score considers:

- **Gain:** Improvement in the objective function due to a split.
- **Regularisation terms:** Calculated based on aspects such as leaf weights and number of leaves.

XGBoost greedily builds trees, choosing splits that maximise this score.

Mathematical Formulation of Split Score:

The score for a split is determined by the following formula (Chen & Guestrin, 2016):

$$\text{Score} = \frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda} - \frac{(G_L + G_R)^2}{H_L + H_R + \lambda} - \gamma \quad (3.13)$$

where G_L and G_R are the sums of the gradients for the left and right child nodes, respectively, and H_L and H_R are the sums of the Hessians for the left and right child nodes, respectively. The λ and γ terms are regularisation parameters that control the complexity of the tree.

Regularisation

The γ parameter in XGBoost controls the minimum loss reduction required for a further split (Chen & Guestrin, 2016). It acts as a threshold that a potential split must exceed in order to be considered for tree construction. A higher value of γ leads to more conservative splits and can help prevent overfitting.

The λ parameter applies L2 regularisation to the leaf weights, which helps to control the complexity of the tree (Chen & Guestrin, 2016). It adds a penalty term to the objective function based on the square of the leaf weights, discouraging large leaf weights and promoting simpler tree structures.

In addition to L2 regularisation, XGBoost also supports L1 regularisation, which adds a penalty term based on the absolute values of the leaf weights (Chen & Guestrin, 2016). XGBoost also allows for setting a maximum depth for the trees, which can further control the model complexity and prevent overfitting.

Handling Missing Values

XGBoost has a built-in mechanism to handle missing values during training (Chen & Guestrin, 2016). When a missing value is encountered in a feature, XGBoost learns the best default split direction for that feature. This allows the algorithm to make split decisions even in the presence of missing data, without the need for explicit imputation (Chen & Guestrin, 2016). The default split direction is learned based on the gradients and Hessians of the available instances, ensuring that the missing values are handled in a way that optimises the objective function (Chen & Guestrin, 2016).

3.6.4 Python Implementation of XGBoost Model

For the Python implementation, the **XGBClassifier** function in the **xgboost** package was used (Pedregosa et al., 2011). More details on the exact implementation can be found on the dissertation GitHub repository.

3.7 Random Forest

This section is based on a paper by Breiman (2001).

3.7.1 Introduction

Random Forest is an ensemble learning method that constructs a multitude of decision trees during training and outputs the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. It combines the concepts of bagging (bootstrap aggregating) and random feature selection to create diverse and uncorrelated trees (Breiman, 2001). Random Forest is considered an improvement over a single decision tree as the ensemble approach helps to reduce overfitting and improve generalisation performance (Breiman, 2001; Hastie et al., 2009).

3.7.2 Underlying Assumptions

1. **Feature Independence:** Random Forest assumes that the features used for building the individual decision trees are independent of each other. This assumption allows for the selection of random subsets of features at each node split (Breiman, 2001).
2. **Diversity:** The key assumption is that by introducing randomness in the tree-building process, the resulting ensemble of trees will be diverse and have uncorrelated predictions, leading to improved generalisation performance (Breiman, 2001).

3.7.3 Algorithm and Mechanism

1. **Bootstrap Sampling:** From the original training dataset of size N , create B bootstrap samples by randomly sampling N instances with replacement. Each bootstrap sample will be used to train a separate decision tree (Breiman, 2001).
2. **Feature Subsampling:** At each node of a decision tree, instead of considering all available features for splitting, randomly select a subset of m features (typically $m = \sqrt{p}$, where p is the total number of features) (Breiman, 2001). This step introduces additional randomness and reduces the correlation between trees. It is key to note that while $\sqrt{p}$ is a common default value, the optimal value of m may depend on the specific dataset and problem at hand (Hastie et al., 2009). Other values, such as $\log_2(p)$, are sometimes used as well (Breiman, 2001).
3. **Tree Building:** Construct a decision tree using the selected bootstrap sample and feature subset. Each tree is grown to the maximum depth without pruning to maintain low bias (Breiman, 2001).
4. **Ensemble Prediction:** To make predictions for a new instance, pass it through all the decision trees in the forest. In classification, take the majority vote of the predictions from individual trees. In regression, take the average of the predictions (Breiman, 2001).

The number of trees (B) in the Random Forest ensemble plays a crucial role in the performance of the algorithm. Increasing the number of trees can improve the stability and accuracy of the predictions, but also increases the computational cost (Breiman, 2001; Hastie et al., 2009).

Mathematical Formulation

Let $T_1, T_2, \dots, T_B$ be the ensemble of B decision trees in the Random Forest. For a new instance $\mathbf{x}$, the prediction of the Random Forest is given by:

- **Classification:** $\hat{y} = \text{majority vote}(\{T_1(\mathbf{x}), T_2(\mathbf{x}), \dots, T_B(\mathbf{x})\})$, where the class with the highest number of votes from the individual trees is selected as the final prediction (Breiman, 2001).
- **Regression:** $\hat{y} = \frac{1}{B} \sum_{i=1}^B T_i(\mathbf{x})$

Random Forest also provides an internal estimate of the generalisation error through the use of out-of-bag (OOB) samples (Breiman, 2001). OOB samples are the instances that are not included in a particular bootstrap sample and can be used as a validation set for each tree. The OOB error estimate provides an unbiased estimate of the Random Forest's performance without the need for a separate validation set (Breiman, 2001). Random Forest has several advantages, including its robustness to outliers and noise, its ability to handle high-dimensional data, and its built-in feature importance measures (Breiman, 2001; Hastie et al., 2009).

3.7.4 Python Implementation of Random Forest Model

For the Python implementation, the **RandomForestClassifier** or **RandomForestRegressor** function in the **sklearn.ensemble** module can be used. More details on the exact implementation can be found on the dissertation GitHub repository.

3.8 Multi-layer Perceptron Model

This section is based on a paper by Hinton et al. (2006).

3.8.1 Introduction

A Multi-layer Perceptron (MLP) is a feedforward artificial neural network that consists of multiple layers of interconnected nodes (neurons). It is a powerful and versatile model capable of learning complex non-linear relationships between input features and output classes. MLPs have been widely used for classification tasks across various domains and have played an influential role in the development of deep learning (Goodfellow et al., 2016).

3.8.2 Underlying Assumptions

1. **Feedforward Architecture:** MLP assumes a feedforward architecture, where information flows from the input layer, through one or more hidden layers, to the output layer. Each layer is fully connected to the next layer.
2. **Non-linearity:** MLPs introduce non-linearity through the use of activation functions at each neuron. This allows the model to learn complex decision boundaries and capture non-linear patterns in the data. Common activation functions include sigmoid, hyperbolic tangent (tanh), and rectified linear unit (ReLU) (Nwankpa et al., 2018).

3.8.3 Algorithm and Mechanism

1. **Network Architecture:** An MLP consists of an input layer, one or more hidden layers, and an output layer. The number of neurons in the input layer corresponds to the number of input features, while the number of neurons in the output layer matches the number of classes in the classification task.
2. **Forward Propagation:** During forward propagation, the input features are passed through the network. Each neuron in a layer computes a weighted sum of its inputs, applies an activation function, and passes the result to the neurons in the next layer (Bishop, 1995). This process continues until the output layer is reached.
3. **Backpropagation:** During training, the MLP uses the backpropagation algorithm to update the weights of the network. The predicted outputs are compared with the true labels, and the error is propagated backward through the network. The gradients of the loss function with respect to the weights are computed and used to update the weights in the direction that minimises the loss (Rumelhart et al., 1986).
4. **Optimisation:** The training process involves iteratively updating the weights using optimisation algorithms such as stochastic gradient descent (SGD), Adam, or RMSprop. The learning rate, batch size, and number of epochs are hyperparameters that control the optimisation process.

-
5. **Regularisation:** To prevent overfitting, regularisation techniques such as L1/L2 regularisation, dropout, and early stopping are commonly used in MLPs (Goodfellow et al., 2016). These techniques help improve generalisation performance and reduce the model’s sensitivity to noise in the training data.
 6. **Prediction:** To make predictions for a new instance, the input features are passed through the trained MLP, and the class with the highest output probability is selected as the predicted class. The softmax activation function is commonly used in the output layer for multi-class classification tasks to convert the raw outputs of the MLP into class probabilities (Goodfellow et al., 2016).

Mathematical Formulation

Let $\mathbf{x} \in \mathbb{R}^d$ be the input feature vector, and let $\mathbf{y} \in \{0, 1\}^C$ be the one-hot encoded target vector for C classes. The MLP can be represented as a function $f_{\mathbf{W}, \mathbf{b}}(\mathbf{x})$ parameterised by weights $\mathbf{W}$ and biases $\mathbf{b}$. The forward propagation through the layers of the MLP can be expressed as (Bishop, 1995):

$$\begin{aligned}
\mathbf{h}^{(1)} &= \phi(\mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)}) \\
\mathbf{h}^{(2)} &= \phi(\mathbf{W}^{(2)}\mathbf{h}^{(1)} + \mathbf{b}^{(2)}) \\
&\vdots \\
\hat{\mathbf{y}} &= \text{softmax}(\mathbf{W}^{(L)}\mathbf{h}^{(L-1)} + \mathbf{b}^{(L)})
\end{aligned} \tag{3.14}$$

where $\mathbf{h}^{(l)}$ represents the hidden activations at layer l , $\phi(\cdot)$ is the activation function, and L is the total number of layers.

The predicted class $\hat{y}$ is obtained by taking the argmax of the output vector:

$$\hat{y} = \underset{c}{\operatorname{argmax}} \hat{\mathbf{y}}_c \tag{3.15}$$

During training, the goal is to minimise the cross-entropy loss between the predicted probabilities and the true labels:

$$\mathcal{L}(\mathbf{W}, \mathbf{b}) = -\frac{1}{N} \sum_{i=1}^N \sum_{c=1}^C y_{i,c} \log(\hat{y}_{i,c}) \tag{3.16}$$

where N is the number of training instances.

3.8.4 Python Implementation of MLP Model

For the Python implementation, the `MLPClassifier` function in the `sklearn` package was used (Pedregosa et al., 2011). More details on the exact implementation can be found on the dissertation GitHub repository.

3.9 Model Evaluation Methods

To assess the performance of the six machine learning models in classifying fuel transactions based on the number of fraud indicators, an evaluation was conducted using several key metrics. These metrics provide insights into different aspects of the models' predictive capabilities and assist in selecting the most suitable model for the given task (Hossin & Sulaiman, 2015; Sokolova & Lapalme, 2009).

3.9.1 Precision

Precision is a metric that measures the proportion of true positive predictions among all positive predictions made by the model. It indicates the model's ability to avoid false positives, which is crucial in fraud detection tasks where incorrectly flagging legitimate transactions as fraudulent can have far-reaching consequences (Powers, 2020). Precision is calculated as follows:

$$\text{Precision} = \frac{TP}{TP + FP} \quad (3.17)$$

where TP represents the number of true positive predictions and FP represents the number of false positive predictions (Powers, 2020).

3.9.2 Recall

Recall, also known as sensitivity or true positive rate, measures the proportion of true positive predictions among all actual positive instances in the dataset. It showcases the model's capability to identify all relevant instances, which is important in fraud detection to minimise the number of fraudulent transactions that go undetected (Hossin & Sulaiman, 2015). Recall is calculated as follows:

$$\text{Recall} = \frac{TP}{TP + FN} \quad (3.18)$$

where TP represents the number of true positive predictions and FN represents the number of false negative predictions (Powers, 2020).

3.9.3 F1-score

The F1-score is the harmonic mean of precision and recall, providing a balanced evaluation of the model's performance. It takes into account both the model's ability to avoid false positives and its capability to identify all relevant instances. The F1-score is particularly useful when the class distribution is imbalanced, as it gives equal importance to precision and recall (Sokolova & Lapalme, 2009). It is calculated as follows:

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (3.19)$$

3.9.4 Accuracy

Accuracy is the most intuitive evaluation metric and represents the overall proportion of correct predictions made by the model across all classes. It measures the model's ability to correctly classify both positive and negative instances (Hossin & Sulaiman, 2015). Accuracy is calculated as follows:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \quad (3.20)$$

where TP represents true positives, TN represents true negatives, FP represents false positives, and FN represents false negatives (Powers, 2020).

3.9.5 Support

Support refers to the number of observations in each class within the test set. It provides information about the class distribution and helps in understanding the model's performance in the context of the available data. Imbalanced class distributions can pose challenges for machine learning models, and considering the support for each class is important when interpreting the evaluation metrics (Sokolova & Lapalme, 2009).

Chapter 4

Data Preparation

The dataset for this dissertation is an amalgamation of three different datasets: a fuel transaction dataset, a GPS tracking dataset, and a vehicle asset register. The fuel transaction dataset contains daily transaction records for the Eastern Cape government fleet from 1 April 2021 to 29 January 2022. The GPS tracking dataset covers a subset of the vehicles, and the vehicle asset register provides additional vehicle information, such as the rate card category. The data was obtained with the assistance of the Clinton Health Access Initiative (CHAI) and the Modelling and Simulation Hub, Africa (MASHA). A high-level overview of the process is shown in Figure 4.1 below.



Figure 4.1: A diagram of the high-level data preparation process.

4.1 Combining the Raw Datasets

The initial step involved merging the fuel transaction dataset with the vehicle asset register. The selected variables from the asset register included the vehicle make, model, department, and rate card category. From the fuel transaction dataset, the chosen variables were merchant name (fuel station), purchase category, number of litres, and transaction amount. The vehicle registration number served as the identifier in both datasets.

Table 4.1 below shows the first five transactions from the raw transactions dataset, with anonymised registration numbers. The full dataset has 207 702 rows and includes details such as the transaction date, merchant name, purchase category, number of litres, transaction amount, vehicle make, and model.

Transaction Date	REG_NUM	Merchant Name	Purchase Category
2021-04-01	ANON.REG.NUM.1	ULUNDI FILLING STATION	FUEL
2021-04-01	ANON.REG.NUM.2	SB DELTA	FUEL
2021-04-01	ANON.REG.NUM.2	SB DELTA	FUEL
2021-04-01	ANON.REG.NUM.3	MOUNT FLETCHER MOTORS EASTERN CAPE	FUEL
2021-04-01	ANON.REG.NUM.3	MOUNT FLETCHER MOTORS EASTERN CAPE	FUEL

No. of Litres	Transaction Amount	Make	Model
43.06	695.45	ISUZU	DMAX 250 HO DC HR 4X
31.02	478.17	ISUZU	DMAX 250 HO DC HR 4X
58.95	908.52	ISUZU	DMAX 250 HO DC HR 4X
58.61	963.66	ISUZU	DMAX 250 HO DC HR 4X
60.02	986.76	ISUZU	DMAX 250 HO DC HR 4X

Table 4.1: The first 5 transactions from the raw transactions dataset (with anonymised registration numbers).

Table 4.2 presents the first five entries from the raw fleet register dataset. The full dataset has 4 877 rows and contains information on the vehicle make, model derivative, department, and rate card category for each vehicle in the fleet.

Reg No	VEHICLE MAKE	MODEL DERIVATIVE	DEPARTMENT	RATE CARD CATEGORY
ANON.REG.NUM.1	MAN	FIRE ENGINE	TRANSPORT	MANAGED MAINTENANCE
ANON.REG.NUM.2		TRAILER BOWSER	EMERGENCY MEDICAL SERVICES	MANAGED MAINTENANCE
ANON.REG.NUM.3	VENTER	OTHER UNKOWN	HEALTH	MANAGED MAINTENANCE
ANON.REG.NUM.4	VENTER	OTHER UNKOWN	HEALTH	MANAGED MAINTENANCE
ANON.REG.NUM.5	AUDI	AUDI Q7 4.2 TDI V8 QUATTRO TIP	SOCIAL DEVELOPMENT	MANAGED MAINTENANCE

Table 4.2: The first five entries from the raw fleet register dataset (with anonymised registration numbers).

In addition to the main fleet register for all government vehicles in the Eastern Cape, a separate dataset with 3 324 rows was provided with additional information for the EMS and White fleet vehicles, as shown in Table 4.3. This dataset includes the site and district for each of those vehicles.

Site	District	Reg
Aliwal North	Joe Gqabi	ANON.REG.NUM.1
Ntabankulu	Alfred Nzo	ANON.REG.NUM.1
ALI WAL NORTH	Joe Gqabi	ANON.REG.NUM.1
Aliwal North	Joe Gqabi	ANON.REG.NUM.1
Dir: HIV/AIDS & STD MANGT *P	Alfred Nzo	ANON.REG.NUM.1

Table 4.3: The first 5 entries from the raw EMS and White fleet register dataset (with anonymised registration numbers).

After the merge, 27 631 out of the original 207 702 transactions were unaccounted for, indicating that these transactions belonged to vehicles not listed in the vehicle asset register and were potentially personal vehicles. They were removed from the dataset and investigated further at the time of writing, and were not included in the analysis.

4.2 Data Cleaning

The data cleaning process is summarised below:

- **Removal of Large Vehicles:** Vehicles such as trucks, tankers, and construction equipment were excluded to avoid skewing the analysis.
- **Purchase Category:** All non-fuel transactions (oil, tyres, general maintenance, etc.) were also removed since the analysis focuses on fuel.
- **Merchant Names:** Many merchant names were nearly identical to one another, likely due to human error. These names were combined.
- **Site Names:** Similar to the merchant names, many site names were nearly identical and were combined.
- **Transaction Amount:** Eight transactions with negative amounts were removed, as they were likely refunds and would skew the analysis.

The remaining variables appeared to be satisfactory.

4.3 Adding Transaction Coordinates

To determine the location of transactions, coordinates were added based on the merchant names (fuel station names). These names were passed to the Google Maps API to retrieve the latitude and longitude values. The activated Google Maps API key is shown below and the code that utilised it can be found in the dissertation GitHub repository.

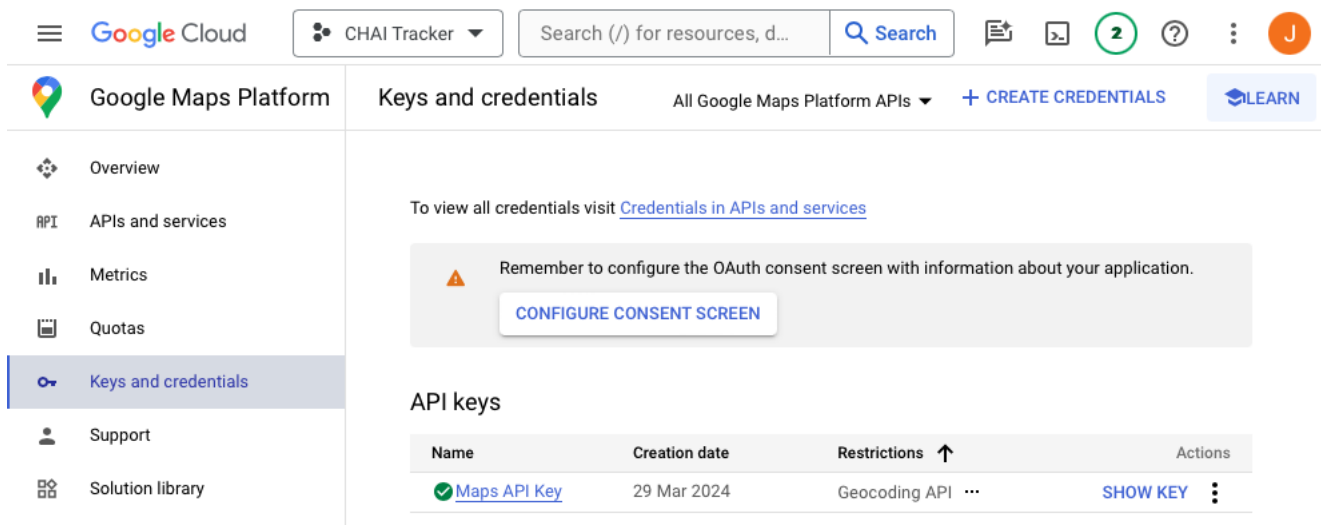


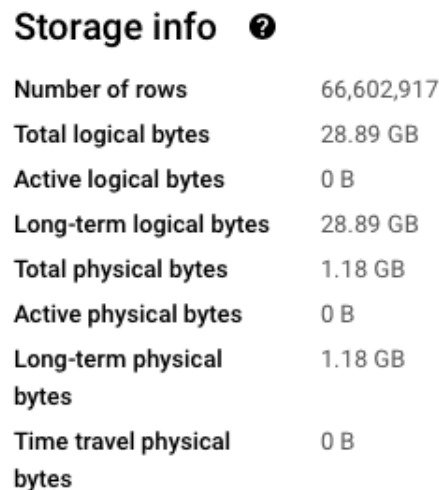
Figure 4.2: A screenshot of the Google Cloud console displaying the activated Google Maps API key to convert site and merchant names to coordinates.

4.4 Adding Government-Mandated Fuel Prices

To verify the estimated price per litre for each transaction, the official government fuel prices for the corresponding period were added to the dataset. It's important to acknowledge that the actual prices at fuel stations differed from those determined by the government due to additional costs for transport and fuel levies. To account for these differences in the analysis, fuel prices were collected from a few fuel stations in the Eastern Cape and compared to the official government prices (Table 6.1). The observed difference was then assumed to have been similar in 2021, the year from which the transaction data was obtained. This assumption allowed for a more accurate comparison between the estimated and actual fuel prices.

4.5 Creating a Vehicle-Level Dataset

The final data preparation step involved creating a second vehicle-level aggregated dataset with fuel efficiency estimates for the vehicles during the period. The GPS tracker dataset, used for estimating vehicle mileage, was too large (28.89 GB) to load into memory, as shown in Figure 4.3. To overcome this challenge, Google BigQuery was chosen for data storage and preparation.



The screenshot shows the 'Storage info' section of a Google BigQuery dataset report. It lists various storage metrics for a dataset with 66,602,917 rows. The total logical and long-term logical bytes are 28.89 GB, while the total physical and long-term physical bytes are 1.18 GB. Active logical and physical bytes are 0 B, and time travel physical bytes are also 0 B.

Storage info ⓘ	
Number of rows	66,602,917
Total logical bytes	28.89 GB
Active logical bytes	0 B
Long-term logical bytes	28.89 GB
Total physical bytes	1.18 GB
Active physical bytes	0 B
Long-term physical bytes	1.18 GB
Time travel physical bytes	0 B

Figure 4.3: A screenshot of the Google BigQuery dataset size report for the full GPS tracker dataset.

A preview of the tracker dataset in the BigQuery console is provided in Figure 4.4. To extract the final odometer readings for each vehicle for each day, a SQL query was used, as shown in Figure 4.5. The resulting dataset was then saved as a CSV file for further processing.

Row	Alias	ReportStart	VehTotalTrips	VehTotalDistance	Make
14	RURAL DEVELOPMENT AND A...	2022/06/30 22:00:00	364	6 865.8 km	Make: Fi
15	RURAL DEVELOPMENT AND A...	2022/06/30 22:00:00	364	6 865.8 km	Make: Fi
16	RURAL DEVELOPMENT AND A...	2022/06/30 22:00:00	364	6 865.8 km	Make: Fi
17	RURAL DEVELOPMENT AND A...	2022/06/30 22:00:00	364	6 865.8 km	Make: Fi
18	RURAL DEVELOPMENT AND A...	2022/06/30 22:00:00	364	6 865.8 km	Make: Fi
19	RURAL DEVELOPMENT AND A...	2022/06/30 22:00:00	364	6 865.8 km	Make: Fi
20	RURAL DEVELOPMENT AND A...	2022/06/30 22:00:00	364	6 865.8 km	Make: Fi
21	RURAL DEVELOPMENT AND A...	2022/06/30 22:00:00	364	6 865.8 km	Make: Fi
22	RURAL DEVELOPMENT AND A...	2022/06/30 22:00:00	364	6 865.8 km	Make: Fi
23	RURAL DEVELOPMENT AND A...	2022/06/30 22:00:00	364	6 865.8 km	Make: Fi
24	RURAL DEVELOPMENT AND A...	2022/06/30 22:00:00	364	6 865.8 km	Make: Fi
25	RURAL DEVELOPMENT AND A...	2022/06/30 22:00:00	364	6 865.8 km	Make: Fi
26	RURAL DEVELOPMENT AND A...	2022/06/30 22:00:00	364	6 865.8 km	Make: Fi
27	RURAL DEVELOPMENT AND A...	2022/06/30 22:00:00	364	6 865.8 km	Make: Fi
28	RURAL DEVELOPMENT AND A...	2022/06/30 22:00:00	364	6 865.8 km	Make: Fi

Figure 4.4: A screenshot of the Google BigQuery dataset preview for the GPS tracker dataset.

```

1 WITH DailyLastOdoEnd AS (
2   SELECT
3     Reg,
4     DATE(PARSE_TIMESTAMP('%Y/%m/%d %H:%M:%S', RTCDatetime)) AS Date,
5     LAST_VALUE(
6       SAFE_CAST(TRIM(VehOdometerEnd) AS FLOAT64)
7     ) OVER (
8       PARTITION BY Reg, DATE(PARSE_TIMESTAMP('%Y/%m/%d %H:%M:%S', RTCDatetime))
9       ORDER BY PARSE_TIMESTAMP('%Y/%m/%d %H:%M:%S', RTCDatetime)
10      RANGE BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING
11    ) AS LastOdoEnd
12 FROM
13   `chai-tracker.chai_gps_ds.chai_gps_table`
14 WHERE
15   VehOdometerEnd IS NOT NULL AND RTCDatetime IS NOT NULL
16   AND PARSE_TIMESTAMP('%Y/%m/%d %H:%M:%S', RTCDatetime) BETWEEN '2021-04-01' AND '2022-01-29'
17 )
18
19 SELECT DISTINCT
20   Reg,
21   Date,
22   LastOdoEnd
23 FROM

```

Figure 4.5: A screenshot of the Google BigQuery query used to get the final odometer reading at the end of each day for each vehicle.

Having collected the mileage for each vehicle after each day within the transaction data period, the process of creating the vehicle-level fuel efficiency dataset involved several steps. First, the time period for each vehicle in the fuel transaction dataset was compared with the available period from the GPS data to determine the overlapping portion. If a vehicle was not found in the GPS dataset, its fuel efficiency could not be determined.

Next, the total distance driven by each vehicle was calculated by obtaining the difference in the odometer readings for the period of overlap. Similarly, the total number of litres purchased by each

vehicle was also calculated for the same overlapping period. This ensured a fair comparison and better estimates, since the mileage range and transaction period were the same.

Finally, the fuel efficiency in kilometres per litre was estimated for each vehicle by dividing the total kilometres driven by the total number of litres purchased during the overlapping period. This process resulted in a extensive vehicle-level dataset containing fuel efficiency estimates, which could be used for further analysis and comparison.

4.6 Breakdown of the Final Transaction Dataset

Transaction Date	REG_NUM	Merchant Name	No. of Litres	Transaction Amount	VEHICLE MAKE
2021-04-01	ANON_REG_NUM.1	Ulundi Filling Station	43.06	695.45	ISUZU
2021-04-01	ANON_REG_NUM.2	Sb Delta	31.02	478.17	ISUZU
2021-04-01	ANON_REG_NUM.2	Sb Delta	58.95	908.52	ISUZU
2021-04-01	ANON_REG_NUM.3	Mount Fletcher Motors Eastern Cape	58.61	963.66	ISUZU
2021-04-01	ANON_REG_NUM.3	Mount Fletcher Motors Eastern Cape	60.02	986.76	ISUZU

MODEL DERIVATIVE	DEPARTMENT	RATE CARD CATEGORY	Site	District
D-MAX 250 HO 4X4 REGULAR CAB HI-RIDER	EMERGENCY MEDICAL SERVICES	8: LDV Ambulance	Libode	OR Tambo
D-MAX 250 HO 4X4 REGULAR CAB HI-RIDER	EMERGENCY MEDICAL SERVICES	8: LDV Ambulance	Jansenville	Sarah Baartman
D-MAX 250 HO 4X4 REGULAR CAB HI-RIDER	EMERGENCY MEDICAL SERVICES	8: LDV Ambulance	Jansenville	Sarah Baartman
D-MAX 250 HO 4X4 REGULAR CAB HI-RIDER	EMERGENCY MEDICAL SERVICES	8: LDV Ambulance	Alwal North	Alfred Nzo
D-MAX 250 HO 4X4 REGULAR CAB HI-RIDER	EMERGENCY MEDICAL SERVICES	8: LDV Ambulance	Alwal North	Alfred Nzo

Site Lat	Site Long	Merchant Lat	Merchant Long	Fuel Type	Estimated Price Per Litre	Coastal Petrol
-31.5387459	29.0284483	-30.6277369	28.2262452	Diesel	16.15	15.62
-32.943528	24.6549185	-32.2968402	26.419389	Diesel	15.41	15.62
-32.943528	24.6549185	-32.2968402	26.419389	Diesel	15.41	15.62
-30.6958414	26.7089406	-30.6914475	28.5051382	Diesel	16.44	15.62
-30.6958414	26.7089406	-30.6914475	28.5051382	Diesel	16.44	15.62

Inland Petrol	Coastal Diesel	Inland Diesel	Month Name	Weekday Name
16.32	13.5212	14.1242	April	Thursday
16.32	13.5212	14.1242	April	Thursday
16.32	13.5212	14.1242	April	Thursday
16.32	13.5212	14.1242	April	Thursday
16.32	13.5212	14.1242	April	Thursday

Table 4.4: The first 5 rows from the anonymised final fuel transaction dataset.

The first 5 rows from the final transaction dataset are shown above (Table 4.4). The dataset contains 175 933 rows and 23 columns, including:

- Transaction Date: The date when the fuel transaction occurred.
- REG_NUM: The vehicle registration number (anonymised for the write-up).
- Merchant Name: The name of the fuel station where the transaction took place.
- No. of Litres: The quantity of fuel purchased in litres.
- Transaction Amount: The total cost of the fuel transaction.
- Vehicle Make: The manufacturer of the vehicle (e.g., Isuzu).
- Model Derivative: The specific model of the vehicle.

-
- Department: The government department to which the vehicle belongs.
 - Rate Card Category: The class of the vehicle according to the rate card. For example, Sedan, LDV Ambulance, Bus etc.
 - Site: The location of the vehicle's home site (unknown for 506 vehicles out of 902).
 - District: The district where the vehicle's home site is located (unknown for 506 vehicles out of 902).
 - Site Lat: The latitude coordinate of the vehicle's home site.
 - Site Long: The longitude coordinate of the vehicle's home site.
 - Merchant Lat: The latitude coordinate of the fuel station.
 - Merchant Long: The longitude coordinate of the fuel station.
 - Fuel Type: The type of fuel purchased (Diesel or Petrol).
 - Estimated Price Per Litre: The estimated price per litre based on the transaction amount and litres purchased. It is calculated as the transaction amount divided by the number of litres purchased.
 - Coastal Petrol: The government-mandated coastal petrol price at the time of the transaction.
 - Inland Petrol: The government-mandated inland petrol price at the time of the transaction.
 - Coastal Diesel: The government-mandated coastal diesel price at the time of the transaction.
 - Inland Diesel: The government-mandated inland diesel price at the time of the transaction.
 - Month Name: The name of the month in which the transaction occurred.
 - Weekday Name: The name of the weekday on which the transaction occurred.

4.7 Breakdown of the Final Vehicle Dataset

Reg	Overlap_Start	Overlap_End	Total_Km	Total_Litres	KMPL	VEHICLE MAKE
ANON_REG.1	2021-07-01	2021-12-03	8968	748.52	11.98	TOYOTA
ANON_REG.2	2021-04-01	2022-01-25	18420	1681.21	10.95	TOYOTA
ANON_REG.3	2021-07-01	2021-12-31	11320	1145.65	9.88	TOYOTA
ANON_REG.4	2021-07-01	2021-12-31	8571	822.54	10.42	TOYOTA
ANON_REG.5	2021-04-07	2021-06-30	8016	754.15	10.62	TOYOTA

MODEL DERIVATIVE	DEPARTMENT	District	Site	Site Lat	Site Long
TOYOTA COROLLA QUEST 1.6	SOCIAL DEVELOPMENT	Unknown	Unknown	-32.2968402	26.419389
TOYOTA COROLLA QUEST 1.6	HEALTH	Alfred Nzo	Tb:Khotsong Santa Hospital *P	-32.7776735	26.6575691
HILUX SC 2.4GD6 4X4 SR MT	HEALTH	Head Office	Cd: Supply Chain Mangt *P	-32.2968402	26.419389
TOYOTA COROLLA QUEST 1.6	SOCIAL DEVELOPMENT	Unknown	Unknown	-32.2968402	26.419389
TOYOTA COROLLA QUEST 1.6	SOCIAL DEVELOPMENT	Unknown	Unknown	-32.2968402	26.419389

Fuel Type	RATE CARD CATEGORY	Total Transaction Amount	Mean Transaction Amount	Total No. of Litres	Mean No. of Litres
Petrol	1: Sedan 1400-1600cc	25911.94	647.79	1506.14	37.65
Petrol	1: Sedan 1400-1600cc	30886.45	630.33	1681.21	34.31
Diesel	11: LDV 4x4 1 ton light	42156.61	936.81	2537.83	56.39
Petrol	1: Sedan 1400-1600cc	29794.70	692.90	1695.23	39.42
Petrol	1: Sedan 1400-1600cc	36452.06	639.50	2007.25	35.21

Table 4.5: The first 5 rows from the final anonymised vehicle-level aggregated dataset.

The first 5 rows from the final vehicle dataset are shown above (Table 4.5). The dataset contains 902 rows and 19 columns. Some of the columns are the same as in the transaction dataset, such as Reg (anonymised vehicle registration number), Vehicle Make, Model Derivative, Department, District, Site, Site Lat, Site Long, Fuel Type, and Rate Card Category. The columns that differ are:

- **Overlap_Start**: The start date of the data overlap period between the transaction data and the mileage data for a specific vehicle.
- **Overlap_End**: The end date of the data overlap period between the transaction data and the mileage data for a specific vehicle.
- **Total_Km**: The total distance driven by the vehicle in kilometres during the overlap period (unknown for 13 out of 902 vehicles).
- **Total_Litres**: The total quantity of fuel purchased by the vehicle during the overlap period.
- **KMPL**: The kilometres per litre (fuel efficiency) of the vehicle, calculated as the total kilometres driven divided by the total litres of fuel purchased during the overlap period (unknown for 13 out of 902 vehicles).
- **Total Transaction Amount**: The total amount spent on fuel transactions for the vehicle for the entire transaction data period.
- **Mean Transaction Amount**: The average amount spent per fuel transaction, calculated as the total transaction amount divided by the number of transactions for the vehicle for the entire transaction data period.
- **Total No. of Litres**: The total quantity of fuel purchased in litres for the vehicle for the entire transaction data period.

-
- Mean No. of Litres: The average number of litres purchased per fuel transaction, calculated as the total number of litres purchased divided by the number of transactions for the vehicle for the entire transaction data period.

Chapter 5

Results: Data Exploration

5.1 Exploratory Data Analysis (EDA)

Before the creation of indicators, it is essential to thoroughly examine the data to gain a more complete understanding of the underlying patterns. This section is divided into two primary subsections: univariate and bivariate EDA. The univariate exploration focuses on analysing each variable independently, while the bivariate exploration investigates the relationships and interactions between pairs of variables.

5.1.1 Univariate EDA

The fleet composition primarily comprises Toyota, Volkswagen (VW), and Isuzu vehicles, with fewer than 50 out of the 902 vehicles falling outside of the top 5 manufacturers: Toyota, VW, Isuzu, Nissan, and Ford (Figure 5.1a). This fleet composition is sensible considering the Eastern Cape's poor infrastructure, as it streamlines the maintenance process by ensuring that spare parts are readily available. When analysing transaction volumes (Figure 5.1b), the same top three makes emerge, although in a different order, with VW vehicles generating the highest number of transactions, potentially due to higher usage in urban areas.

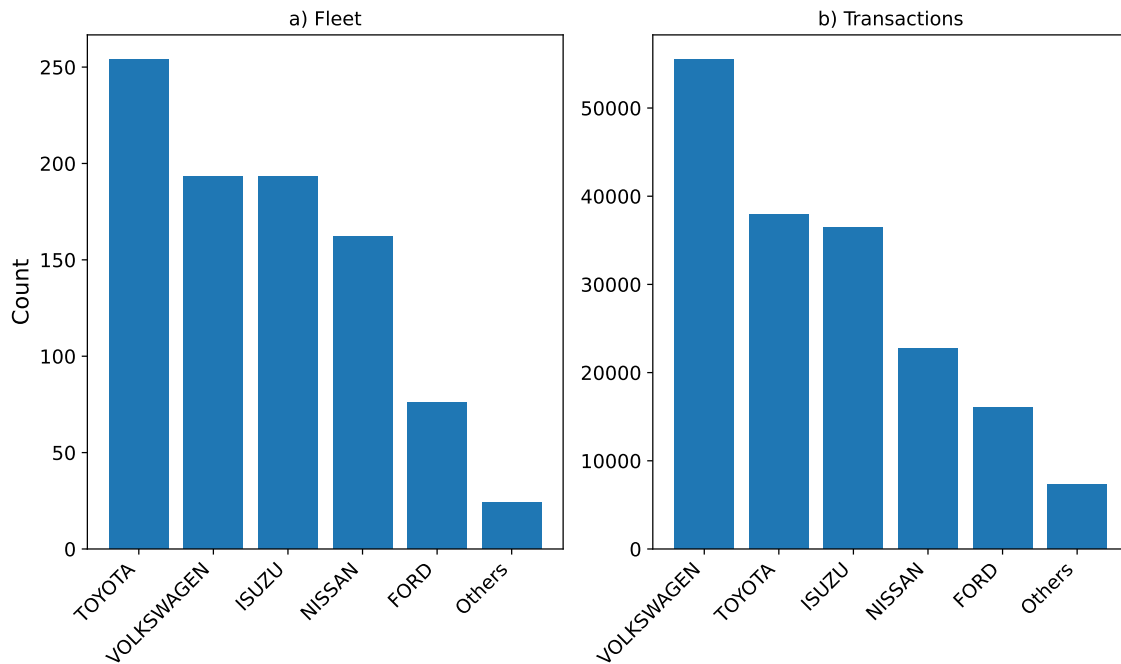


Figure 5.1: Bar plots of the top counts per make for the a) fleet and b) transactions.

The aforementioned patterns become more evident when examining the model derivatives (Figure 5.2a). The VW Polo emerges as the most prevalent vehicle, with approximately 120 cars, closely followed by the Toyota Corolla Quest, which accounts for just over 100. As previously mentioned, it is plausible that these vehicles are primarily utilised in urban settings and therefore may experience higher usage compared to vehicles deployed in more rural areas, thus contributing to their increased transaction volume (Figure 5.2b).

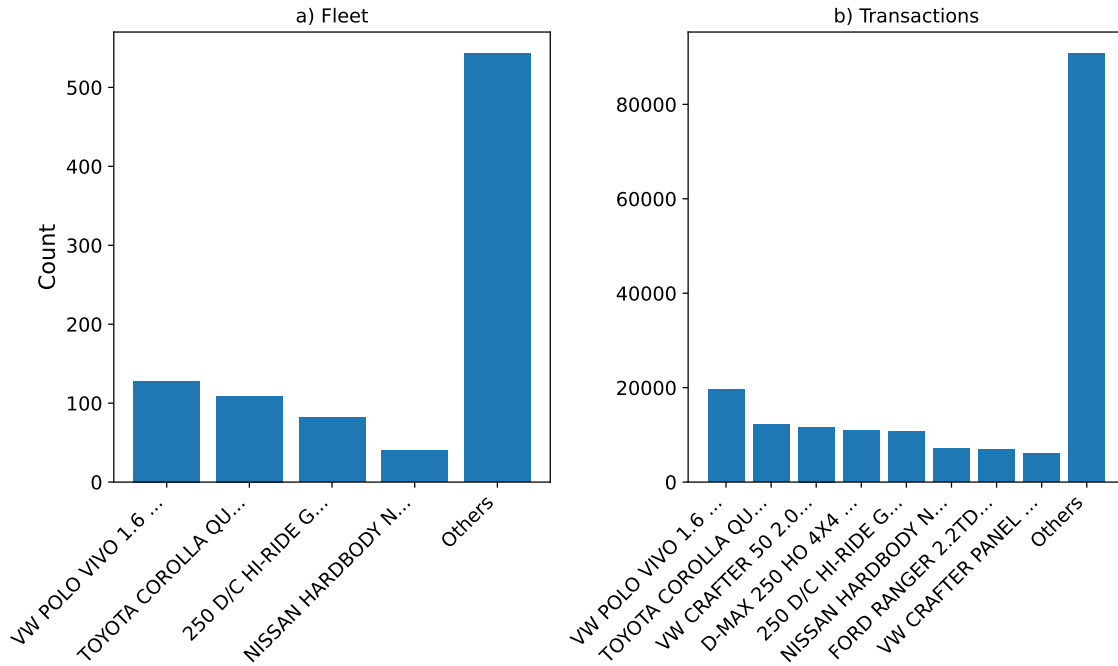


Figure 5.2: Bar plots of the top counts per vehicle derivative for the a) fleet and b) transactions.

For the fleet and transaction counts across rate card categories, the majority of vehicles belong to the sedan and light-duty vehicle (LDV) classes (Figure 5.3a). The LDV category primarily comprises various pick-up trucks (bakkies) that serve a wide range of use cases across different departments. When considering transactions (Figure 5.3b), the "Managed Maintenance" category is assigned to vehicles undergoing maintenance. However, this may not accurately reflect the vehicle category at the time of the fuel transactions, as the fleet asset register used for this analysis is from a later date. Apart from this, the remaining category counts align with the patterns mentioned above.

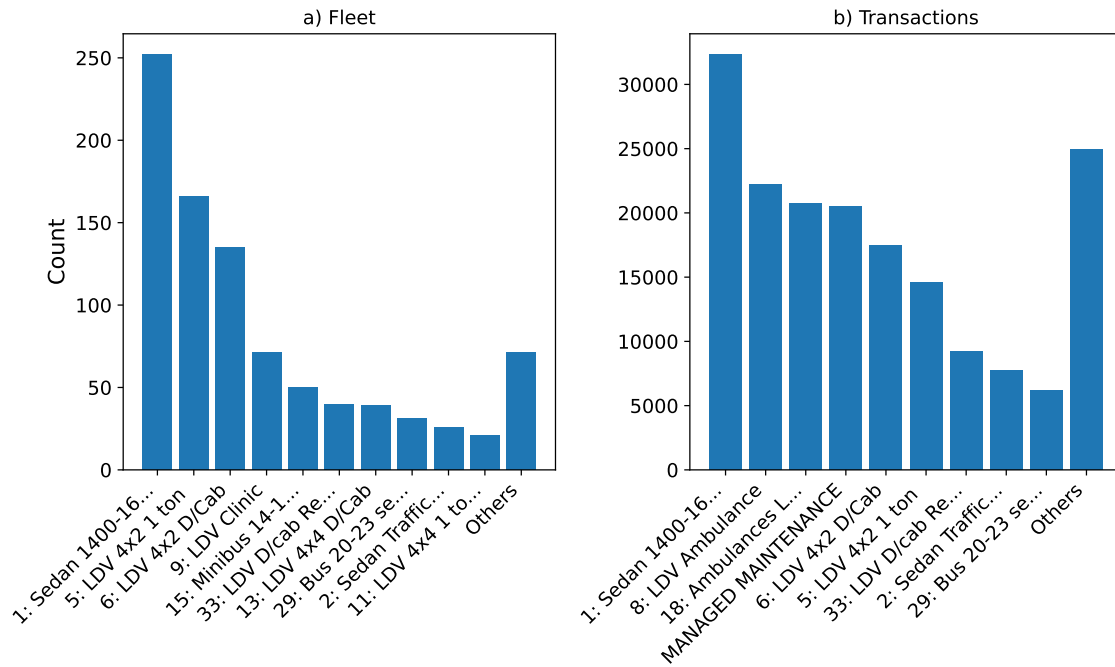


Figure 5.3: Bar plots of the top counts per rate card category for the a) fleet and b) transactions.

Analysis of the vehicle and transaction counts across departments reveals that the Department of Health, which includes Emergency Medical Services (EMS), possesses the largest fleet, with over 400 vehicles (Figure 5.4a). EMS is recorded separately due to its specialised vehicles belonging to a different class. The Department of Social Development has the second-largest fleet, with approximately 140 vehicles. Notably, EMS vehicles account for the highest number of transactions, exceeding 60 000, while the rest of the Health Department vehicles recorded around 40 000 transactions (Figure 5.4b). This disparity in transaction numbers, despite EMS being a subset of the Department of Health, highlights the intensive use of EMS vehicles. Overall, the Department of Health is responsible for the majority of the transactions, which could also be attributed to the increased demand for health services during the ongoing COVID-19 pandemic.

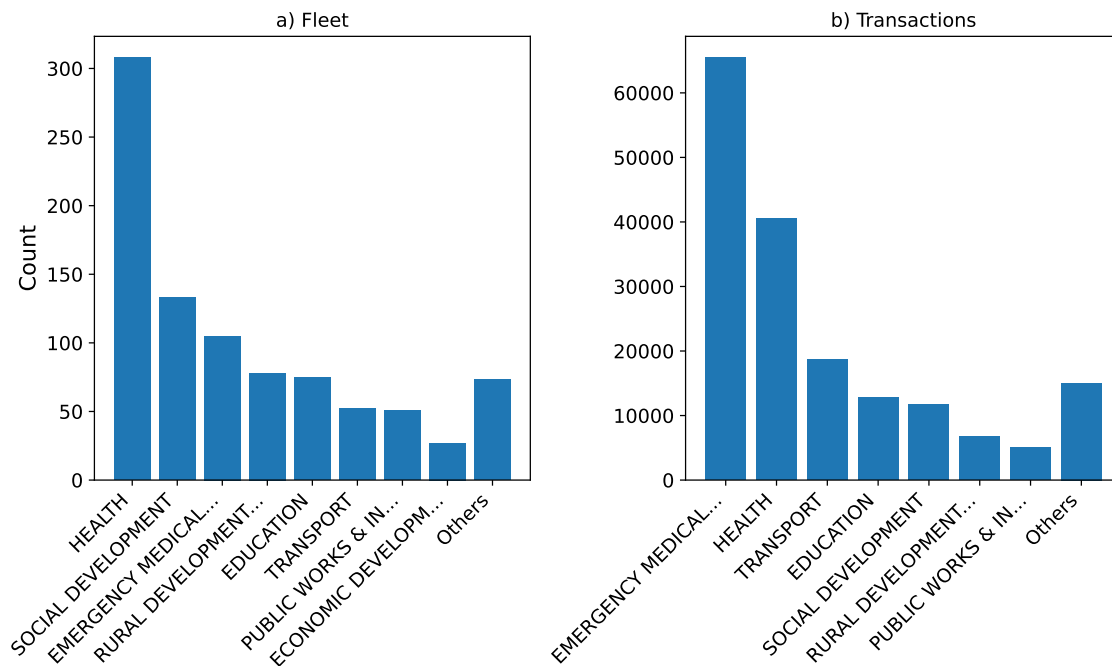


Figure 5.4: Bar plots of the top counts per department for the a) fleet and b) transactions.

Transaction timing patterns also yield intriguing insights (Figure 5.5a). The middle of the week emerges as the most popular refuelling period, with Tuesday, Wednesday, and Thursday exhibiting similar transaction counts exceeding 30 000. Monday and Friday follow closely, with approximately 27 000 transactions each, while weekends experience lower activity. This pattern may align with the operational requirements of EMS vehicles, which may need to be utilised throughout the week, including weekends, while the rest of the fleet is not being used. Days on which fuel prices change might also be a factor to consider.

Furthermore, monthly transaction counts reveal that the fourth quarter of the 2021 year experienced the highest transaction volumes (Figure 5.5b). October and November 2021 recorded the highest transaction counts, with around 20 000 each, followed by August and September. Intriguingly, January 2022 and April 2021 exhibited markedly lower counts, potentially due to reduced vehicle requirements during the pandemic. December also falls within the holiday season, so many regular operations may slow down during this period.

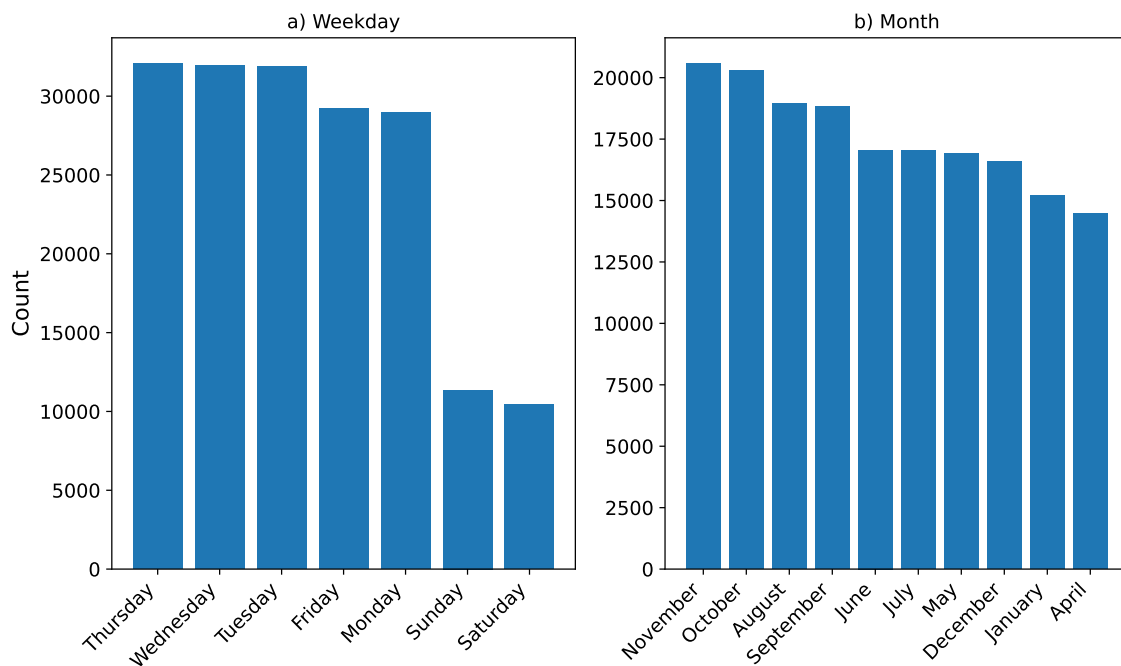


Figure 5.5: Bar plots of the top transaction counts over the a) weekdays and b) months.

The home districts and sites to which the vehicles were assigned also unearth interesting patterns, although it must be acknowledged that district and site information was unavailable for a substantial portion of the fleet (Figure 5.6a). Among the vehicles with known district data, OR Tambo (61 vehicles), Chris Hani (59 vehicles), and Alfred Nzo (50 vehicles) were allocated the most vehicles, which is understandable given their larger populations and the need for a diverse range of vehicles to serve these regions. Nelson Mandela Metro (49 vehicles), Sarah Baartman (49 vehicles), Amathole (39 vehicles), Buffalo City (34 vehicles), and Joe Gqabi (24 vehicles) follow in terms of vehicle allocation. The "Other" category, which contains all districts and geographic groupings outside of the top 8, excluding all vehicles with unknown district information (506 vehicles), accounts for a relatively small number of vehicles, including:

- Nelson Mandela (16 vehicles)
- Head Office (11 vehicles)
- EMS College (4 vehicles)

Additionally, site allocations are highly fragmented, with numerous sites having only a few vehicles allocated to each, making it challenging to identify differences between them (Figure 5.6b). However, it is worth noting that East London and Port Elizabeth (PE), being cities with higher populations, appear to have the most vehicles allocated to them, although this may not be representative of the total number of vehicles allocated to those areas due to the large number of vehicles with unknown site information.

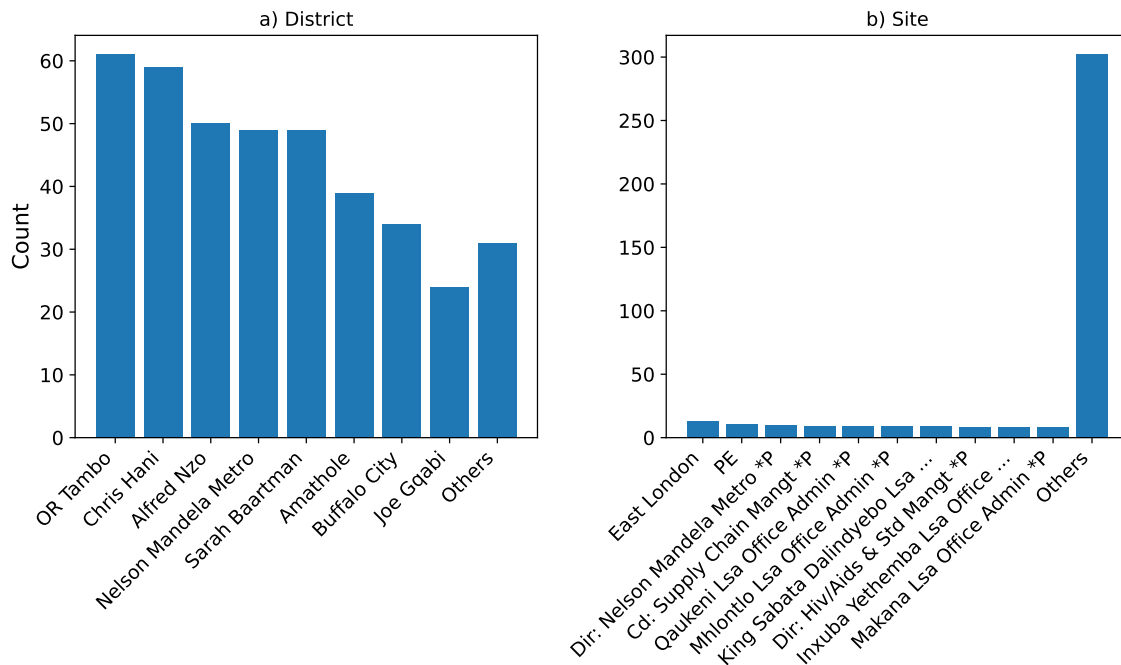


Figure 5.6: Bar plots of the number of vehicles per district and site.

Pivoting to the merchant and fuel type distributions, Total Savoy Mthatha emerges as the merchant with the highest transaction count, followed by Smith & Wiggil-South Queens, Total Oxford East London, and Masepa Motors Cc (Figure 5.7a). The prominence of these merchants suggests that they are strategically located in high-traffic areas or along frequently travelled routes, catering to a sizeable portion of the government fleet’s refuelling needs. Furthermore, the presence of well-known brands such as Total and Smith & Wiggil indicates a preference for established, reliable fuel providers. However, it is worth noting that there seems to be a wide variety of refuelling locations, with no fuel stations dominating the rest, indicating that the fleet appears to be geographically spread out across the Eastern Cape or that there is a substantial amount of travel taking place.

The breakdown of fuel types reveals that diesel is the predominant choice for the Eastern Cape government fleet, with transaction counts markedly higher than those of petrol (Figure 5.7b). This preference for diesel vehicles aligns with the fleet composition mentioned earlier, which includes a high proportion of larger, more heavy-duty vehicles such as buses and utility vehicles. Moreover, diesel vehicles are generally more fuel-efficient than their petrol counterparts (Hearst Autos Research (2024)), which may further contribute to their prevalence in the fleet.

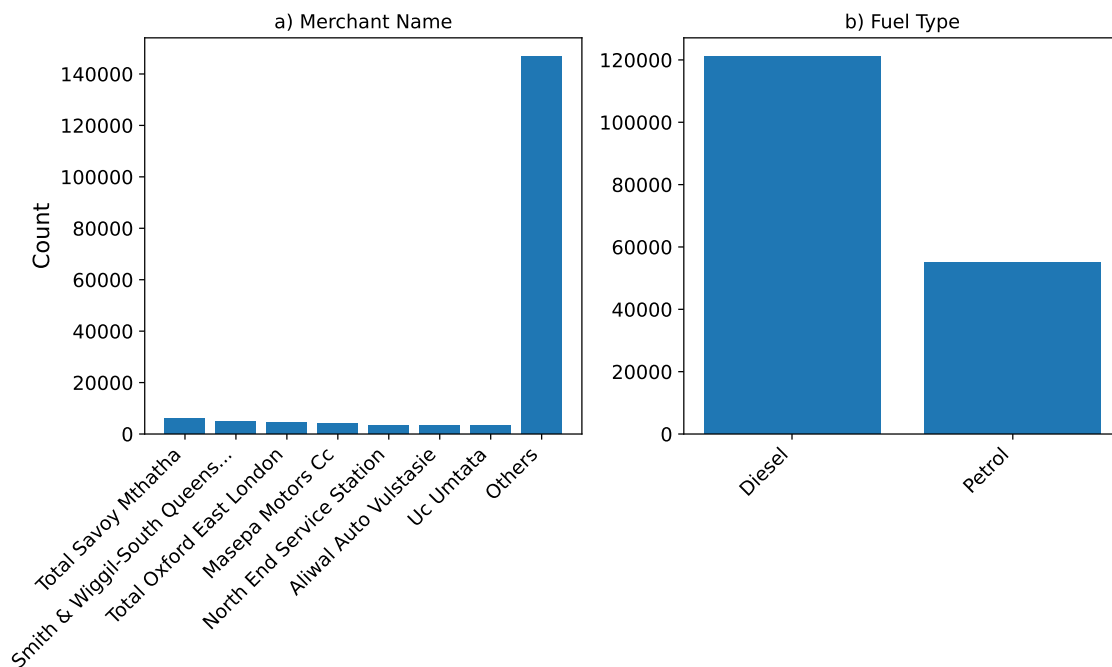


Figure 5.7: Bar plots of the top transaction counts over the a) merchants and b) fuel types.

Transitioning to the time intervals between transactions, the histogram of days between transactions grouped by vehicles (Figure 5.8a) reveals a positively skewed distribution, with a peak at around 10 days. This indicates that a substantial portion of the fleet refuels relatively frequently, likely due to high usage or the need to maintain consistent fuel levels. However, the long tail of the distribution also suggests that some vehicles have much longer intervals between transactions, possibly indicating lower usage or more efficient fuel consumption. These longer intervals could also be attributed to vehicles operating in remote areas with limited access to refuelling stations or those undergoing maintenance or repairs.

The histogram of average days between transactions for each vehicle (Figure 5.8b) provides further insights into refuelling patterns. The distribution is skewed to the right, with a peak around 5-10 days, implying that, on average, vehicles in the fleet refuel every week. This might align with typical usage patterns for government vehicles, but it warrants further investigation. The presence of vehicles with much higher average intervals between transactions, extending beyond 60 days, raises questions about their usage. These outliers could represent vehicles with exceptionally low usage, data entry errors, or other anomalies, and they should be examined more closely to determine the underlying reasons for such deviations from the norm.

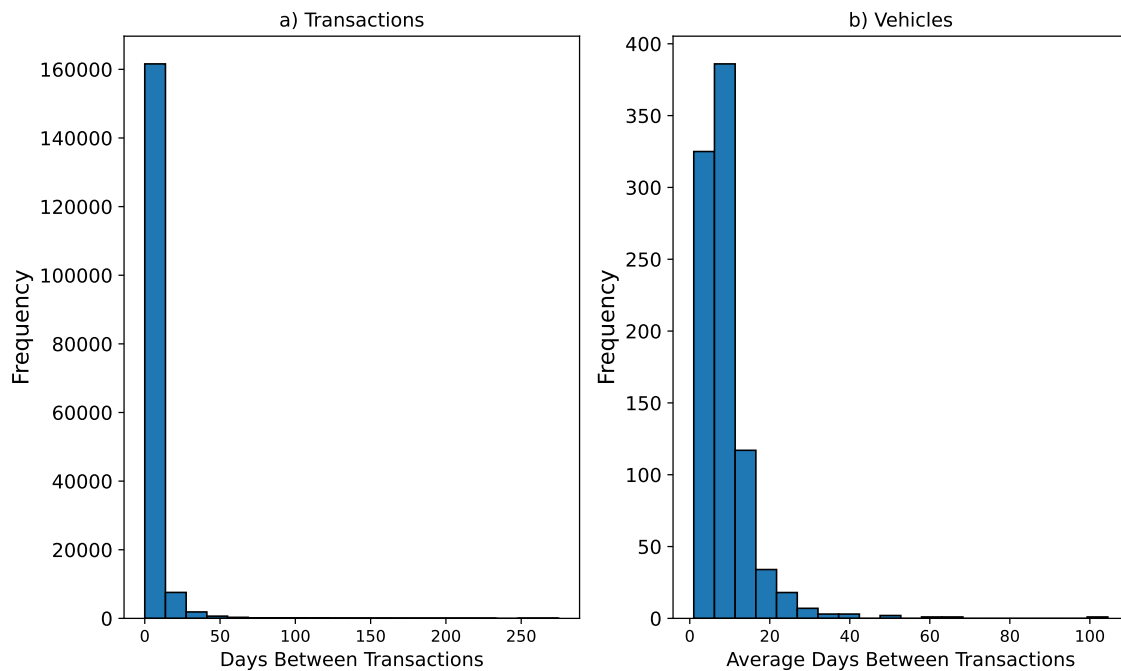


Figure 5.8: Histograms of the a) days between transactions grouped by vehicles for the transaction dataset and b) average days between transactions for each vehicle.

Finally, the distributions of transaction amounts and the number of litres across all transactions provide valuable insights into refuelling patterns (Figure 5.9a). The plots indicate that most transaction amounts have a median of around R900, with a relatively symmetric distribution. Transactions rarely exceed R1 500 or fall below R500, suggesting that most transactions are for full refills rather than top-ups, considering the most common vehicles being refuelled (sedans typically have tank sizes ranging from 45 to 55 litres, while pickups tend to have tanks around 80 litres). This observation is reinforced by the histogram of the number of litres (Figure 5.9b), which shows that most transactions centre around 50 litres, indicating that a substantial portion of the tank is being filled each time. The number of litres per transaction rarely falls below 25 or exceeds 75, implying a habit of periodic full refills rather than frequent top-ups with small quantities of fuel. Considering the frequency of refills, it is worth highlighting that most vehicles are being nearly completely refilled once a week, implying very high usage.

Furthermore, the histogram of kilometres per litre (KPL) provides a critical perspective on the fuel efficiency of the vehicles in the fleet. The distribution appears to be slightly right-skewed, with a peak around 10 KPL and a tail extending towards higher values (Figure 5.9c). This suggests that while most vehicles in the fleet have a fuel efficiency close to 10 KPL, there are some vehicles that achieve considerably higher efficiency, possibly due to factors such as vehicle type, age, or driving conditions. However, it is necessary to consider that the frequency of vehicles with exceptionally high KPL values (above 15) is relatively low, indicating that these instances are more likely to be outliers rather than the norm. Another concern is the vehicles with less than 5 KPL. This could be due to limited GPS data for those vehicles, or it could be indicative of very inefficient usage or even fuel theft, and will need to be investigated further.

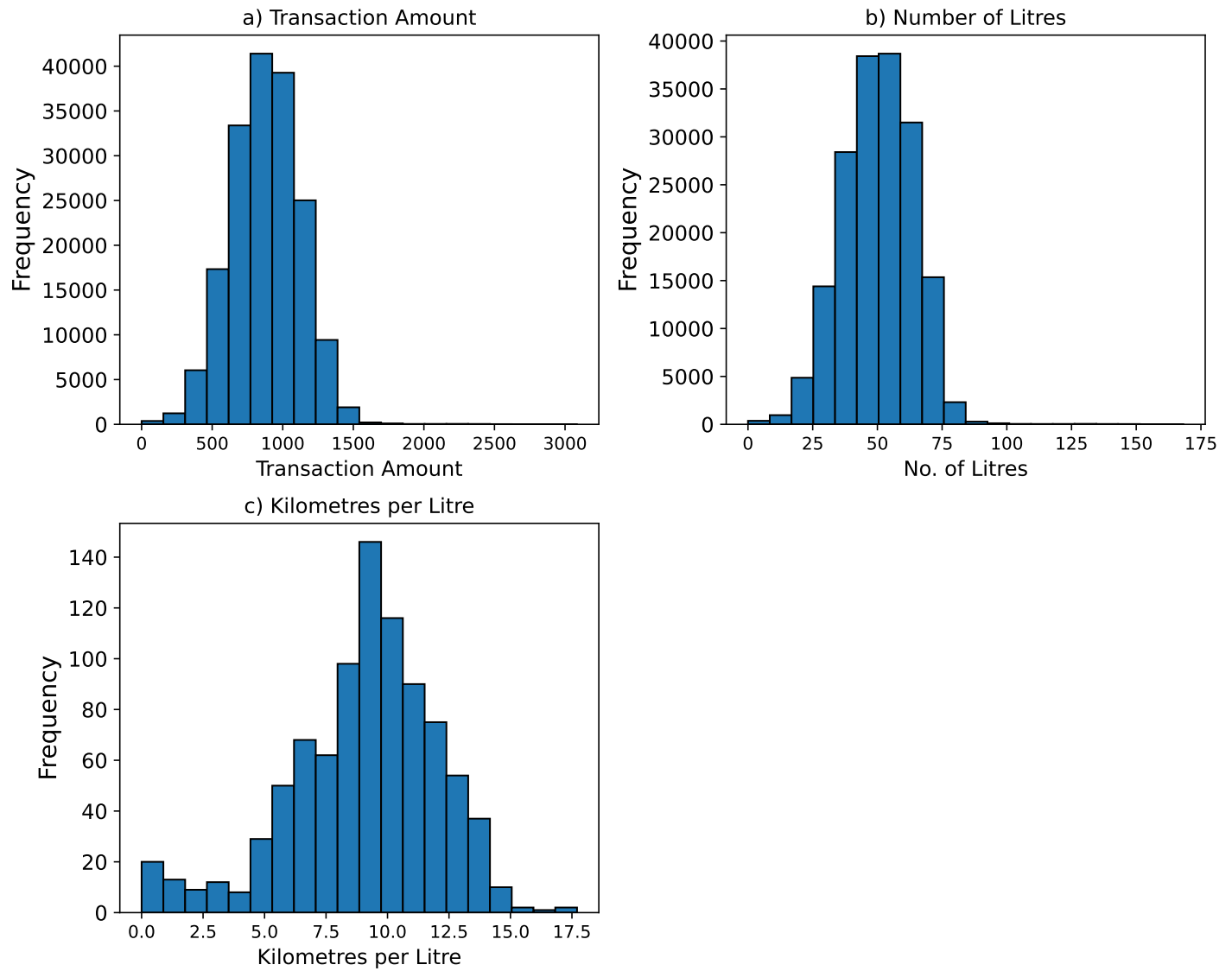


Figure 5.9: Histograms of the a) transaction amounts, b) number of litres for all transactions and c) kilometers per litre for all vehicles.

5.1.2 Bivariate EDA

The bivariate exploration reveals patterns in the average transaction amounts and litres across different vehicle makes (Figure 5.10). Mercedes-Benz vehicles have the highest average transaction amounts, exceeding R1 000. Ford and Jeep follow, with all three top makes spending on average over R800 per visit and refuelling with more than 50 litres. This indicates that sedans are almost always refilled completely, while pick-ups are replenished by more than 60% of their tank capacity each time. In contrast, the average transaction amounts and litres for all other vehicles are lower compared to these top vehicle makes.

Another observation is the presence of luxury vehicle makes, such as Mercedes and Audi, in a government fleet register. These luxury makes have high fuel consumption and higher relative average transaction amounts compared to the other makes, which raises questions about their utility and justification for government operations based on these factors.

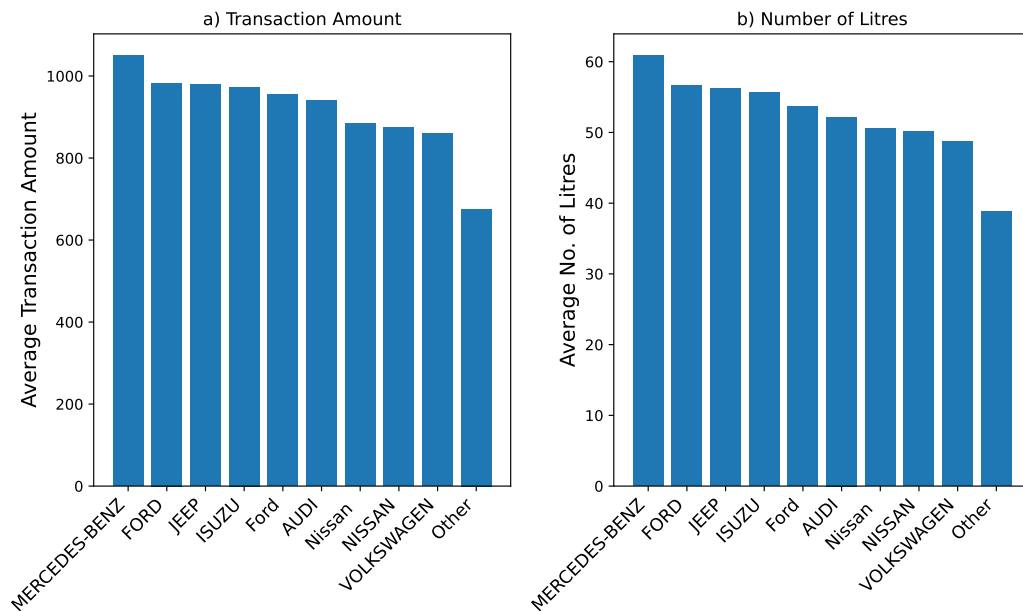


Figure 5.10: Bar plots of the average transaction amount and average number of litres for each vehicle make.

Building upon the previous observations, the model derivatives are explored. As seen below, most of the derivatives are either pick-ups or SUVs, yet the average number of litres filled up per transaction for the top 3 derivatives surpasses the standard maximum capacity of around 90L for these vehicles (Figure 5.11). This finding would raise concerns if these vehicles do not have an extra tank equipped or a larger capacity as an add-on feature. However, both the Land Cruiser 79 and Prado possess this option, likely explaining the observed pattern. The other vehicles appear to fall within the standard tank range for their vehicle classes, further reinforcing the habit of complete refills during each visit.

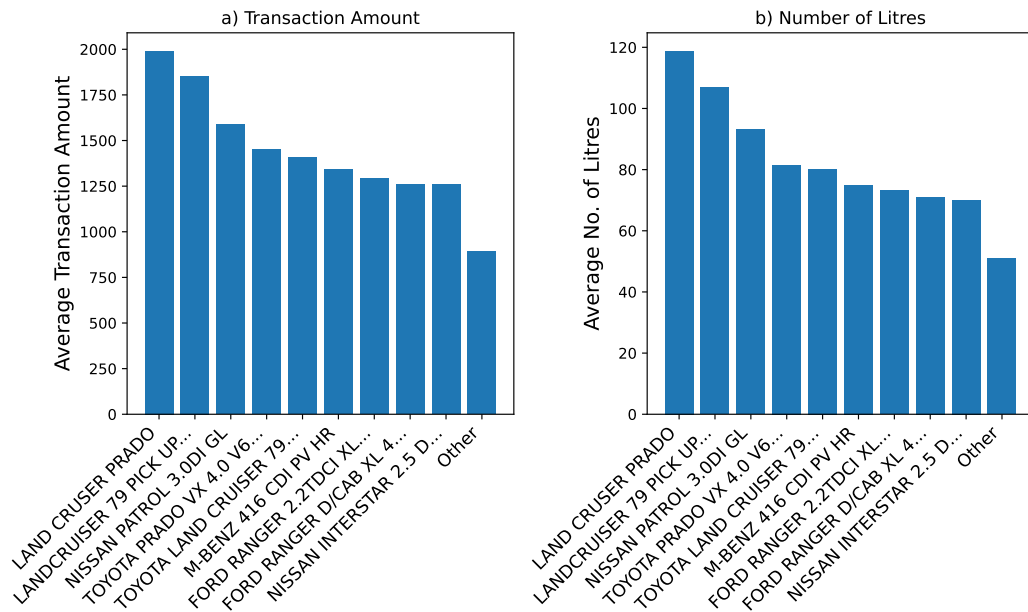


Figure 5.11: Bar plots of the average transaction amount and average number of litres for each model derivative.

Analysing the transaction amounts and number of litres across vehicle categories reveals that their general patterns mirror each other (Figure 5.12). The only standout category is the LDV 4x4 segment, which comprises large patrol vehicles with substantial tank capacities, such as the Land Cruiser Prado and Land Cruiser 79 derivatives, as previously discussed.

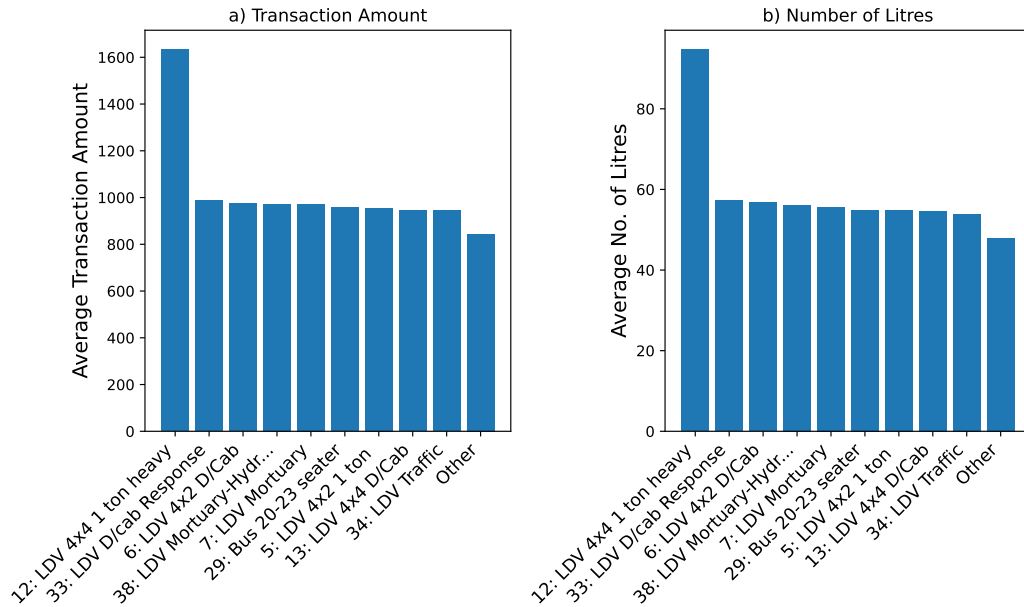


Figure 5.12: Bar plots of the average transaction amount and average number of litres for each rate card category.

The plots of the average transaction amount and average number of litres over different subsets of days between refuelling reveal a minor difference in refuelling patterns based on the time interval between transactions (Figure 5.13). For refills occurring more than 30 days apart, the average transaction amount and number of litres are only marginally higher (less than R100 and about 5 litres, respectively) compared to refills happening less than 2 days apart. This observation reinforces the previous findings, suggesting that vehicles tend to fill up most of their tank capacity during each refuelling event, regardless of the time elapsed since the last refill.

However, a closer examination of the transaction frequency raises concerns. The data indicates that a large number of transactions (68 849) occur less than 2 days apart, implying that some vehicles are refilling their tanks at least twice a week. This high refuelling frequency is alarming and may be suggestive of potential fuel theft or misuse. Such a pattern warrants further investigation to identify the underlying causes and develop appropriate countermeasures to prevent any fraudulent activities or inefficiencies in fuel consumption.

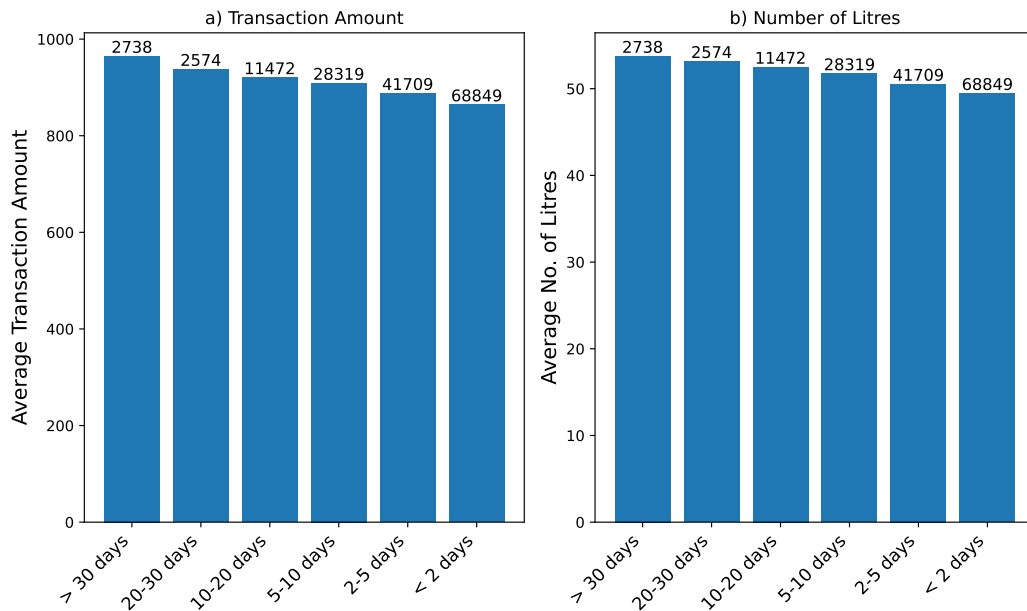


Figure 5.13: Bar plots of the average transaction amount and average number of litres over different subsets of days between refuelling with the number of transactions in each category above each bar.

Reinforcing the above result is the fact that there seems to be no substantial difference in the overall distributions of the litres purchased between consecutive transactions across different categories of days between transactions (Figure 5.14a). Transactions occurring less than 2 days apart have roughly the same median value as all the other categories, at around 50 litres, with an inter-quartile range between 40 and 60 litres. This further suggests that the vehicles are either heavily utilised or there might be fraudulent behaviour occurring, as the refuelling patterns remain consistent regardless of the time elapsed between transactions.

Moreover, the median difference in litres purchased for all day categories is nearly 0, with the inter-quartile range falling within 10 litres in either direction (Figure 5.14b). This finding further indicates that the refuelling pattern is not consistent with top-ups, but rather with full or near-full refills. The lack of variation in the difference between consecutive transactions, irrespective of the time between refuelling events, raises questions about potential fuel misuse or fraudulent activities.

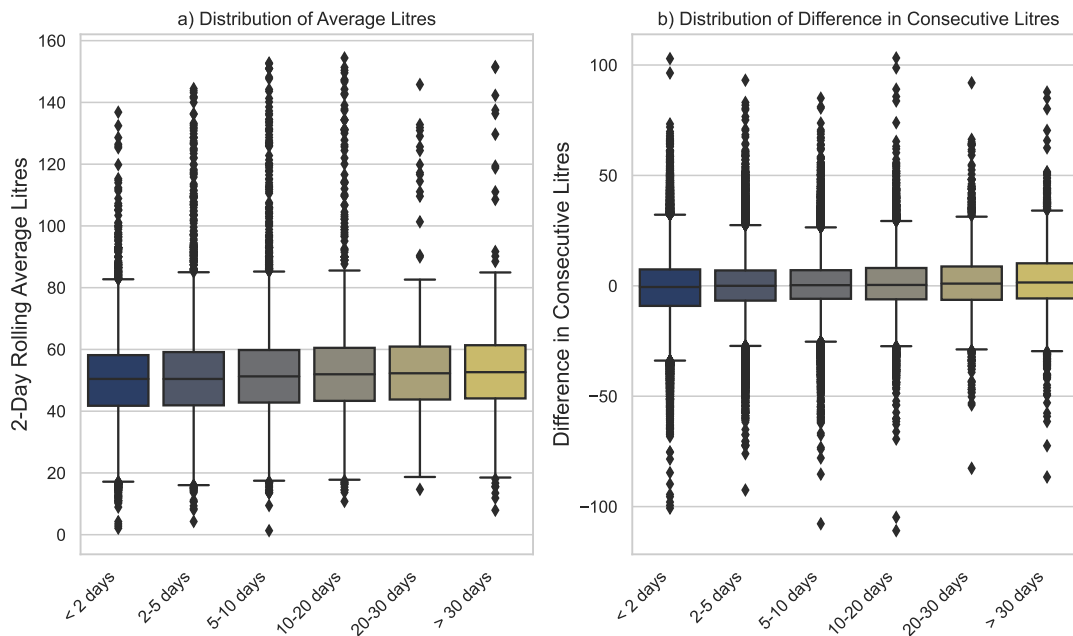


Figure 5.14: Box plots displaying the distribution of the 2-day rolling average number of litres (the average number of litres purchased for every two consecutive purchases per vehicle) over different subsets of days between refuelling.

Examining the number of litres purchased over months, the values remained relatively stable (Figure 5.15). However, the average transaction amounts exhibited fluctuations. This variation is most likely attributable to changes in fuel prices over time, as well as the predominant usage of vehicles with larger or smaller tank capacities during specific periods. Furthermore, there appears to be no marked difference in the average transaction amounts or number of litres across weekdays (Figure 5.16).

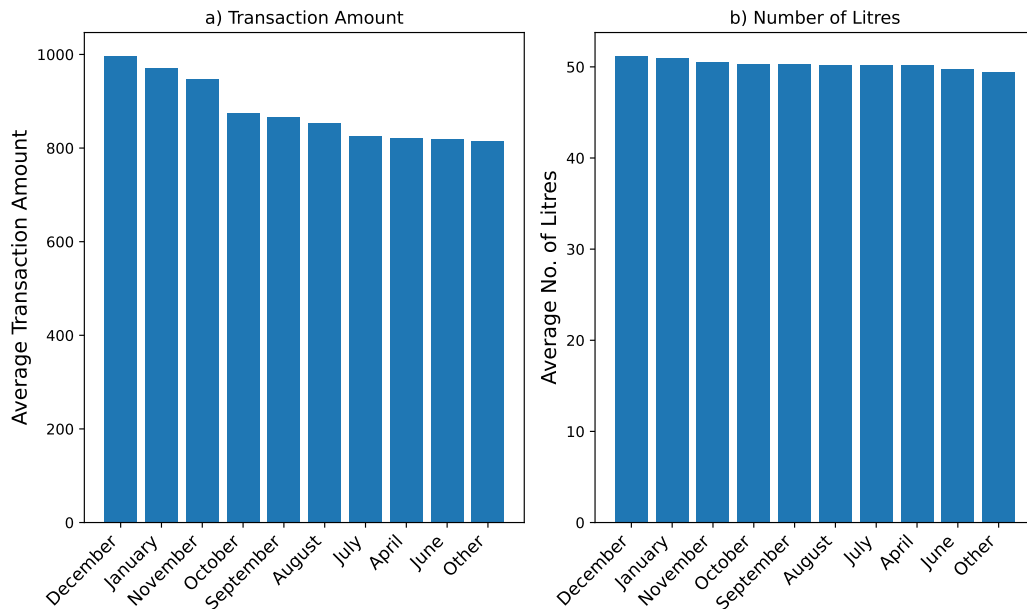


Figure 5.15: Bar plots of the average transaction amount and average number of litres for each month.

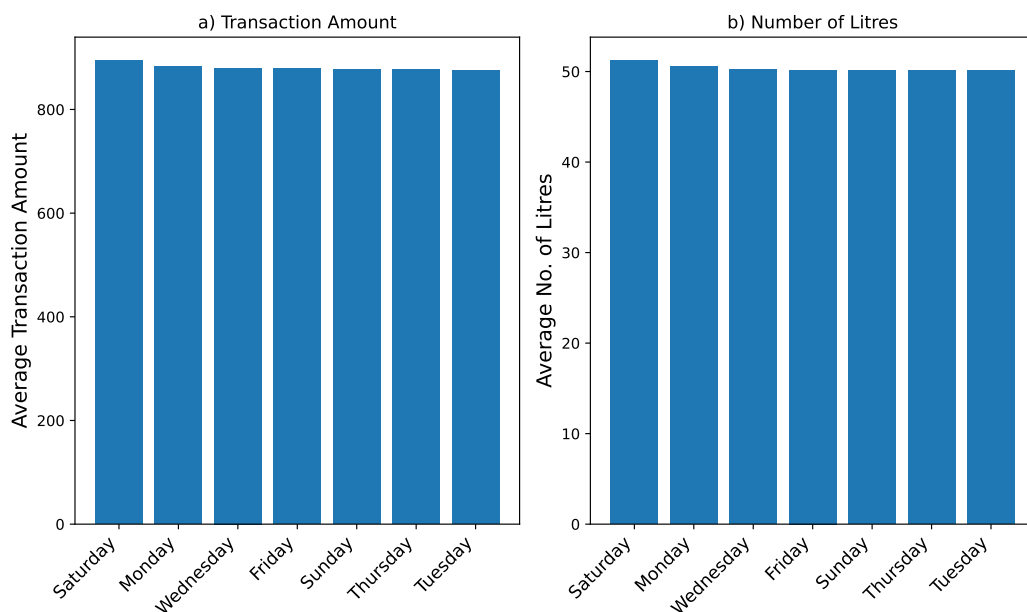


Figure 5.16: Bar plots of the average transaction amount and average number of litres for each weekday.

Building upon the previous observations in the univariate analysis, the box plots below provide a more granular view of the fuel efficiency (kilometres per litre, or KPL) across different vehicle model derivatives and rate card categories (Figure 5.17a). The plot for model derivatives reveals that the Toyota Corolla Quest and VW Polo Vivo have the highest median KPL values, hovering around 11.5 and 12.5, respectively. These sedan models, known for their fuel efficiency, have relatively narrow interquartile ranges, suggesting a consistent performance across vehicles of the same model.

In contrast, the Isuzu D-Max and Nissan Hardbody pickup trucks exhibit lower median KPL values, around 7.5 and 8.5, respectively, with similar interquartile ranges as the other top model derivatives (Figure 5.17a). The "Other" category, which includes a diverse mix of vehicle models outside of the most popular ones in the fleet, has a median KPL similar to the pickups but with a much wider spread, encompassing both highly efficient and less efficient vehicles outside of the top 5 most popular derivatives.

Shifting focus to the rate card categories, the box plot reveals that sedans (category 1) have the highest median KPL, aligning with the observations from the model derivative plot (Figure 5.17b). Light-duty vehicles (LDVs) in categories 5 and 6, which include 4x2 1-ton and 4x4 double-cab variants, have lower median KPL values, consistent with the performance of the Isuzu D-Max and Nissan Hardbody models. Interestingly, the LDV Clinic category (9) has a substantially lower median KPL compared to other LDV categories, possibly due to the specific operational requirements and driving patterns of these vehicles.

Finally, the minibus category (15) exhibits a relatively wide interquartile range, suggesting varied fuel efficiency across vehicles in this group (Figure 5.17b). This could be attributed to factors such as passenger load, route characteristics, and vehicle maintenance. As with the model derivative plot, the "Other" category in the rate card plot encompasses a broad range of KPL values, reflecting the diversity of vehicle types and their respective fuel efficiencies outside of the most popular categories.

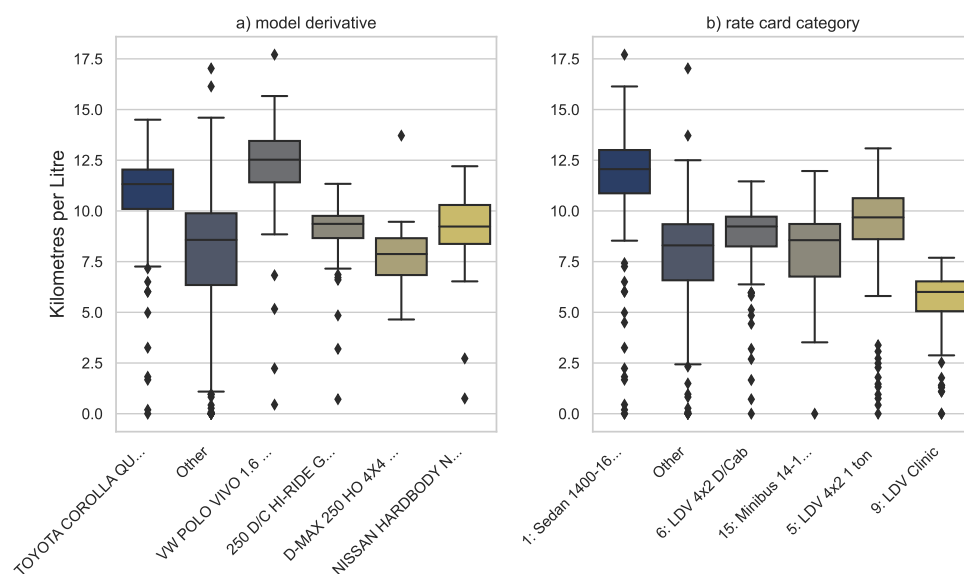


Figure 5.17: Box plots of the kilometres per litres for the top a) model derivatives and b) rate card categories.

The geographical distribution of fuel transactions within the Eastern Cape province (Figure 5.18) aligns with the findings from the count plots of the merchants in the univariate analysis. The clustering of transactions in major cities and towns, such as East London, Gqeberha, and Mthatha (also known as Umtata), corroborates the prominence of merchants such as Total Savoy Mthatha, Smith & Wiggil-South Queens, and Total Oxford East London. This convergence of evidence supports the notion that these merchants are strategically positioned in high-traffic areas, catering to the refuelling needs of a sizeable portion of the government fleet. Furthermore, the transaction locations coincide with the population centres of the respective districts, indicating that the fleet's refuelling patterns are closely tied to the demographic distribution and economic activity within the province.

Expanding the geographical scope to include all of South Africa reveals a pattern of refuelling locations outside the Eastern Cape province (Figure 5.19). The presence of transactions in major cities such as Johannesburg and Cape Town, as well as along the routes connecting them to the Eastern Cape, suggests that some vehicles in the government fleet occasionally undertake long-distance trips for various purposes. These trips could be related to administrative work, such as attending meetings or conferences, or for logistical reasons, such as procuring supplies or equipment.

Moreover, the timing of these out-of-province transactions during the COVID era raises the possibility of medical-related travel. The Eastern Cape faced considerable challenges in managing the pandemic, and the need to secure medical supplies or collaborate with healthcare facilities in other parts of the country may have necessitated these long-distance journeys.

Figure 5.18: Map of transaction locations within the Eastern Cape.



Figure 5.19: Map of transaction locations within South Africa.

5.2 Clustering

5.2.1 Vehicle Clustering

Before developing indicators and predictive models, it is beneficial to investigate how vehicles cluster based on their usage patterns. This analysis will be conducted using the k-means++ and agglomerative clustering algorithms.

Tuning

The optimal number of clusters must be determined for both methods (Figure 5.20). For the k-means++ and agglomerative clustering methods, three and four clusters were selected, respectively. Although there was no definitive elbow point for k-means++, three clusters was the most suitable choice, as the decrease in within-cluster sum of squares became less pronounced after that point.

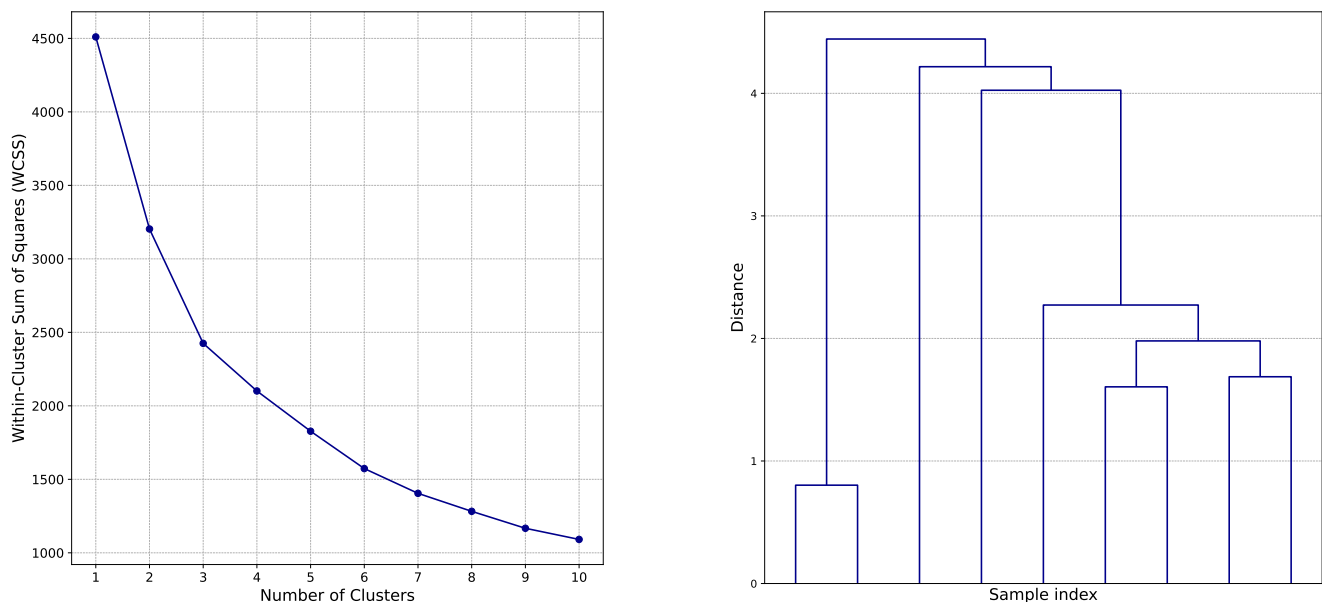


Figure 5.20: (Left) A line plot of within-cluster sum of squares over clusters to determine the optimal number of clusters for the k-means++ algorithm. (Right) A dendrogram to determine the number of clusters for agglomerative clustering.

Combined Results

The plot below shows the counts of each cluster for the k-means++ and agglomerative clustering approaches (Figure 5.21). While these counts are primarily intended to guide the interpretation of the following plots, it is useful to have a prior understanding of the cluster sizes. In the k-means++ algorithm, clusters 3 and 1 each contain nearly double the number of vehicles (approximately 360 for cluster 3 and 340 for cluster 1) compared to cluster 2 (approximately 170 vehicles). On the other hand, the agglomerative clustering method yields more balanced cluster sizes, ranging from approximately 160 to 270 vehicles per cluster.

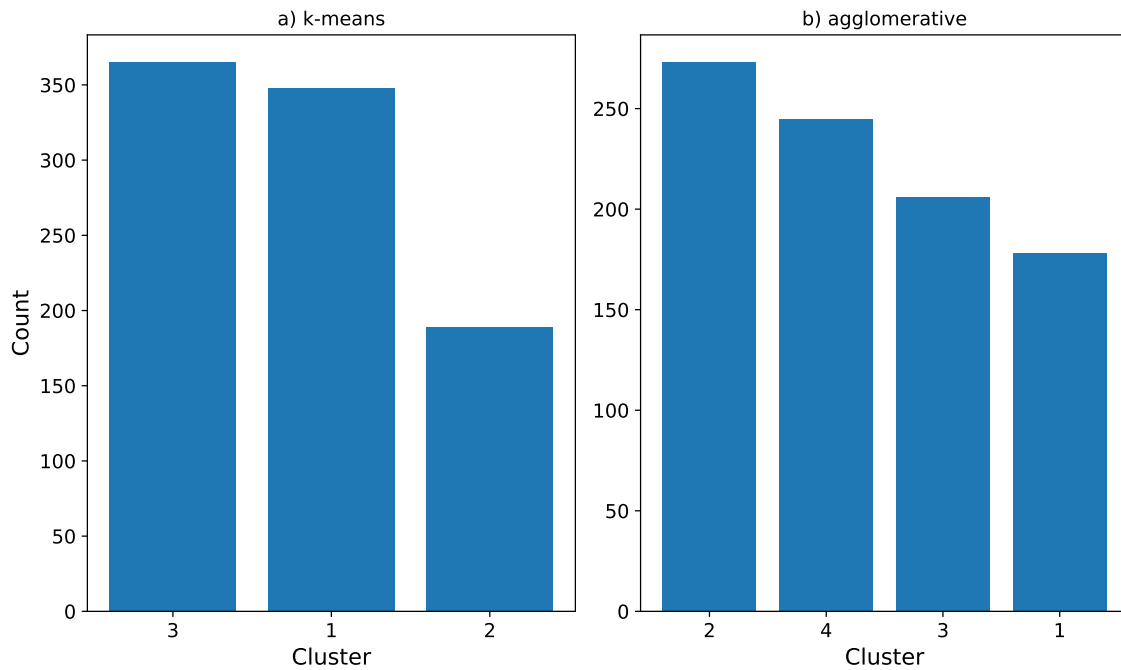


Figure 5.21: Bar plots for the number of vehicles in each cluster for a) k-means++ and b) agglomerative clustering.

To further understand the characteristics of these clusters, the box plots below (Figure 5.22) illustrate the distribution of mean transaction amounts across the clusters identified by the clustering methods. For the k-means++ approach, cluster 1, which contains approximately 340 vehicles, exhibits the lowest mean transaction amounts, with a median value of approximately R750. In contrast, clusters 2 and 3 are very similar in terms of average transaction amounts, with medians of nearly R1 000 and interquartile ranges of approximately R700 - R1 250.

The agglomerative clustering method yields similar patterns. Cluster 3 displays the highest median transaction amount, approximately R1 000, followed by clusters 1 and 2 with medians around R900 and R800, respectively. Cluster 4 in the agglomerative approach has the lowest median transaction amount at just below R750, consistent with the k-means++ results.

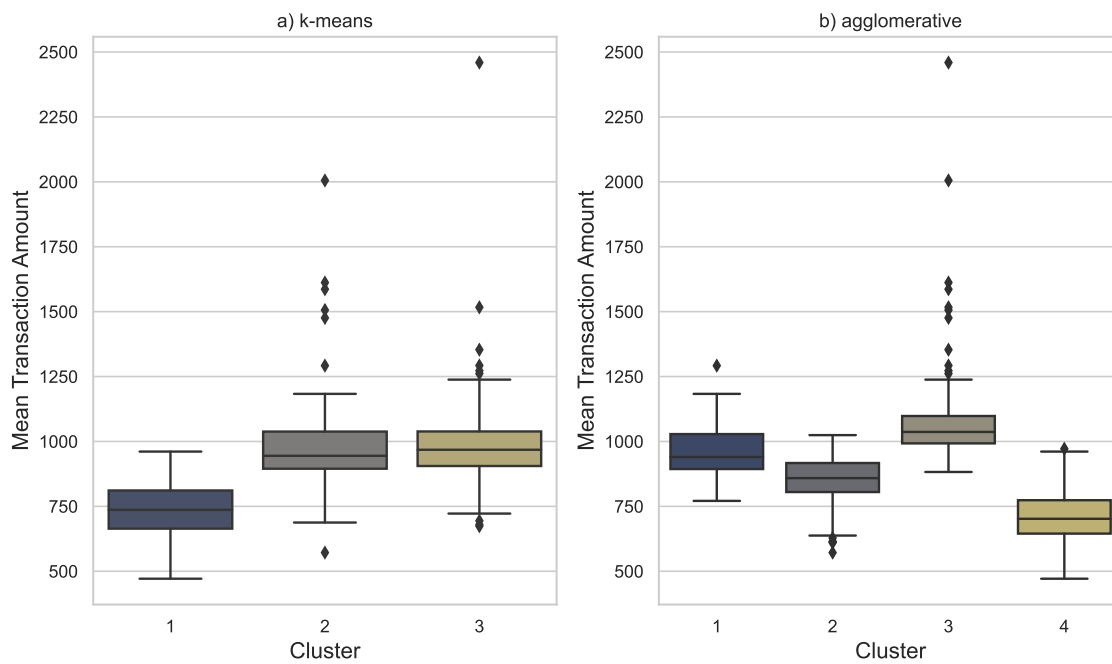


Figure 5.22: Box plots of the average transaction amount across clusters for a) k-means++ and b) agglomerative clustering.

Both methods appear to separate the vehicles with relatively low average transaction amounts from the rest of the fleet, suggesting that there are two major groupings in terms of average transaction amounts (Figure 5.23). These findings are also reflected in the plots of the average number of litres distributions, which are presented below.

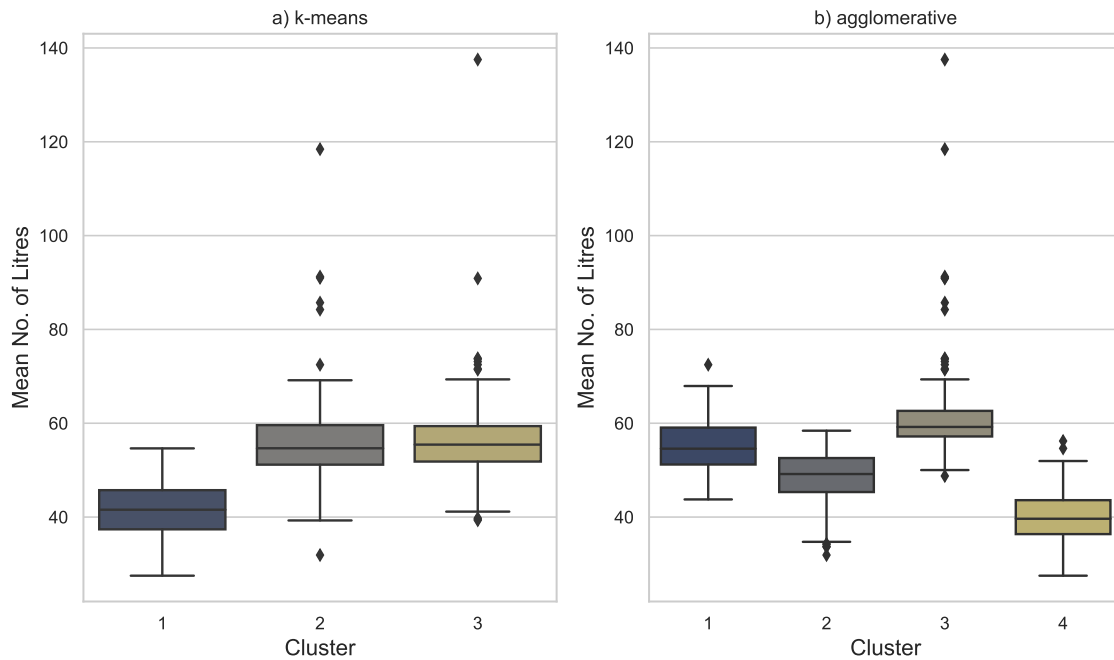


Figure 5.23: Box plots of the average number of litres across clusters for a) k-means++ and b) agglomerative clustering.

k-means++ Cluster Compositions

The stacked bar plots (Figure 5.24) reveal the composition of the k-means++ clusters across various attributes. For the model derivative, cluster 1 is dominated by VW Polo Vivo and Toyota Corolla Quest models, together accounting for 64% of the vehicles in this cluster. In contrast, clusters 2 and 3 exhibit a more diverse mix of model derivatives, with the "Others" category constituting the majority at 69% and 76%, respectively. Besides that, the Isuzu 250 double cab, Isuzu D-Max 250, and Nissan Hardbody models also have a notable presence in clusters 2 and 3, collectively exceeding 20% for both clusters.

Regarding the department composition, the Health and Social Development departments have the highest representation across all three clusters. However, cluster 2 stands out with a substantial proportion of vehicles from the Emergency Medical Services at 31%. The "Others" category is also substantial in each cluster, ranging from 16% to 31%, indicating the level of diversity of model derivatives within the clusters outside of those that are most popular.

Geographically, due to data limitations, the districts to which the majority of vehicles belonged were unknown, which is reflected in the compositions within each cluster. Among the known districts, OR Tambo has the highest representation in cluster 2 at 14%, while Chris Hani, Alfred Nzo, and Sarah Baartman have similar proportions across the clusters. Additionally, cluster 2 is the only cluster that contained a majority of known districts as opposed to unknown (68% known).

Lastly, the rate card category composition varies considerably among the clusters. Cluster 1 is predominantly composed of sedan vehicles (69%). Cluster 2 has a more balanced distribution, with LDV 4x2 D/Cab (26%) and LDV Clinic (16%) being the most prominent categories. Cluster 3 is characterized by a high proportion of LDV 4x2 1 ton vehicles (31%). In terms of diversity of categories, cluster 2 stood out with 42% of the cluster being comprised of categories outside of the top 5.

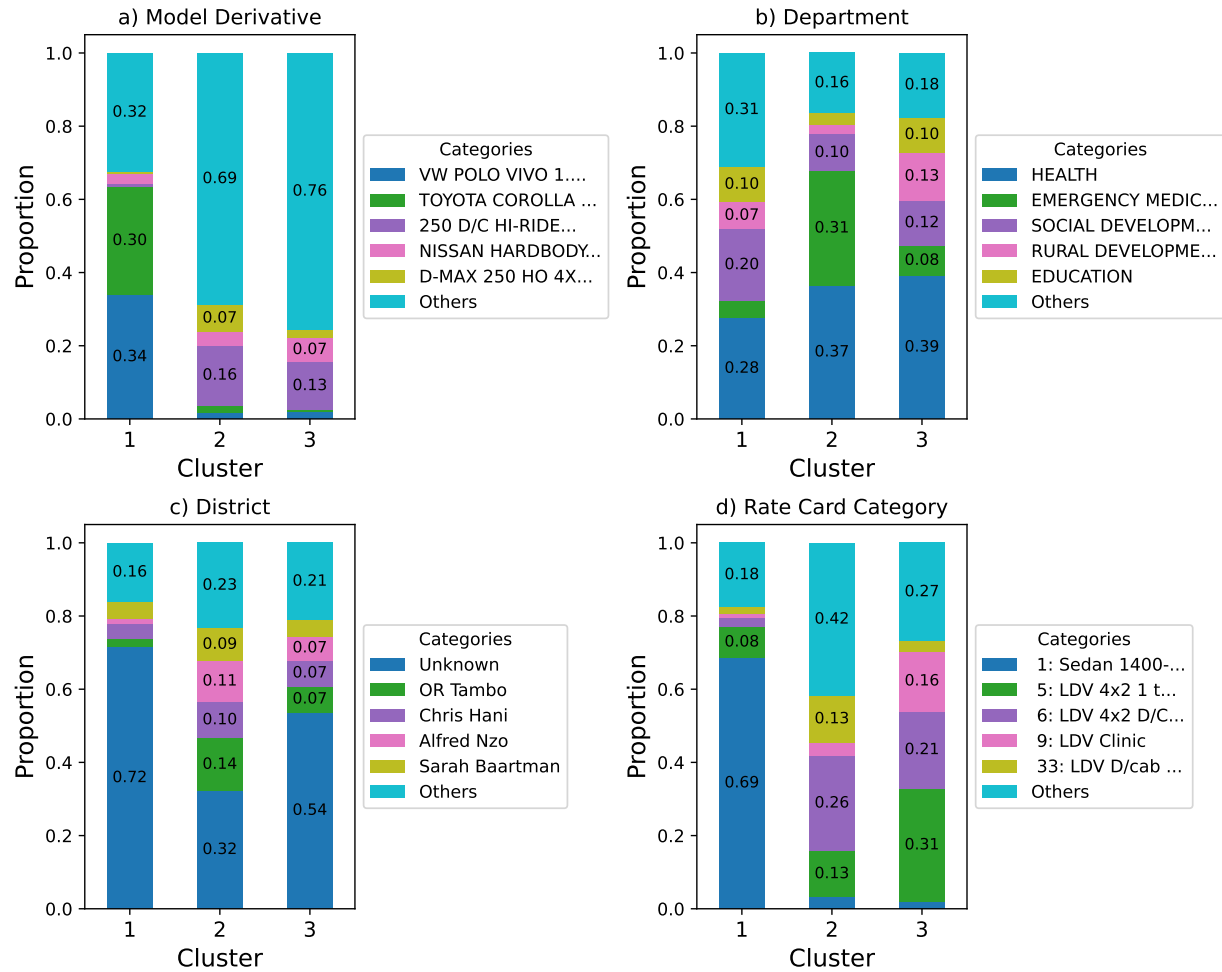


Figure 5.24: Stacked bar plots for the compositions of the k-means++ clusters across: a) model derivative, b) department, c) district and d) rate card category.

Agglomerative Clustering Compositions

For the agglomerative clustering approach, there appear to be notable differences in the composition of the clusters compared to the k-means++ method (Figure 5.25). Cluster 4 is characterized by a high concentration of VW Polo Vivo (31%) and Toyota Corolla Quest (40%) sedan models, while the other clusters exhibit a more diverse mix of model derivatives. The Isuzu 250 double cab and Nissan Hardbody models have a weighty presence in clusters 1 and 3, respectively (over 20%). The category consisting of all other derivatives dominates clusters 1, 2, and 3, ranging from 63% to 77%.

In terms of department composition, the Health department once again has the highest representation across all clusters, with proportions ranging from 29% to 42% (excluding EMS). The Social Development Rural Development departments also have a notable presence, particularly in clusters 1 and 2. Additionally, the diversity of departments is the greatest in cluster 4 at 31%.

Geographically, just as with k-means++, the majority of vehicles in clusters 2 and 4 are associated with an "Unknown" district (68% each). Among the known districts, OR Tambo has the highest representation in cluster 1 at 13%, while the other districts have relatively similar proportions across the clusters.

The rate card category composition varies considerably among the clusters. Cluster 4 is predominantly composed of sedan vehicles (79%), while clusters 1 and 3 have a high proportion of LDV 4x2 1 ton vehicles (25% and 31%, respectively). Cluster 2 has a more balanced distribution across various categories. Furthermore, the "Others" category is substantial in clusters 1, 2, and 3, ranging from 21% to 48%.

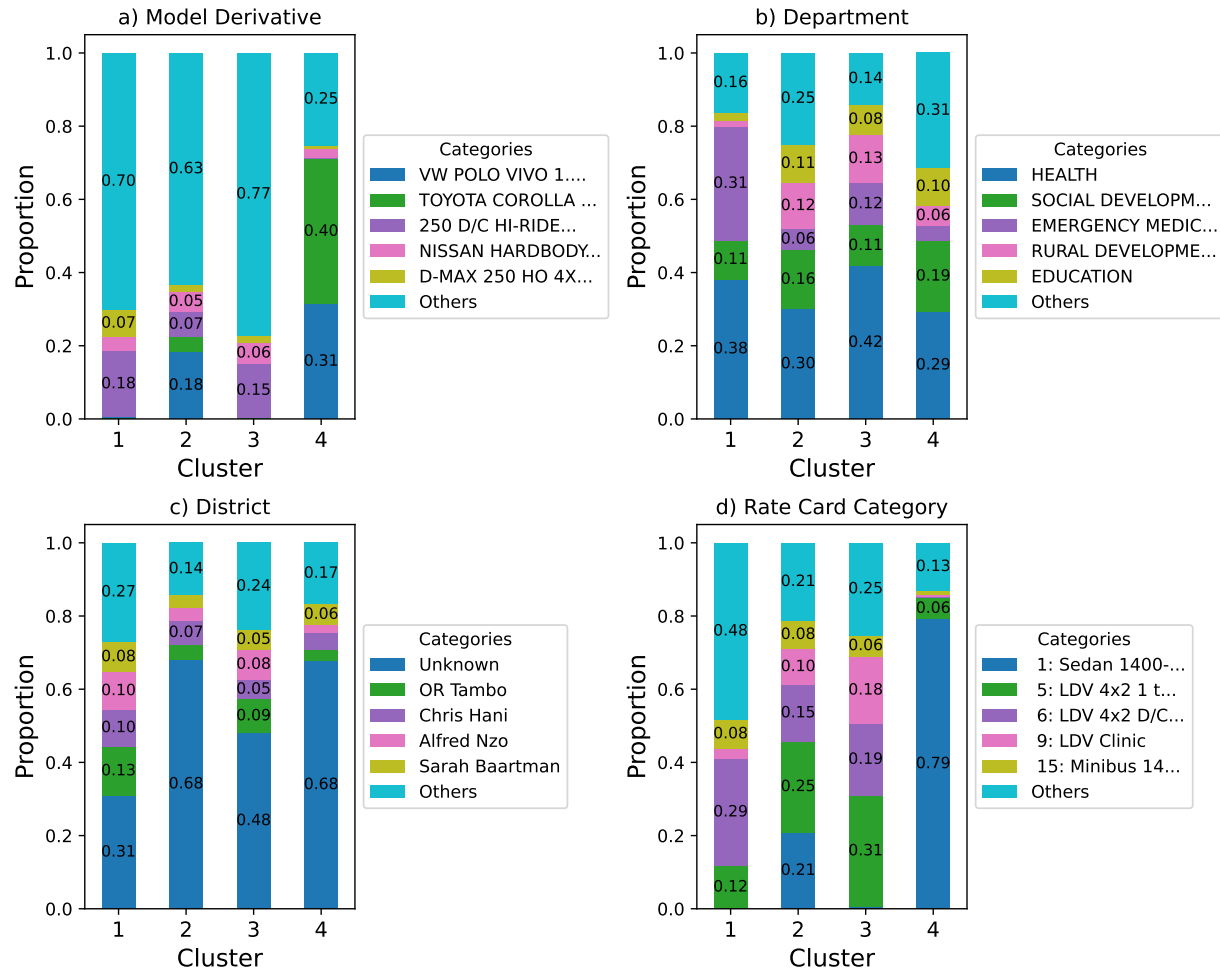


Figure 5.25: Stacked bar plots for the compositions of the agglomerative clusters across: a) model derivative, b) department, c) district and d) rate card category.

Comparing the agglomerative and k-means++ clustering results, several key observations can be made:

- The agglomerative approach yields a more distinct separation of sedan vehicles (cluster 4) from the other vehicle types, while the k-means++ clusters exhibit a more balanced distribution of vehicle types across clusters.
- Both methods show that the Health department (including EMS) has the highest representation across all clusters, followed by the Social Development department.
- Geographically, both methods are limited by the high proportion of vehicles with unknown district information. Among the known districts, OR Tambo has the highest representation in specific clusters (cluster 2 for k-means++ and cluster 1 for agglomerative).
- The diversity of model derivatives and rate card categories within clusters is more pronounced in the agglomerative clustering results, particularly in clusters 1, 2, and 3, where the "Others" category dominates.

-
- The k-means++ method produces clusters with a more even distribution of vehicles across departments and rate card categories, while the agglomerative approach results in clusters with higher concentrations of specific departments and rate card categories.

5.2.2 Clustering Transactions

Having gained a deeper understanding of the vehicle usage patterns, the next step is to cluster the transactions. Given the substantially larger number of observations in the transaction dataset compared to the vehicle dataset, the k-means++, BIRCH, and GMM algorithms were employed for this analysis. These three methods were chosen for their ability to efficiently process large datasets.

Tuning

For the k-means++ transaction clustering, the elbow method plot (Figure 5.26, left) depicts the within-cluster sum of squares (WCSS) against the number of clusters. As the number of clusters increases, the WCSS decreases, suggesting improved clustering. However, in this case, a distinct elbow point is not apparent. Consequently, 4 clusters were selected as a compromise.

The BIRCH clustering, which is comparable to agglomerative clustering but more efficient for larger datasets, employs the silhouette scores plot (Figure 5.26, right) to determine the optimal number of clusters. A higher silhouette score signifies better-defined and well-separated clusters. In this plot, the highest silhouette score is achieved with 2 clusters, followed by a slight decrease for 3 and 4 clusters, and a more pronounced drop for 5 clusters. Based on this information, 2 clusters are considered optimal.

For the Gaussian Mixture Model (GMM) clustering, four clusters were selected, using the k-means++ optimal number as a guide and for comparison purposes.

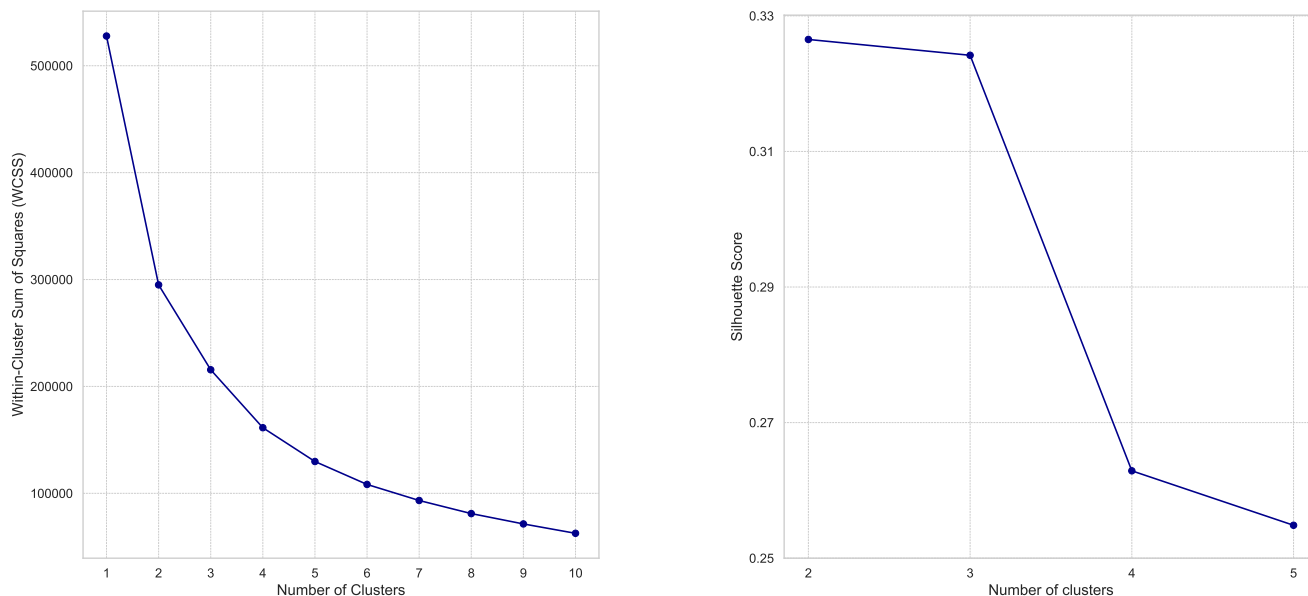


Figure 5.26: (Left) A line plot of the within cluster sum of squares (WCSS) over number of clusters to determine the number of clusters to use for the k-means++ algorithm. (Right) A line plot of the silhouette score over the number of clusters to determine the optimal number for the BIRCH algorithm.

Combined Results

The transaction counts for each cluster across the three clustering approaches are shown in Figure 5.27. The k-means++ and GMM methods exhibit similar cluster distributions, with cluster 2 containing the highest number of transactions (approximately 50 000 and 80 000, respectively), followed by clusters 1, 4, and 3. In contrast, the BIRCH method yields a more balanced distribution, with cluster 2 containing more than twice the number of transactions (approximately 120 000) compared to cluster 1 (about 50 000).

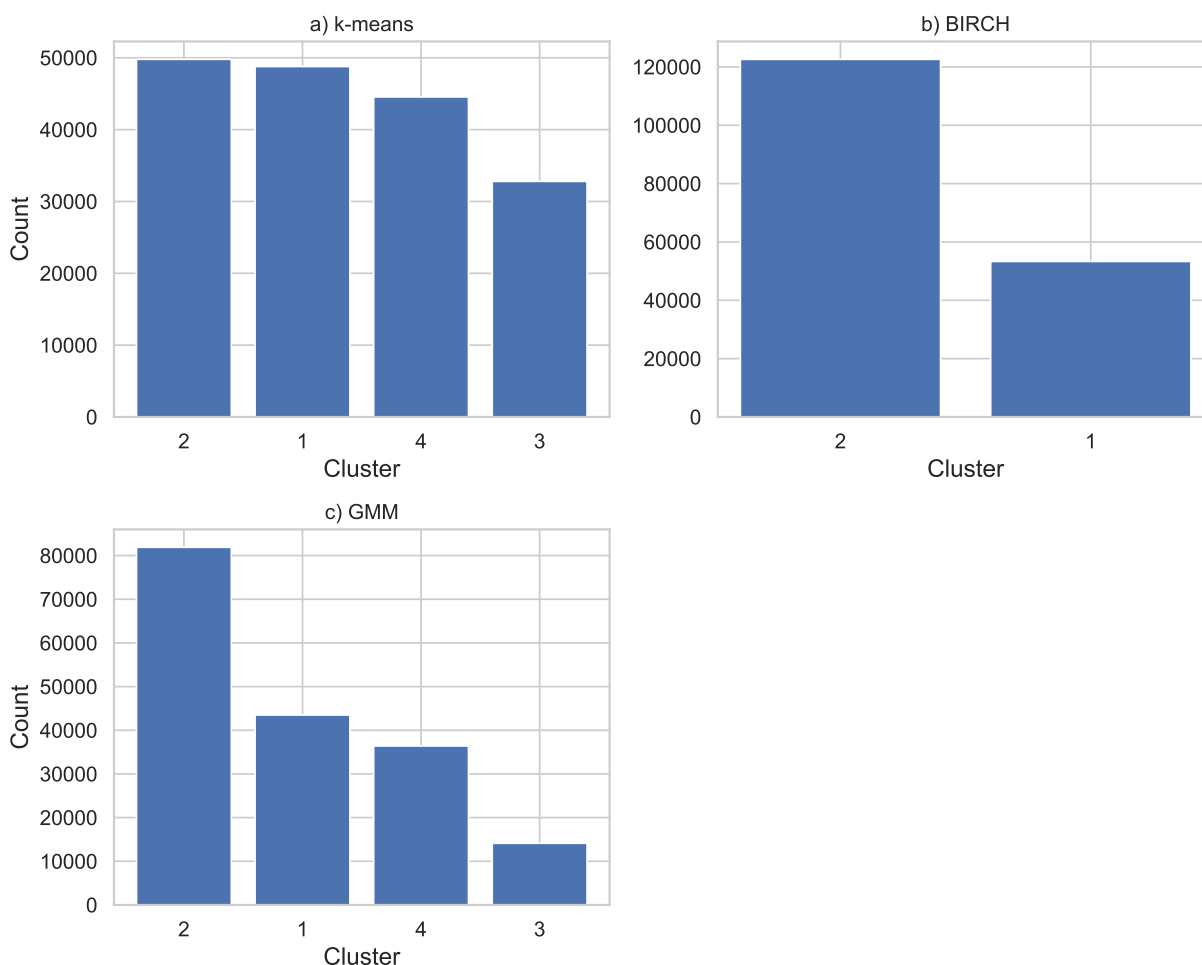


Figure 5.27: Bar plots of the transactions per cluster for a) k-means++, b) BIRCH and c) GMM.

The transaction amount distributions across clusters are visualized using box plots for each cluster and clustering method in Figure 5.28. For k-means++ and GMM, cluster 4 exhibits the highest median transaction amount (approximately R1 200 and R1 000, respectively), with a large number of outliers extending up to R3 000. Clusters 2, 3, and 4 have lower median transaction amounts, ranging from approximately R500 to just under R1 000, with fewer outliers. In contrast, the BIRCH method reveals a distinct pattern, with cluster 2 having a lower median transaction amount (approximately R800) and fewer outliers compared to cluster 1, which has a higher median (around R1 100) and a greater number

of outliers reaching up to R3 000. It appears that cluster 2 captures all the smaller transactions (with no outliers extending beyond R1 500), while cluster 1 captures all the larger transactions (including all outliers above R1 500). These patterns are nearly identically mirrored in the number of litres box plots presented in Figure 5.29.

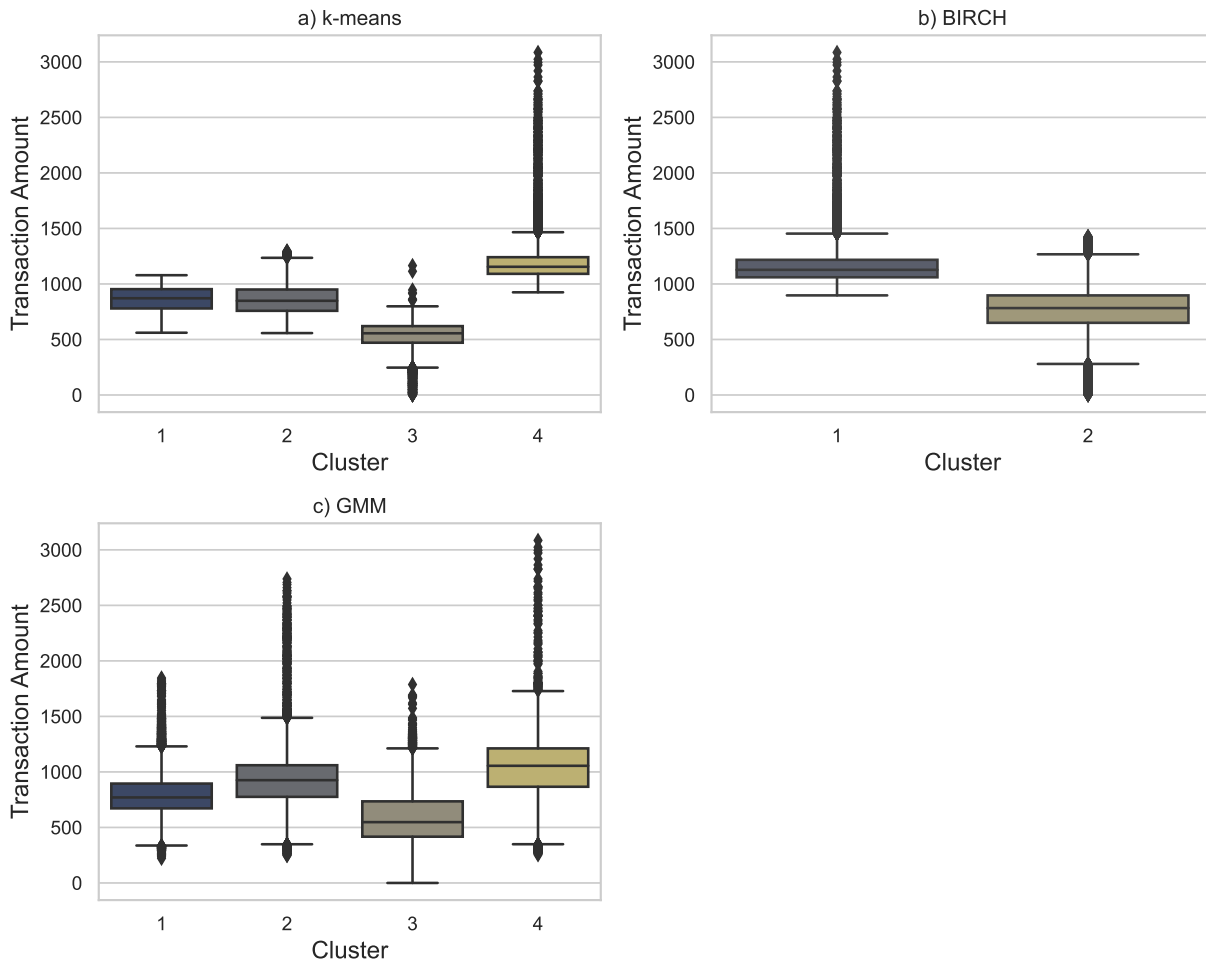


Figure 5.28: Box plots of the transaction amount per cluster for a) k-means++, b) BIRCH and c) GMM.

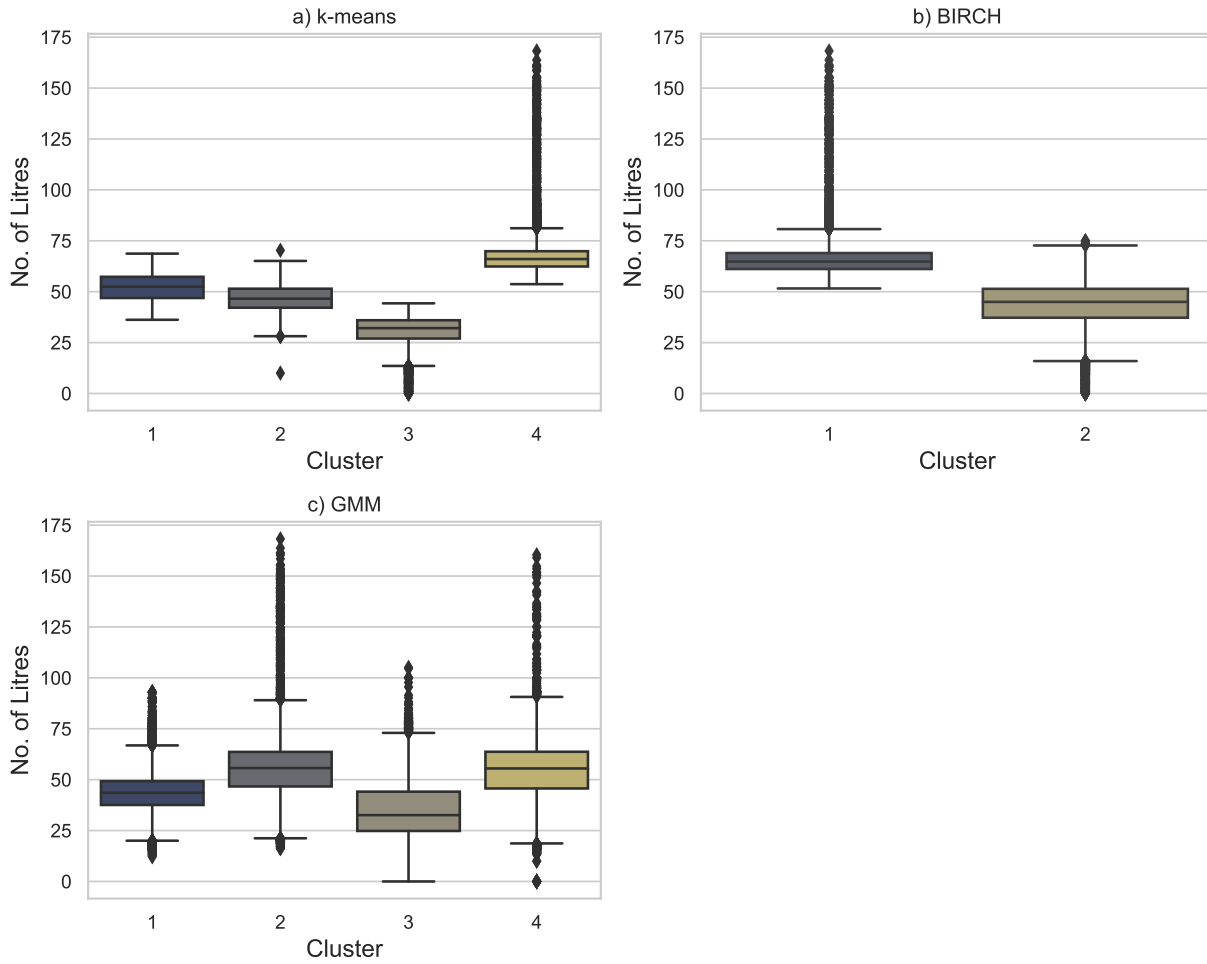


Figure 5.29: Box plots of the number of litres per cluster for a) k-means++, b) BIRCH and c) GMM.

k-means++ Cluster Compositions

Examining the compositions of each cluster across model derivative, department, district, and rate card category reveals interesting patterns (Figure 5.30). Focusing on the k-means++ method and excluding the less popular derivatives, cluster 2 primarily consists of vehicles from the VW Polo Vivo and Toyota Corolla model derivatives, with a relatively balanced distribution across departments and districts. This suggests that these popular vehicle models are widely used across various government functions and geographical areas. In contrast, cluster 1 has a sizeable proportion of VW Crafter 50 model derivatives and vehicles from Emergency Medical Services, with a higher representation from the Sarah Baartman district. This indicates a specialised cluster of vehicles catering to emergency medical needs, particularly in the Sarah Baartman region. Cluster 3 exhibits a more diverse composition across model derivatives, departments, and districts, implying a general-purpose fleet segment. Cluster 4 has a notable presence of Isuzu D-Max 250 and 250 double cab model derivatives, with a higher proportion of vehicles from the Health departments (including EMS). This cluster likely represents a mix of off-road capable vehicles used for health-related services.

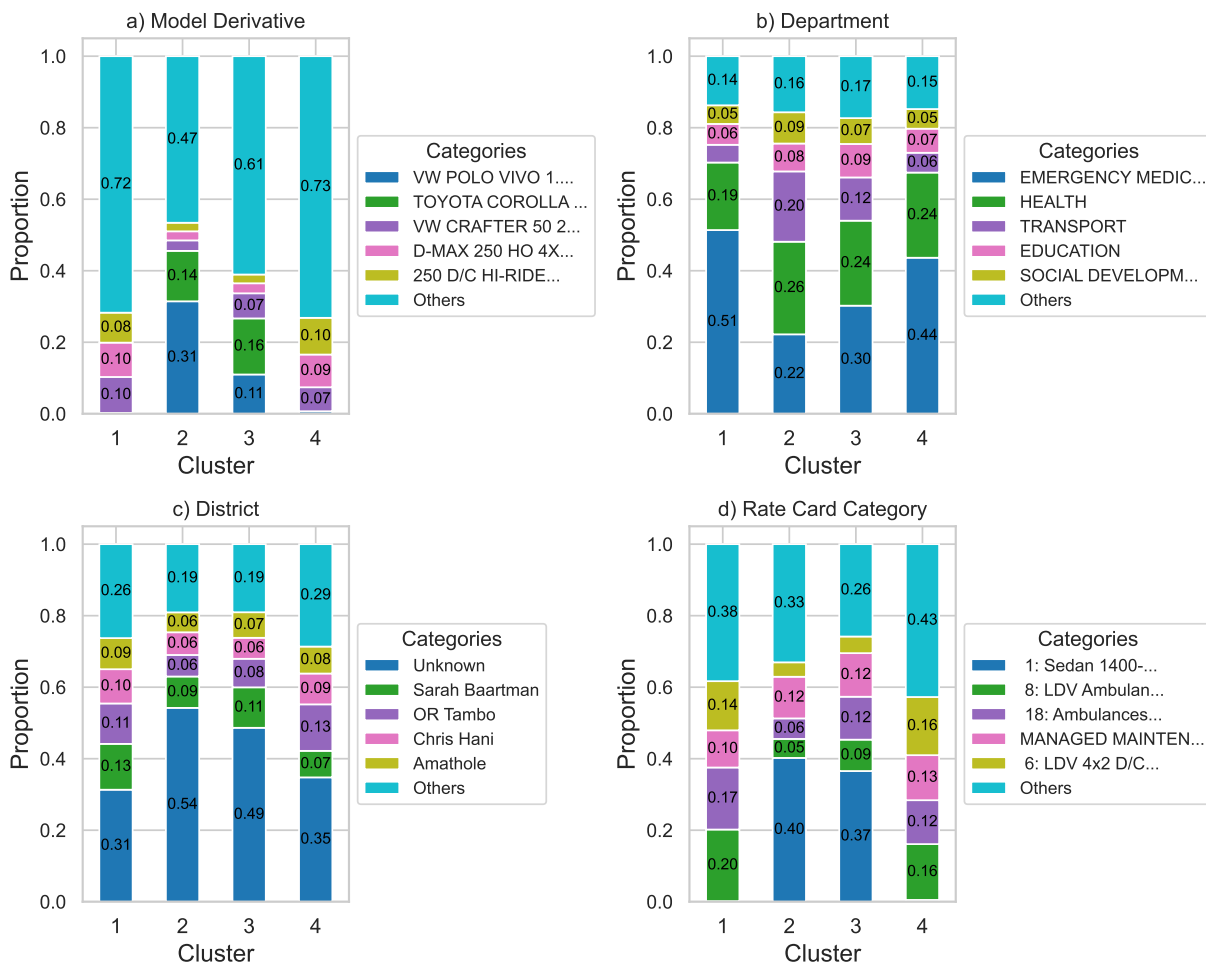


Figure 5.30: Stacked bar plots for the compositions of the k-means++ clusters across: a) model derivative, b) department, c) district and d) rate card category.

BIRCH Cluster Compositions

The BIRCH method reveals a different perspective on the cluster compositions (Figure 5.31). Cluster 1 is dominated by vehicles from the Emergency Medical Services department, with a noteworthy proportion of VW Crafter 50 and Isuzu 250 double cab model derivatives. This cluster appears to be specialized for emergency medical services, with a focus on larger vehicles suitable for transporting medical equipment and personnel. In contrast, cluster 2 has a more balanced composition across departments and districts, with a higher proportion of VW Polo Vivo and Toyota Corolla model derivatives. This cluster likely represents a general-purpose fleet segment, similar to cluster 3 in the k-means++ method, consisting of smaller, more fuel-efficient vehicles used for various administrative and operational tasks across different government departments and geographical areas.

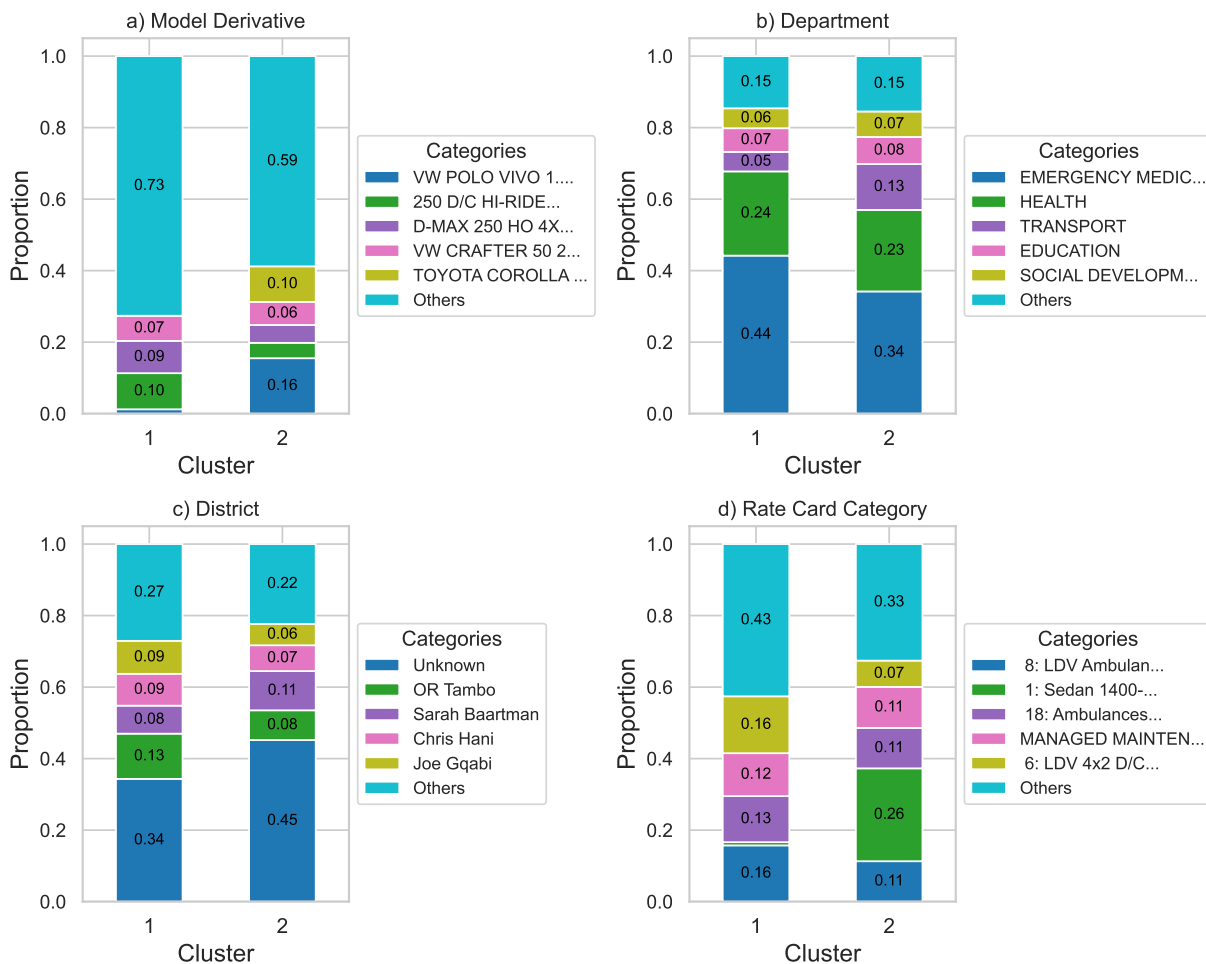


Figure 5.31: Stacked bar plots for the compositions of the BIRCH clusters across: a) model derivative, b) department, c) district and d) rate card category.

GMM Cluster Compositions

The GMM method yields cluster compositions that offer a nuanced view of the fleet segments (Figure 5.32). Cluster 1 is characterized by a high proportion of VW Polo Vivo and Toyota Corolla Quest model derivatives, with a relatively balanced distribution across departments and a higher representation from the transactions where the district was unknown. This cluster is similar to cluster 2 in the k-means++ method, likely representing a general-purpose fleet segment with smaller, fuel-efficient vehicles used across various departments.

Clusters 2 and 4 are primarily composed of vehicles from Emergency Medical Services (Department of Health), suggesting a specialization in emergency response and medical transportation. Cluster 2 has a notable presence of VW Crafter 50 model derivatives, indicating a focus on larger vehicles suitable for ambulance services. In contrast, cluster 4 exhibits a more diverse composition across model derivatives, possibly representing a mix of vehicles used for different aspects of emergency medical services.

Cluster 3 has a relatively balanced composition across departments and districts, with a slightly higher proportion of vehicles from the Health and Transport departments. This suggests a general-purpose fleet segment with a focus on health and transportation services, similar to cluster 2 from the BIRCH method and cluster 3 from the k-means++ method.

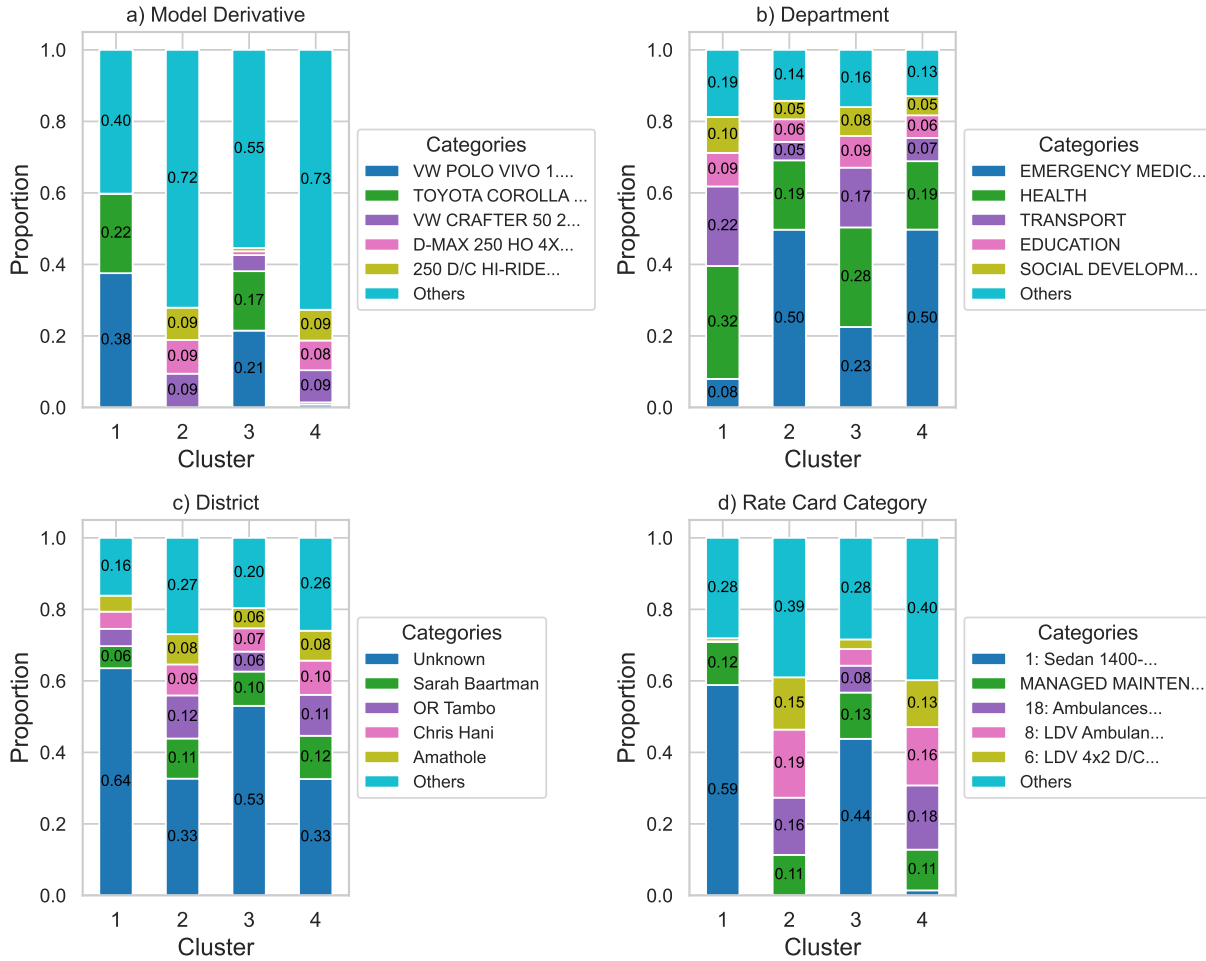


Figure 5.32: Stacked bar plots for the compositions of the GMM clusters across: a) model derivative, b) department, c) district and d) rate card category.

The following is a summary of the key findings and differences in the cluster compositions derived from the k-means++, BIRCH, and GMM clustering methods:

Key Shared Points:

- All methods identify clusters with a sizeable proportion of vehicles from the Emergency Medical Services (Department of Health), though the distribution of these vehicles across clusters varies. This suggests that Emergency Medical Services vehicles have distinct usage patterns that set them apart from other fleet segments, regardless of the clustering method used. The unique operational requirements and deployment patterns of emergency medical vehicles likely contribute to their differentiation from other fleet segments.
- Clusters with a high proportion of VW Polo Vivo and Toyota Corolla model derivatives are present in all methods, representing a general-purpose fleet segment. The presence of these popular model derivatives across clusters indicates their versatility and widespread use within

the fleet. These vehicles are likely chosen for their reliability, fuel efficiency, and suitability for various administrative and operational tasks across different departments and geographical areas.

- Clusters with a balanced composition across departments and districts are identified by all methods, suggesting the presence of versatile fleet segments. These clusters likely represent vehicles that are used for a variety of purposes and are not tied to specific departments or geographical areas. The balanced distribution of these vehicles across different contexts highlights their adaptability and utility in meeting diverse fleet requirements.

Key Differences:

- The k-means++ and GMM methods produce more granular clusters, capturing specific fleet segments based on model derivatives, departments, and districts, while the BIRCH method yields more general clusters. This difference in granularity can be attributed to the underlying algorithms and their sensitivity to the data structure. The k-means++ and GMM methods are better suited for identifying fine-grained patterns and distinctions within the fleet, while the BIRCH method focuses on broader, more homogeneous groupings.
- The BIRCH method clearly separates the Emergency Medical Services vehicles into a single cluster, while the k-means++ and GMM methods distribute these vehicles across multiple clusters. This suggests that the BIRCH method is more effective at isolating the unique characteristics of Emergency Medical Services vehicles, while the other methods capture more nuanced patterns within this department. The BIRCH method's ability to create a distinct cluster for these vehicles highlights the pronounced differences in their usage patterns compared to other fleet segments.
- The k-means++ and GMM methods identify clusters with a focus on specific model derivatives (e.g., VW Crafter 50, Isuzu D-Max 250), while the BIRCH method does not highlight such distinctions. This indicates that the k-means++ and GMM methods are more sensitive to the differences in usage patterns among specific model derivatives, allowing for a more detailed understanding of fleet composition.

Chapter 6

Results: Predictive Modelling

6.1 Indicators

Having completed the data exploration phase, the next step is to develop fuel fraud indicators. To this end, three indicators were developed: the abnormally large transactions indicator, the frequent transactions indicator, and the fuel price difference indicator. Each of these indicators is described in greater detail below.

6.1.1 Abnormally Large Transactions Indicator

The abnormally large transactions indicator is designed to identify transactions that substantially deviate from the norm within a specific context. This indicator considers the vehicle category, location (district), and time (month) to determine the average transaction amount. Based on the results from the exploratory data analysis and given the fact that time, location and vehicle category are considered, transactions exceeding this average by more than 50% should be flagged.

The average transaction amount can be calculated using the following equation:

$$\bar{x}_{c,d,m} = \frac{\sum_{i=1}^n x_i}{n} \quad (6.1)$$

where:

- $\bar{x}_{c,d,m}$ is the average transaction amount for a specific vehicle category c , district d , and month m
- x_i is the transaction amount for the i -th transaction within the specified context
- n is the total number of transactions within the specified context

To determine if a transaction is an anomaly, the following condition is checked:

$$x_i > 1.5 \times \bar{x}_{c,d,m} \quad (6.2)$$

If the condition is true, the transaction is flagged as an anomaly.

6.1.2 Frequent Transactions Indicator

This indicator identifies instances where a vehicle is refuelling more frequently than expected, with transaction amounts exceeding the average for its category. Transactions occurring within a short time frame (less than 2 days apart) and surpassing the average transaction amount are flagged. This indicator helps pinpoint cases where a vehicle may be consuming fuel inefficiently or where fraudulent behaviour might be occurring, such as side fueling. The calculation for the average amount is the same as the one for the large transactions indicator above. The only difference is that it takes the time between transactions into account.

6.1.3 Fuel Price Difference Flag

The fuel price difference flag addresses discrepancies between the actual and expected fuel prices. This flag compares the adjusted coastal diesel (adjusted government price) or coastal petrol price with the estimated price per litre calculated from the fuel transaction data.

$$\text{Estimated Price Per Litre} = \frac{\text{Transaction Amount}}{\text{Number of Litres}} \quad (6.3)$$

When the disparity between the fuel station's price and the government-mandated price surpasses a predefined threshold of R1, the transaction is flagged for potential irregularities. This threshold provides a degree of flexibility, acknowledging that slight discrepancies between the official and actual prices are common. The flagging mechanism serves as a valuable tool for detecting potential issues such as merchant overcharging, data entry errors, or fuel transaction fraud such as card inflation.

To determine the price adjustment for diesel, three different fuel stations in the Eastern Cape were called on 1 March 2024, and the prices were recorded. The stations were Sasol Cape Road, BP Cape Road, and Engen Cuyler. Table 6.1 presents the diesel prices from these stations along with the government-specified diesel price for the same date. The petrol prices all matched the government price, indicating that petrol pricing was not an issue.

Source	Diesel Price (Rand/litre)
Sasol Cape Road	22.75
BP Cape Road	23.34
Engen Cuyler	23.43
Government-specified price	20.64

Table 6.1: Diesel prices from various sources on 1 March 2024

The mean diesel price from the three stations (R23.17) was calculated, and the difference between this mean price and the government-specified price (R20.64) was found to be R2.53. This difference was added to the coastal diesel prices in the historical data, assuming a similar trend in the past. The adjusted coastal diesel prices were then used to create the fuel price difference flag.

Table 6.2 below presents the count of transactions flagged by each indicator.

Indicator	Flagged Transactions	Total Transactions
Abnormally large transactions	1653	175 933
Frequent transactions	26 673	175 933
Fuel price difference	9626	175 933

Table 6.2: Count of transactions flagged per each indicator

6.1.4 Combining the Flags

The bar plot in Figure 6.1 illustrates the distribution of transactions across the number of indicators (flags) triggered. The vast majority of transactions, approximately 140 000, have no flags associated with them, suggesting that most transactions do not exhibit anomalous characteristics based on the defined indicators. However, a substantial number of transactions, around 33 700, trigger one flag, indicating the presence of a single anomalous feature. The count of transactions with two flags is considerably lower at about 2 100, while only 13 transactions have triggered all three flags, pointing to a small subset of transactions that exhibit multiple anomalous characteristics simultaneously.

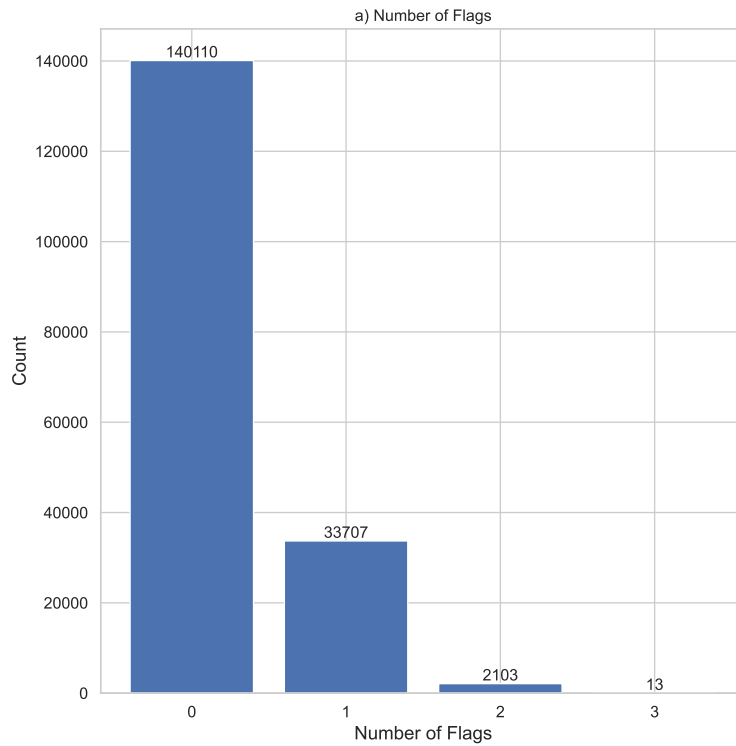


Figure 6.1: Barplot of the counts across number of indicators (flags).

To better investigate the patterns, Figure 6.2 presents box plots of transaction amounts and the number of litres across the number of triggered flags. The transaction amount box plots reveal a notable increase in the median and a decrease in the spread of transaction amounts as the number of flags increases. Transactions with no flags have a median amount of approximately R800, while those with three flags have a median close to R1 500, with a much wider spread. This suggests that transactions triggering multiple flags tend to be associated with higher monetary values, which aligns with the indicator definitions. Similarly, the box plots for the number of litres mirror those of the transaction amounts, displaying an increasing trend in the median and a decrease in the spread of litres pumped as the number of flags increases. Transactions with no flags have a median of around 50 litres, while those with three flags have a median exceeding 75 litres. This indicates that transactions with multiple anomalous characteristics often involve larger fuel volumes.

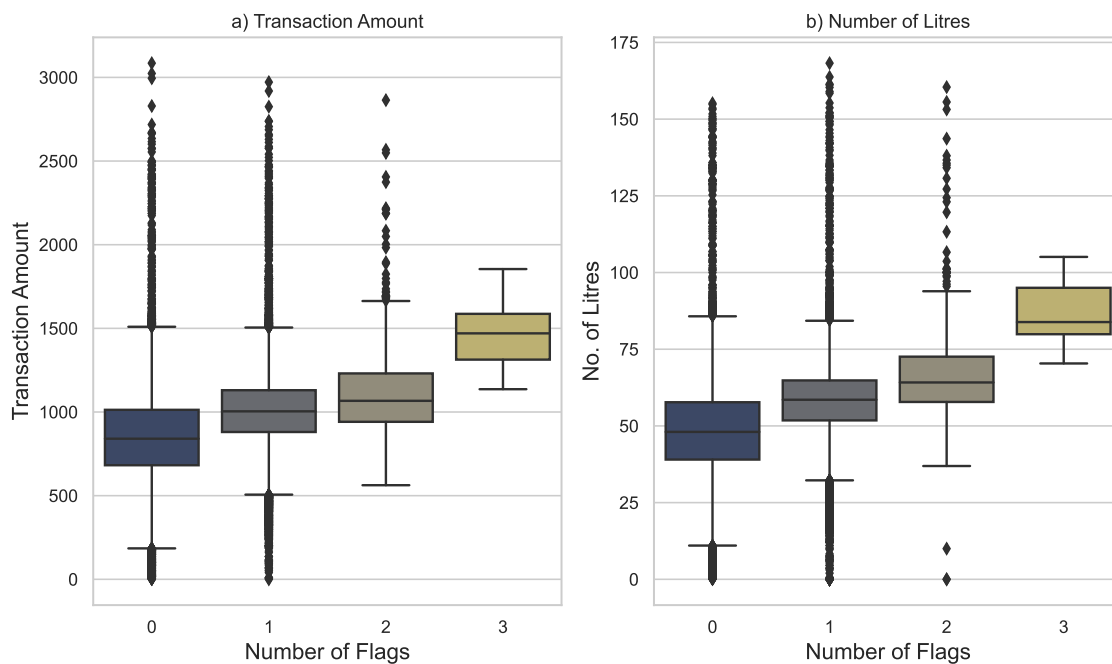


Figure 6.2: Box plots of the a) transaction amounts and b) number of litres across the number of indicators (flags).

The stacked bar plots in Figure 6.3 illustrate the composition of transactions with varying numbers of flags across different categories. For model derivatives, a large proportion of transactions belong to vehicles outside of the most popular ones, with proportions ranging from 87% for transactions with no flags to 31% for those with three flags. The D-Max 250 and VW Crafter 50 models exhibit higher proportions of flagged transactions compared to other specific models, with the VW Amarok 2.0 and Toyota Prado models showing notable proportions (15% and 23%, respectively) of transactions with three flags.

In terms of department composition, Health (including EMS) accounts for the largest proportions of transactions across all flag categories. The Transport and Cooperative Governance and Traditional Affairs departments have lower proportions of flagged transactions, while the Education department

has no transactions with three flags (of which there were only 13). Departments outside of the top 5 in terms of counts account for a substantial portion of transactions, particularly for those with no flags (21%), indicating that there are no standout departments in terms of no flags.

Regarding district composition, among the known districts, Sarah Baartman has the highest proportion of transactions with two flags (32%), while OR Tambo and Chris Hani have relatively consistent proportions across flag categories. The Amathole district has a lower representation in transactions with three flags (8%) compared to other flag categories. Transactions for vehicles where the district allocation was not available in the dataset seemed to dominate both the no flags and 3 flag groups (46% and 54%, respectively).

When examining rate card categories, the "Managed Maintenance" category has the highest proportion of transactions with three flags (46%), while categories 8 and 18 have similar proportions across all flag categories. Sedans (category 1) have the highest proportion of transactions with no flags among the top categories (21%) and a lower representation in flagged transactions, indicating that sedans seem to follow certain patterns regardless of district or time period. The category containing all other rate card categories accounts for a weighty portion of transactions across all flag categories, with proportions ranging from 37% for no flags to 15% for three flags, indicating a decrease in the diversity of categories as the number of flags increases (popular categories are more likely to be flagged than others, dominating the rest). It is important to once again note that the "Managed Maintenance" category may not accurately reflect the actual vehicle category, as the vehicle asset register came from a later time than the transaction data.

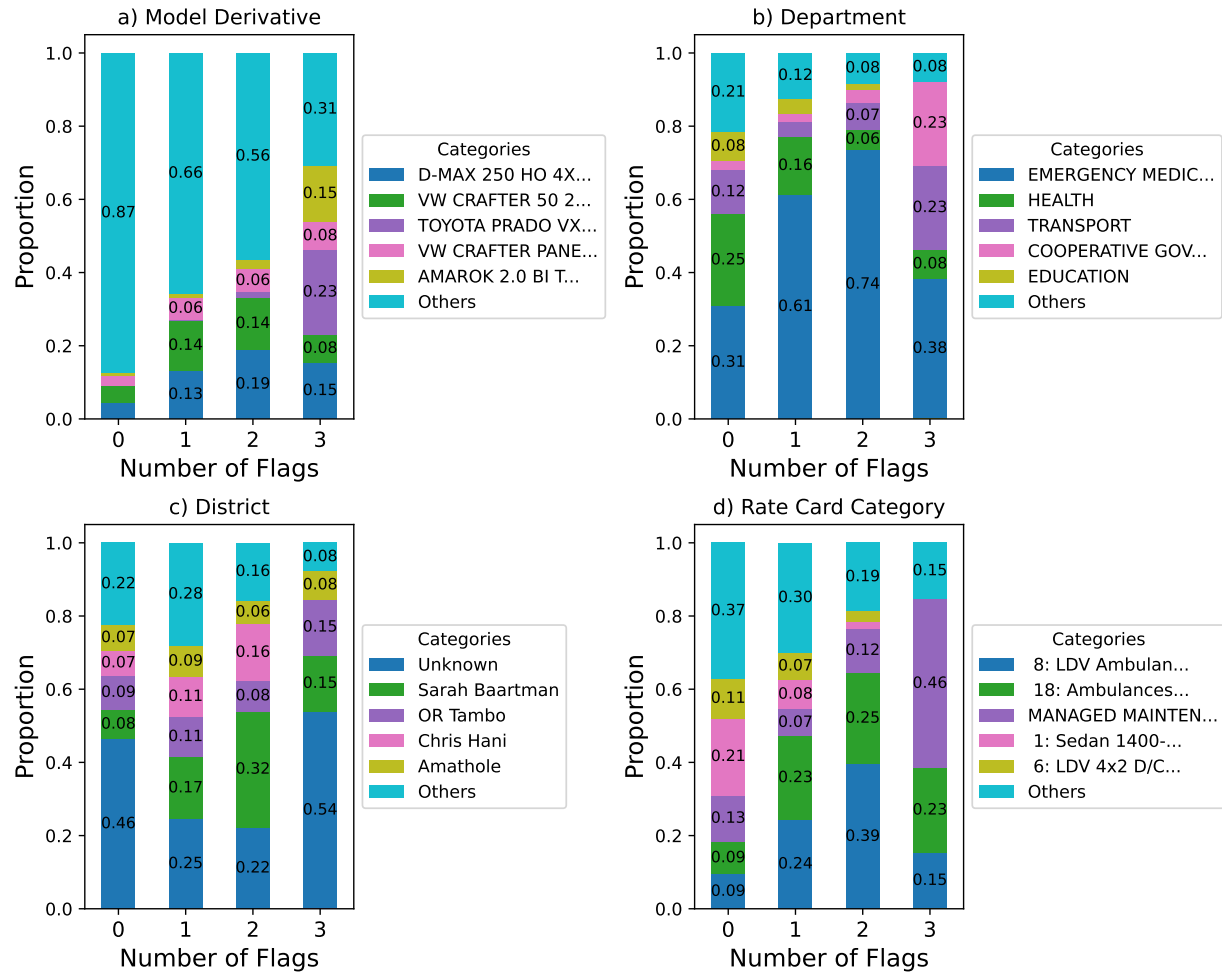


Figure 6.3: Stacked bar plots for the compositions of the transactions with different numbers of flags across: a) model derivative, b) department, c) district and d) rate card category.

6.2 Predictive Models

Having deterministically derived the indicators from the exploratory data analysis (EDA) and clustering results, predictive models can be developed to automatically classify transactions for new datasets. An 80-20 train-test split was utilised to train and subsequently evaluate the models. The performance of each model on the test set, which provides an indication of generalisability to unseen data, is discussed below. For details on model tuning parameters and the associated code, refer to the GitHub project accompanying this dissertation.

6.2.1 Overall Results

A comparison of the six different machine learning models used was conducted to evaluate their performance in classifying fuel transactions based on the number of fraud indicators. The models included in this analysis were a Linear Support Vector Machine (SVM), Naive Bayes, XGBoost, Random Forest, Multi-layer Perceptron, and Stochastic Gradient Descent (SGD) Classifier. The performance of each model was assessed using several key metrics, namely precision, recall, F1-score and accuracy, as shown in Table 6.3.

Model	Class (Flags)	Precision	Recall	F1-score	Support	Accuracy
Linear SVM	0	0.87	0.89	0.88	28022	0.80
	1	0.48	0.48	0.48	6742	
	2	0.26	0.02	0.03	423	
Naive Bayes	0	0.89	0.64	0.75	28022	0.58
	1	0.28	0.30	0.29	6742	
	2	0.04	0.69	0.07	423	
XGBoost	0	0.88	0.95	0.91	28022	0.85
	1	0.66	0.45	0.54	6742	
	2	0.65	0.24	0.35	423	
Random Forest	0	0.90	0.87	0.89	28022	0.81
	1	0.52	0.59	0.55	6742	
	2	0.43	0.41	0.42	423	
Multi-layer Perceptron	0	0.90	0.96	0.93	28022	0.87
	1	0.72	0.55	0.62	6742	
	2	0.64	0.49	0.56	423	
SGD Classifier	0	0.88	0.89	0.88	28022	0.79
	1	0.49	0.39	0.44	6742	
	2	0.17	0.53	0.25	423	

Table 6.3: Comparison of classification reports for different models on the test set (20% of the observations).

As evident from Table 6.3, the Multi-layer Perceptron model achieves the highest accuracy on the test set at 0.87, closely followed by XGBoost at 0.85 and Random Forest at 0.81. These models demonstrate strong overall performance in classifying fuel transactions given the following predictors: Transaction

Amount, Number of Litres, District, Vehicle Make, Fuel Type, Government-mandated Fuel Price, and Vehicle Category.

However, it is important to consider the performance of each model across individual classes, especially in the context of fraud detection, where the minority classes (1 and 2 fraud indicators) are of particular interest. Upon closer examination of the classification report, it becomes apparent that the Multi-layer Perceptron exhibits the best precision and F1-score for both minority classes, indicating its ability to accurately identify transactions with fraud indicators while minimising false positives. The XGBoost and Random Forest models also show promising results for these classes, with higher precision and F1-scores compared to the other models. These findings suggest that neural networks and tree-based models are better equipped to capture the intricate patterns and relationships within the data, leading to improved performance in identifying fraudulent transactions.

In contrast, the Naive Bayes model struggles with the minority classes, as evidenced by its low precision and F1-scores, despite having a high recall for transactions with 2 flags. This suggests that while the model is able to identify most transactions with 2 fraud indicators, it also generates a large number of false positives, which can be problematic when applying the model to new datasets.

The Linear SVM and SGD Classifier show moderate performance across all classes, with a notable weakness in identifying transactions with 2 fraud indicators, as indicated by their relatively low recall and F1-scores for this class.

In summary, the evaluation of the models on the test set reveals that the Multi-layer Perceptron, XGBoost, and Random Forest models exhibit the strongest performance in classifying fuel transactions based on the number of fraud indicators. These models not only achieve high overall accuracy but also demonstrate their ability to effectively handle minority classes. The following sections will delve deeper into the analysis of these three top-performing models, while the analyses of the other lower-performing models can be found in Appendix B.

6.2.2 XGBoost

The confusion matrix for the XGBoost model (Figure 6.4, left) and the corresponding ROC curves (Figure 6.4, right) provide a detailed breakdown of the model's performance in classifying fuel transactions based on the number of fraud indicators. The model demonstrates a strong ability to correctly identify transactions with no fraud indicators (class 0), accurately predicting 95.34% of these instances, with an AUC of 0.89. This aligns with the high precision (0.88), recall (0.95), and F1-score (0.91) for class 0, as shown in Table 6.3. However, the model misclassifies 4.64% of class 0 transactions as having 1 fraud indicator and 0.02% as having 2 indicators, suggesting room for improvement in reducing false positives.

For transactions with 1 fraud indicator (class 1), the XGBoost model correctly identifies 45.22% of instances, with an AUC of 0.87, indicating a moderate ability to detect these suspicious transactions. The model misclassifies 54.03% of class 1 transactions as having no fraud indicators and 0.74% as having 2 indicators, highlighting the challenge in accurately identifying transactions with a single fraud indicator. When it comes to transactions with 2 fraud indicators (class 2), the XGBoost model accurately predicts 24.11% of instances, with an impressive AUC of 0.98. However, the model misclassifies 12.53% of class 2 transactions as having no fraud indicators and 63.36% as having 1 indicator, suggesting that it struggles to distinguish between transactions with 1 and 2 fraud indicators. The low support (423) for class 2 compared to class 0 (28 022) and class 1 (6 742) may contribute to the model's difficulty in accurately identifying these transactions, as well as the discrepancy between the high class 2 AUC and the relatively low accuracy, which can be attributed to the limited representation of class 2 in the dataset hindering the model's ability to learn and generalise patterns specific to this class.

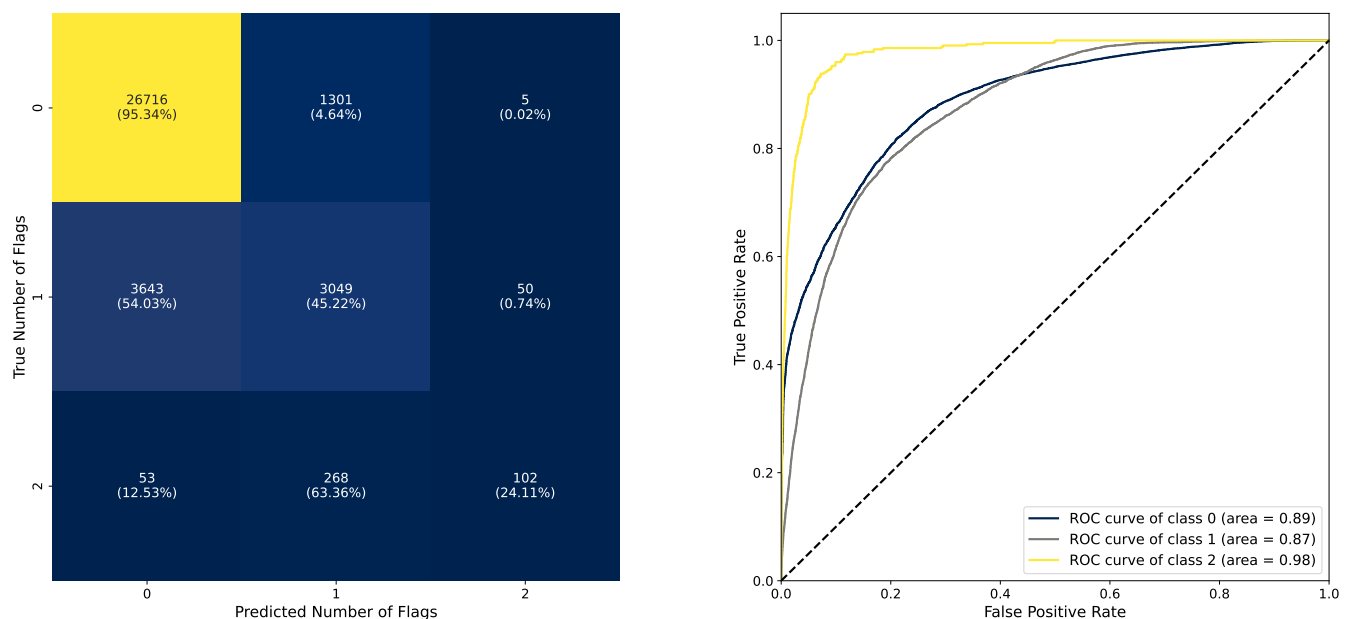


Figure 6.4: (Left) The confusion matrix with percentages of total observations for the XGBoost model on the test set. (Right) The ROC curves for the XGBoost model for each class on the test set using a one-vs-all approach, where the class of interest is treated as the positive class, and all other classes are combined to form the negative class.

The XGBoost model's performance in detecting fraudulent fuel transactions in the Eastern Cape government fleet is consistent with the findings of several studies that employed similar tree-based ensemble methods for credit card fraud detection. While the specific context and data may differ, the overall effectiveness of these models in identifying fraudulent activities is comparable. Zareapoor and Shamsolmoali (2005) used a bagging ensemble classifier for credit card fraud detection, demonstrating its ability to handle data imbalance while achieving high detection rates. The strong performance of the XGBoost model in accurately identifying transactions with no fraud indicators (class 0) and its excellent ability to distinguish between transactions with 2 fraud indicators (class 2) and those with fewer indicators align with the findings of Zareapoor and Shamsolmoali (2005), highlighting the effectiveness of ensemble methods in fraud detection tasks. Similarly, the moderate performance of the XGBoost model in identifying transactions with a single fraud indicator (class 1) is consistent with the challenges faced by ensemble models in accurately classifying instances from imbalanced datasets, as observed in Zareapoor and Shamsolmoali (2005).

In summary, the XGBoost model demonstrates strong performance in identifying transactions with no fraud indicators and excellent performance in distinguishing between transactions with 2 fraud indicators and those with fewer indicators. However, the model's performance in accurately identifying transactions with a single fraud indicator is moderate, as evidenced by the lower precision, recall, and F1-score for class 1. Despite these challenges, the XGBoost model's overall performance, as indicated by the accuracy of 0.85 on the test set, suggests that it is a valuable tool for detecting fraudulent transactions in the Eastern Cape government fleet fuel management system.

6.2.3 Multi-layer Perceptron (MLP)

The confusion matrix for the MLP model (Figure 6.5, left) and the corresponding ROC curves (Figure 6.5, right) demonstrate the model's strong performance in classifying fuel transactions based on the number of fraud indicators, with notable improvements compared to the XGBoost model. The MLP correctly identifies 95.57% of transactions with no fraud indicators (class 0), with an AUC of 0.92, indicating an excellent ability to distinguish between legitimate transactions and those with at least one fraud indicator. This high accuracy for class 0 is supported by the impressive precision (0.90), recall (0.96), and F1-score (0.93) shown in Table 6.3.

For transactions with 1 fraud indicator (class 1), the MLP accurately predicts 54.54% of instances, with an AUC of 0.90, demonstrating the model's strong ability to differentiate between transactions with 1 fraud indicator and those with either no indicators or 2 indicators. Although there is still room for improvement, the model's performance in identifying transactions with a single fraud indicator is commendable, as it correctly flags more than half of these cases. The MLP exhibits better performance in identifying transactions with 2 fraud indicators (class 2) than the XGBoost model, correctly classifying 49.41% of these instances, with an exceptional AUC of 0.99. Despite the relatively low support (423) for class 2, the MLP demonstrates a stronger ability to identify the most anomalous transactions while minimising misclassifications.

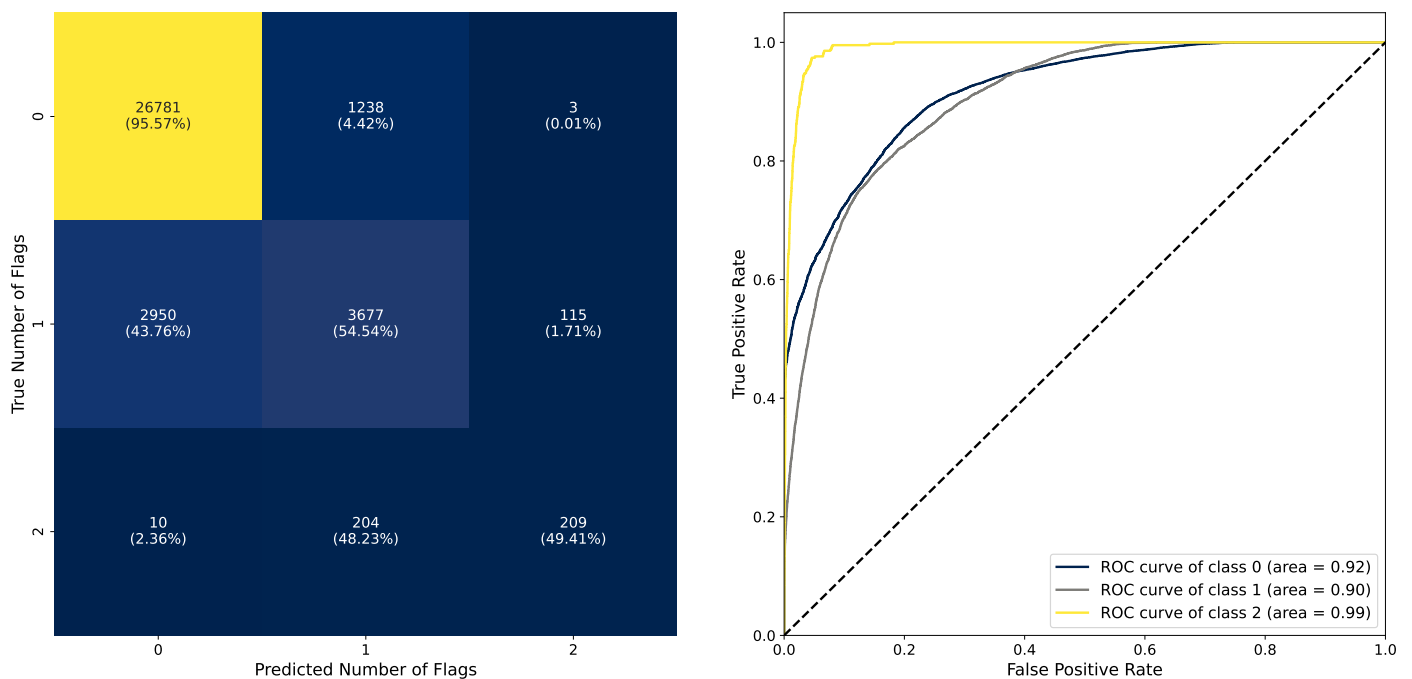


Figure 6.5: (Left) The confusion matrix with percentages of total observations for the Multi-layer Perceptron model on the test set. (Right) The ROC curves for the Multi-layer Perceptron model for each class on the test set using a one-vs-all approach, where the class of interest is treated as the positive class, and all other classes are combined to form the negative class.

The MLP model’s outstanding performance in detecting fraudulent fuel transactions in the Eastern Cape government fleet is consistent with the findings of several studies that employed similar neural network-based methods for credit card fraud detection. The effectiveness of the MLP model in accurately identifying transactions with varying numbers of fraud indicators aligns with the results reported in the literature, despite differences in context and data. Khan, Akhtar, and Qureshi (2014) used a real-time credit card fraud detection system based on an artificial neural network tuned by a simulated annealing algorithm. The high precision, recall, and F1-scores achieved by the MLP model across all classes, particularly for transactions with no fraud indicators (class 0) and those with 2 fraud indicators (class 2), are consistent with the findings of Khan, Akhtar, and Qureshi (2014), highlighting the effectiveness of neural network-based approaches in fraud detection tasks. Similarly, Ganesh and Sena (2012) proposed a novel artificial neural network and logistic approach for detecting credit card fraud, demonstrating the ability of neural networks to accurately identify fraudulent transactions. The MLP model’s superior performance compared to the XGBoost model in identifying transactions with a single fraud indicator (class 1) is in line with the findings of Ganesh and Sena (2012), which showcase the strengths of neural networks in handling complex patterns and imbalanced datasets in fraud detection.

In summary, the MLP model demonstrates excellent performance in classifying fuel transactions based on the number of fraud indicators, with notable improvements over the XGBoost model. The high precision, recall, and F1-scores for all classes, along with the impressive AUC values, indicate the model’s ability to accurately identify transactions with no fraud indicators, 1 fraud indicator, and 2 fraud indicators while minimising false positives and false negatives. Despite the challenges posed by the imbalanced dataset, the MLP model’s overall performance, as indicated by the accuracy of 0.87 on the test set, makes it the most effective model among those evaluated for detecting fraudulent transactions in the Eastern Cape government fleet fuel management system.

6.2.4 Random Forest

The confusion matrix for the Random Forest model (Figure 6.6, left) and the corresponding ROC curves (Figure 6.6, right) provide insights into its performance in classifying fuel transactions based on the number of fraud indicators. The model correctly identifies 87.22% of transactions with no fraud indicators (class 0), with an AUC of 0.86, demonstrating a good ability to distinguish between legitimate transactions and those with at least one fraud indicator. This is supported by the high precision (0.90), recall (0.87), and F1-score (0.89) for class 0, as shown in Table 6.3.

For transactions with 1 fraud indicator (class 1), the Random Forest model accurately predicts 59.11% of instances, with an AUC of 0.84, showing a satisfactory performance in distinguishing between transactions with 1 fraud indicator and those with either no indicators or 2 indicators. While the model's performance in this class is better than that of the XGBoost model, it still falls short of the Multi-layer Perceptron's accuracy. When it comes to transactions with 2 fraud indicators (class 2), the Random Forest model correctly identifies 40.90% of instances, with a high AUC of 0.94, demonstrating a reasonable ability to detect the most questionable transactions. Although the model's performance in this class is not as impressive as that of the Multi-layer Perceptron, it still outperforms the XGBoost model.

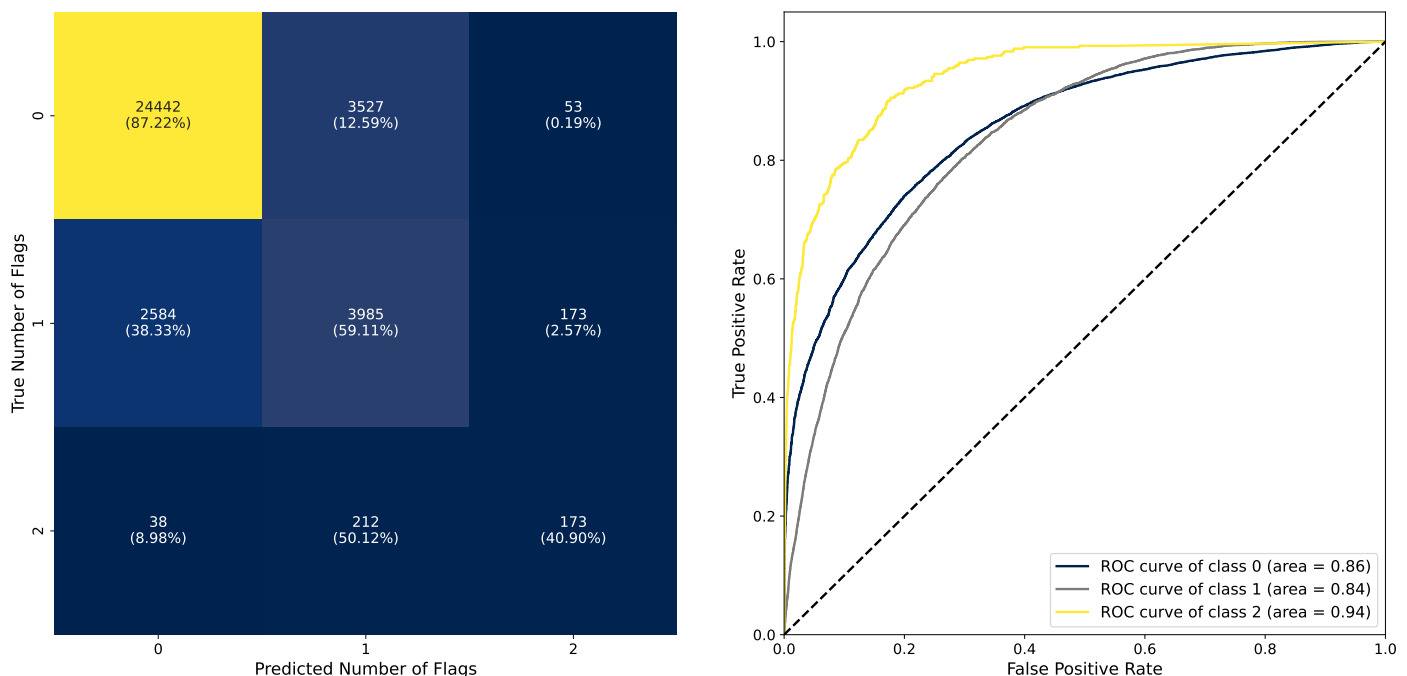


Figure 6.6: (Left) The confusion matrix with percentages of total observations for the Random Forest model on the test set. (Right) The ROC curves for the Random Forest model for each class on the test set using a one-vs-all approach, where the class of interest is treated as the positive class, and all other classes are combined to form the negative class.

The Random Forest model's performance is consistent with the findings of a study by Sahin and Duman (2011) that compared the performance of decision tree methods and Support Vector Machines (SVMs) on a real-world credit card fraud detection dataset. Although the study primarily focused on SVMs, it also evaluated the performance of decision tree models, such as C&RT, which share similarities with the Random Forest model in terms of their tree-based nature. The Random Forest model's strong performance in accurately identifying legitimate transactions (class 0) and its reasonable ability to detect transactions with 2 fraud indicators (class 2) align with the findings of Sahin and Duman (2011), where the decision tree models, especially C&RT, demonstrated high accuracy in identifying fraudulent transactions. Furthermore, the Random Forest model's moderate performance in identifying transactions with a single fraud indicator (class 1) is consistent with the challenges faced by the decision tree models in Sahin and Duman (2011) when dealing with imbalanced datasets and limited features.

In conclusion, the Random Forest model demonstrates good overall performance, with a particularly strong ability to identify legitimate transactions. Although its accuracy in identifying transactions with 1 and 2 fraud indicators is not as impressive as that of the Multi-layer Perceptron, it outperforms the XGBoost model in these classes. The Random Forest model's ensemble approach, which combines multiple decision trees, allows it to capture intricate patterns and relationships in the data, leading to its satisfactory results.

Chapter 7

Discussion and Conclusion

The research presented in the preceding sections highlights several key findings and implications concerning fuel usage patterns and fuel fraud detection within the Eastern Cape government fleet. The univariate exploratory data analysis provided an extensive overview of the fleet composition, transaction patterns, and fuel consumption trends within the Eastern Cape government fleet. The fleet predominantly consisted of Toyota, Volkswagen, and Isuzu vehicles, with the VW Polo and Toyota Corolla Quest being the most common model derivatives. This preference for reliable and widely available vehicles is understandable given the challenging infrastructure and the need for simplified maintenance in the region. However, the presence of luxury vehicle makes such as Mercedes-Benz and Audi in a government fleet raises concerns about their necessity and cost-effectiveness, as these vehicles typically have higher fuel consumption and maintenance costs compared to more practical alternatives.

The analysis of transaction volumes revealed that sedans such as the VW Polo and Toyota Corolla Quest were the most frequently refuelled, possibly due to their higher usage in urban areas. Light-duty vehicles (LDVs) and pick-up trucks, such as the Isuzu D-Max and Nissan Hardbody, were also well-represented in the fleet, serving a wide range of use cases across different departments. The Department of Health, which includes Emergency Medical Services (EMS), had the largest fleet, followed by Social Development. Interestingly, EMS vehicles accounted for the highest number of refuelling transactions, which could be attributed to the increased demand for emergency services during the COVID-19 pandemic.

The bivariate analysis yielded further insights into the relationships between vehicle attributes and fuel consumption patterns. Luxury vehicle makes, such as Mercedes-Benz and Jeep, exhibited the highest average transaction amounts and number of litres per refill, reinforcing the concern about their cost-effectiveness in a government fleet. The analysis of refuelling intervals revealed that a substantial number of transactions occurred within a two-day period, with vehicles refuelling more frequently than anticipated, potentially indicating fuel theft or misuse. The examination of fuel efficiency (kilometres per litre) across different vehicle categories demonstrated that sedans generally exhibited higher efficiency compared to light delivery vehicles (LDVs) and pick-up trucks, emphasising the importance of considering vehicle types when optimising fuel consumption.

The application of clustering techniques, including k-means, agglomerative, Gaussian Mixture Model (GMM), and BIRCH, yielded valuable insights into the underlying patterns of vehicle usage and fuel consumption. The clustering of vehicles revealed distinct segments based on their attributes and usage patterns. For instance, the k-means++ clustering identified a cluster dominated by Volkswagen Polo Vivo and Toyota Corolla Quest sedans, which were extensively utilised across various government departments and geographical areas. Conversely, another cluster comprised a substantial proportion of vehicles from the Department of Health’s Emergency Medical Services (EMS), indicating a specialised fleet segment catering to emergency medical needs.

The clustering of transactions also uncovered notable patterns. Departments such as Health, including EMS, and specific vehicle models such as the Isuzu D-Max and VW Crafter, were often associated with transactions having higher monetary values and fuel volumes. This finding suggests that certain departments and vehicle types may be more prone to abnormal fuel consumption or potential fraudulent activities. The BIRCH clustering, in particular, effectively separated smaller transactions from larger ones, highlighting the importance of considering transaction size when detecting anomalies.

In terms of fuel fraud indicators, three key indicators were created to flag potentially fraudulent activities: abnormally large transactions, frequent transactions, and fuel price differences. The abnormally large transactions indicator identified transactions that markedly exceeded the average transaction amount for a given vehicle category within a specific district and time period. By considering the contextual factors, this indicator helps detect transactions that deviate substantially from the norm, which may warrant further investigation. The frequent transactions indicator flagged transactions that occurred within a short time frame (less than two days apart) and had a transaction amount greater than the average for the corresponding vehicle category. This indicator aims to identify instances where vehicles are refuelling more frequently than expected for large amounts of fuel, potentially suggesting fraudulent behaviour. Lastly, the fuel price difference indicator flagged transactions where the discrepancy between the actual adjusted fuel price and the estimated price per litre from the transaction data exceeded a predefined threshold of R1. This indicator helps detect anomalies in the fuel procurement process or potential fraud by merchants.

The application of these indicators to the dataset revealed a substantial number of transactions triggering one or more flags. Transactions with multiple flags were associated with higher monetary values and fuel volumes, reinforcing the suspicion of fraudulent activities. The composition analysis of flagged transactions provided valuable insights into the vehicle types, departments, and geographical areas that were more prone to anomalies. For instance, vehicles from the Department of Health’s Emergency Medical Services (EMS) and specific models such as the Isuzu D-Max and Volkswagen Crafter had a higher proportion of flagged transactions, indicating the need for closer monitoring and investigation.

To automate the process of fraud detection, predictive models were developed using various algorithms, including XGBoost, Multi-layer Perceptron, and Random Forest. The feature selection process was guided by insights gained from the EDA, clustering and indicator creation results, which helped identify the most informative variables. From the large pool of available features, a key subset was selected to train these models, including transaction amount, number of litres, district, vehicle make, fuel type, and vehicle category. By focusing on these critical features, the models were designed to classify

transactions based on the number of fraud indicators triggered, enabling the identification of potentially fraudulent activities with improved accuracy and efficiency.

The Multi-layer Perceptron emerged as the top-performing model, demonstrating excellent accuracy across all classes, particularly in identifying legitimate transactions and those with multiple fraud indicators. The model's ability to capture complex, non-linear relationships in the data through its hidden layers contributed to its superior performance. The XGBoost and Random Forest models also exhibited strong performance, with the Random Forest model showing better accuracy in classifying transactions with a single fraud indicator compared to XGBoost. It is worth noting that the Random Forest model is able to scale better without worrying about overfitting due to its ensemble nature and the use of bootstrap aggregating (bagging) technique. On the other hand, XGBoost prevents overfitting through a different approach, which includes regularisation terms in its objective function and the use of a learning rate to control the contribution of each tree.

The evaluation of these predictive models using confusion matrices and ROC curves provided a comprehensive assessment of their performance. The Multi-layer Perceptron achieved high precision, recall, and F1-scores for all classes, indicating its effectiveness in minimising false positives while maintaining a high true positive rate. The XGBoost and Random Forest models also demonstrated good performance, with the Random Forest model showing better results for transactions with one fraud indicator.

The findings and methodologies presented in this dissertation contribute substantially to the understanding of fuel usage patterns, anomaly detection, and fraud prevention in government fleets. However, it is essential to acknowledge the limitations of the study and identify potential avenues for future research to build upon the current work and address remaining challenges.

One of the primary limitations of this research is the scope of the data, which was limited to a specific time period (April 2021 to January 2022) and region (Eastern Cape, South Africa). While the insights gained from this data are valuable, they may not be directly generalisable to other time periods or geographical contexts. Future studies could expand the scope by incorporating more recent data and extending the analysis to other provinces or countries. This would allow for a more comprehensive understanding of fuel usage patterns and the identification of potential regional differences in fleet management practices and fraud prevalence.

Another limitation of this study is the presence of missing or incomplete data for certain variables, such as district allocations for some vehicles in the fleet. This could potentially introduce bias or reduce the accuracy of the clustering and predictive models. To address this issue, future research should focus on improving data collection processes and exploring advanced imputation techniques for handling missing data more effectively. Collaboration with government agencies and fleet managers is crucial to ensure consistent and thorough data collection, which would greatly enhance the quality and reliability of future analyses.

A methodological limitation relates to the approach used for determining the optimal number of clusters. The elbow method, while intuitive and computationally efficient, relies on visual inspection and can be subjective, especially in datasets where the separation between clusters is less pronounced. Although this method was deemed sufficient given the relatively clean dataset and small expected

number of clusters, alternative approaches like the Gap Statistic or Gower distance could have been considered. The Gap Statistic is more rigorous and less subjective, providing a formal statistical measure to determine the optimal number of clusters by comparing within-cluster dispersion against a null reference distribution. Gower distance, on the other hand, is particularly well-suited for datasets with mixed data types, as it accounts for both numerical and categorical variables, making it a versatile metric for calculating dissimilarities. Integrating these methods in future research could enhance the robustness of clustering results, especially when working with more complex or heterogeneous datasets.

The current study primarily relies on quantitative data, such as transaction records and vehicle attributes. While this data provides valuable insights, incorporating qualitative data could offer additional context and validation for the findings. Future research could include interviews with fleet managers, drivers, and other stakeholders to gain a deeper understanding of the challenges, practices, and perceptions related to fuel management and fraud prevention. This qualitative data could complement the quantitative findings and provide a more holistic view of the issues at hand.

To further enhance the practical applicability of the findings, future research could explore the integration of real-time monitoring systems and Internet of Things (IoT) devices for early detection of fuel fraud. By collecting and analysing data from sensors and telematics systems installed in vehicles, fleet managers could detect anomalies and suspicious activities as they occur, enabling quick intervention and minimising financial losses. This real-time monitoring approach would complement the predictive models developed in this study, providing a more proactive and responsive fraud detection system.

Another potential avenue for future research is the examination of the relationship between vehicle maintenance, fuel efficiency, and fraud risk. Analysing data on vehicle age, maintenance records, and fuel consumption patterns could provide insights into the impact of vehicle condition on fuel efficiency and the potential for fraudulent activities. This information could guide fleet renewal strategies and help prioritise maintenance interventions to optimise fuel usage and reduce the risk of fraud. By incorporating vehicle maintenance data into the predictive models, future studies could develop more comprehensive and accurate fraud detection systems that consider the interplay between vehicle condition, fuel efficiency, and fraudulent behaviour.

In terms of indicators, future research could include a broader range of fraud indicators and data sources to enhance the accuracy and effectiveness of the predictive models. The incorporation of additional variables, such as driver behaviour, fuel card usage patterns, and external factors such as weather conditions and road quality, could provide a more comprehensive view of the factors contributing to fuel fraud. Collaborating with experts from diverse domains, such as criminology and psychology, could also provide valuable perspectives and help identify novel fraud indicators and modelling techniques. This interdisciplinary approach could lead to the development of more sophisticated and effective fraud detection systems.

Future research could also focus on the development of user-friendly interfaces and decision support systems that integrate the fraud detection models and indicators developed in this study. By creating intuitive dashboards and visualisation tools, fleet managers and investigators could more easily interpret the results, prioritise investigations, and make data-driven decisions. These tools could streamline the fraud detection process, reducing the time and resources required to identify and investigate suspicious activities. Additionally, the development of automated alert systems that notify fleet managers of potential fraud in real-time could further enhance the practical utility of the predictive models.

In conclusion, this dissertation contributes to the field of fuel management and fraud detection in government fleets, particularly as the first analysis of fuel transaction data focusing on fuel theft in the Eastern Cape, South Africa. The findings and methodologies presented offer valuable insights into fuel usage patterns, anomaly detection, and fraud prevention strategies. However, there remain numerous opportunities for future research to build upon this work and address the limitations of the current study.

Expanding the scope of the data, incorporating qualitative insights, exploring real-time monitoring systems, and investigating the role of vehicle maintenance are just a few of the potential avenues for further investigation. By addressing these limitations and building upon the findings of this study, future research can continue to advance our understanding of fuel usage patterns, anomaly detection, and fraud prevention in fleet management.

Ultimately, this ongoing effort will support the development of more efficient, cost-effective, and transparent fleet operations, benefiting government agencies, taxpayers, and society as a whole. In a world where public trust in government institutions is often strained, every step towards greater accountability and responsible use of resources is a step in the right direction - and this dissertation is a part of that journey.

References

- Abdallah, A., Maarof, M. A., & Zainal, A. (2016). Fraud detection system: A survey. *Journal of Network and Computer Applications*, 68, 90–113. <https://doi.org/https://doi.org/10.1016/j.jnca.2016.04.007>
- Adewumi, A. O., & Akinyelu, A. A. (2016). A survey of machine-learning and nature-inspired based credit card fraud detection techniques. *International Journal of System Assurance Engineering and Management*, 8, 937–953.
- Arthur, D., & Vassilvitskii, S. (2007). K-means++: The advantages of careful seeding. *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, 1027–1035.
- Avis. (2023). *Part 1: Most common ways fuel fraud is committed*. Retrieved June 8, 2023, from <https://blog.avisfleet.co.za/latest-news/articles/part-1-most-common-ways-fuel-fraud-is-committed>
- Bhusari, V., & Patil, S. (2011). Study of hidden markov model in credit card fraudulent detection. *Int J Comput Appl*, 20, 33–36.
- Biernacki, C., Celeux, G., & Govaert, G. (2003). Choosing starting values for the em algorithm for getting the highest likelihood in multivariate gaussian mixture models. *Computational Statistics & Data Analysis*, 41(3-4), 561–575.
- Bishop, C. M. (1995). *Neural networks for pattern recognition*. Oxford University Press.
- Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer.
- Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent. *Proceedings of COMPSTAT'2010*, 177–186.
- Bottou, L., Curtis, F. E., & Nocedal, J. (2018). Optimization methods for large-scale machine learning. *Siam Review*, 60(2), 223–311.
- Breiman, L. (2001). Random forests. *Machine Learning*, 45(1), 5–32. <https://doi.org/10.1023/A:1010933404324>
- Burges, C. J. (1998). A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2), 121–167.
- Chen, T., & Guestrin, C. (2016). Xgboost: A scalable tree boosting system. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785–794.
- Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3), 273–297.
- Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1), 1–22.
- Diesel Guard. (2024). Complete diesel theft protection [Accessed: 2024-03-14]. <https://dieselguard.co.za/>

-
- Domingos, P., & Pazzani, M. (1997). On the optimality of the simple bayesian classifier under zero-one loss. *Machine Learning*, 29(2), 103–130.
- Duman, E., & Ozcelik, M. (2011). Detecting credit card fraud by genetic algorithm and scatter search. *Exp Syst Appl*, 38, 13057–13063.
- Ester, M., Kriegel, H.-P., Sander, J., & Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, 226–231.
- Fraley, C., & Raftery, A. E. (1998). How many clusters? Which clustering method? Answers via model-based cluster analysis. *The Computer Journal*, 41(8), 578–588.
- Fraley, C., & Raftery, A. E. (2002). Model-based clustering, discriminant analysis, and density estimation. *Journal of the American Statistical Association*, 97(458), 611–631.
- Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *Annals of statistics*, 1189–1232.
- Friedman, J. H. (2002). Stochastic gradient boosting. *Computational statistics & data analysis*, 38(4), 367–378.
- Gan, G., Ma, C., & Wu, J. (2007). *Data clustering: Theory, algorithms, and applications*. SIAM.
- Ganesh, K., & Sena, P. (2012). Novel artificial neural networks and logistic approach for detecting credit card deceit. *Int J Comput Sci Netw Secur*, 13, 58–65.
- Ghosh, J., Strehl, A., & Merugu, S. (2002). Cluster ensembles. *Proceedings of the 2002 SIAM International Conference on Data Mining*, 126–137.
- Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.
- Hand, D. J., & Yu, K. (2001). Idiot’s bayes—not so stupid after all? *International statistical review*, 69(3), 385–398.
- Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: Data mining, inference, and prediction*. Springer.
- Hearst Autos Research. (2024). Diesel vs. gasoline: Everything you need to know [Accessed: 2024-03-16].
- Hendricks, C. (2021). Stagnant economy in eastern cape continues to have dire effects on residents [Accessed: 15-Mar-2024]. <https://www.sabcnews.com/sabcnews/stagnant-economy-in-eastern-cape-continues-to-have-dire-effects-on-residents-mec/>
- Hinton, G. E., Osindero, S., & Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural computation*, 18(7), 1527–1554.
- Hossin, M., & Sulaiman, M. (2015). A review on evaluation metrics for data classification evaluations. *International Journal of Data Mining & Knowledge Management Process*, 5(2), 1.
- iFleet. (2023). *Ifleet - fleet management solutions*. Retrieved June 8, 2023, from <https://ifleet.co.za/>
- Jain, A. K., Murty, M. N., & Flynn, P. J. (1999). Data clustering: A review. *ACM Computing Surveys (CSUR)*, 31(3), 264–323.
- Kaufman, L., & Rousseeuw, P. J. (2009). *Finding groups in data: An introduction to cluster analysis* (Vol. 344). John Wiley & Sons.
- Khan, A., Mahajan, V., Shaikh, S., & Koli, A. (2013). Credit card fraud detection system through observation probability using hidden markov model. *Int J Thesis Proj Diss*, 1, 7–16.
- Khan, A., Akhtar, N., & Qureshi, M. (2014). Real-time credit-card fraud detection using artificial neural network tuned by simulated annealing algorithm. *Conference on recent trends in information, telecommunication and computing, ITC*, 113–121.

-
- Khan, M., Pathan, J., & Ahmed, A. (2014). Credit card fraud detection system using hidden markov model and k-clustering. *Int J Adv Res Comput Commun Eng*, 3, 5458–5461.
- Lu, Q., & Ju, C. (2011). Research on credit card fraud detection model based on class weighted support vector machine. *J Converge Inf Technol*, 6, 62–68.
- Madke, P. B., & Jaybhaye, M. D. (2016). Simulation of a fuel theft detection system for commercial vehicles. *Proceedings of the Modern Era Research in Mechanical Engineering-2016 (MERME-16)*.
- Maes, S., Tuyls, K., Vanschoenwinkel, B., & Manderick, B. (2002a). Credit card fraud detection using bayesian and neural networks. *Proceedings of the 1st international naio congress on neuro fuzzy technologies*, 261–270.
- Maes, S., Tuyls, K., Vanschoenwinkel, B., & Manderick, B. (2002b). Credit card fraud detection using bayesian and neural networks. *Proceedings of the 1st international naio congress on neuro fuzzy technologies*, 261–270.
- Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to information retrieval*. Cambridge University Press.
- Mazele, O., & Amoah, C. (2022). The causes of poor infrastructure management and maintenance in south african municipalities. *Property Management*, 40(2), 192–206. <https://doi.org/10.1108/PM-06-2021-0042>
- McLachlan, G. J., & Peel, D. (2004). *Finite mixture models*. John Wiley & Sons.
- Mhamane, S., & Lobo, L. (2012a). Fraud detection in online banking using hmm. *International conference on information and network technology (ICINT 2012)*, 200–204.
- Mhamane, S., & Lobo, L. (2012b). Use of hidden markov model as internet banking fraud detection. *Int J Comput Appl*, 45, 5–10.
- Mhlauli, N. (2017). Language and access in the public healthcare system in south africa with a particular focus on primary public health facilities in grahamstown and cofimvaba in the eastern cape.
- Mitchell, T. M. (2017). *Machine learning yearning*. Technical report.
- Modi, H., Lakhani, S., Patel, N., & Patel, V. (2013). Fraud detection in credit card system using web mining. *Int J Innov Res Comput Commun Eng*, 1, 175–179.
- Moyo, L., & Osunkunle, O. (2021). Covid-19 pandemic in South Africa: An exploration on citizen adherence to protocols and communication. *Journal of Human Ecology*, 77–88.
- Müllner, D. (2011). Modern hierarchical, agglomerative clustering algorithms. *arXiv preprint arXiv:1109.2378*.
- Ngcobo, B. L. (2014). The impact of infrastructure on agricultural economic development in bizana, eastern cape.
- Ngiam, X. W. (2005). People living with, not dying from disease: Patient empowerment through anti-retroviral therapy and possibilities for social change.
- Ngqula, Z. N. (2022). Umama wekhaya: Local subjectivities, water infrastructures and grounded perceptions of development in agnes rest, eastern cape, south africa. *Anthropology Southern Africa*, 45, 208–218.
- Nielsen, D. (2016). Tree boosting with xgboost-why does xgboost win” every” machine learning competition?
- Nwankpa, C., Ijomah, W., Gachagan, A., & Marshall, S. (2018). Activation functions: Comparison of trends in practice and research for deep learning. *arXiv preprint arXiv:1811.03378*.

-
- Oyediran-Tidings, S. O., Nekhwevha, F., Ondari-Okemwa, E., & Salubi, O. G. (2021). Access to educational information enabled by ICT tools in the fort beaufort education district (fbed), Eastern Cape, South Africa. *Information Development*, 37, 402–416.
- Oyelana, A. A., & Fiseha, G. G. (2014). An Investigation into the Effect of Small and Medium Enterprises (SMEs) on the Socio-Economic Development of Alice in Eastern Cape Province, South Africa. *Mediterranean journal of social sciences*, 5, 674.
- Patel, R., & Singh, D. (2013). Credit card fraud detection & prevention of fraud using genetic algorithm. *Int J Soft Comput Eng*, 2, 2231–2307.
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.
- Powers, D. M. (2020). Evaluation: From precision, recall and f-measure to roc, informedness, markedness and correlation. *arXiv preprint arXiv:2010.16061*.
- Pun, J.-F. (2011). *Improving credit card fraud detection using a metalearning strategy* [Master's thesis, University of Toronto].
- Qian, N. (1999). On the momentum term in gradient descent learning algorithms. *Neural Networks*, 12(1), 145–151.
- RamaKalyani, K., & UmaDevi, D. (2012). Fraud detection of credit card payment system by genetic algorithm. *Int J Sci Eng Res*, 3, 1–6.
- ReviewNitro. (2023). *Best locking gas cap - reviewnitro*. Retrieved June 8, 2023, from <https://reviewnitro.com/best-locking-gas-cap/>
- Rish, I. (2001). An empirical study of the naive bayes classifier. *IJCAI 2001 workshop on empirical methods in artificial intelligence*, 3(22), 41–46.
- Robbins, H., & Monro, S. (1951). A stochastic approximation method. *The Annals of Mathematical Statistics*, 400–407.
- Rokach, L., & Maimon, O. (2005). Clustering methods. *Data mining and knowledge discovery handbook*, 321–352.
- Ross, R. J. (2013). The borders of race in colonial south africa: The rebellion.
- Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323(6088), 533–536.
- Sahin, Y., & Duman, E. (2011). Detecting credit card fraud by decision trees and support vector machines. *Proceedings of the international multiConference of engineers and computer scientists (IMECS 2011)*, 1, 1–6.
- Saini, M., & Khan, S. (2021). Gsm based fuel theft detection. *2021 1st International Conference on Power Electronics and Energy (ICPEE)*. <https://doi.org/10.1109/ICPEE50452.2021.9358644>
- Schölkopf, B., & Smola, A. J. (2002). *Learning with kernels: Support vector machines, regularization, optimization, and beyond*. MIT Press.
- Seeja, K., & Zareapoor, M. (2014). Fraudminer: A novel credit card fraud detection model based on frequent itemset mining. *Sci World J*, 2014, 10.
- Sen, S., & Dash, S. (2013). Meta learning algorithms for credit card fraud detection. *Meta*, 6, 16–20.
- SmartFuel. (2023). *Smart surveillance - theft and fraud management*. Retrieved June 8, 2023, from <https://www.smartfuel.co.za/Products/SmartSurveillanceTheftandFraudManagement.aspx>
- Sneath, P. H. A., & Sokal, R. R. (1973). *Numerical taxonomy: The principles and practice of numerical classification*. W. H. Freeman.

-
- Sokolova, M., & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, 45(4), 427–437.
- Soltani, N., Akbari, M., & Javan, M. (2012). A new user-based model for credit card fraud detection based on artificial immune system. *2012 16th CSI international symposium on artificial intelligence and signal processing (AISP)*, 029–033.
- Soltani Halvaiee, N., & Akbari, M. (2014). A novel model for credit card fraud detection using artificial immune systems. *Appl Soft Comput*, 24, 40–49.
- StatsSA. (2023, October). Eastern Cape home to over 7.2 million people [Accessed: 2023-03-15]. <https://www.statssa.gov.za/?p=17341>
- Stolfo, S., Fan, W., Lee, W., Prodromidis, A., & Chan, P. (1997). Credit card fraud detection using meta-learning: Issues and initial results. *AAAI Workshop on AI approaches to fraud detection and risk management*, 83–90.
- Strehl, A., & Ghosh, J. (2002). Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research*, 3, 583–617.
- Sutskever, I., Martens, J., Dahl, G., & Hinton, G. (2013). On the importance of initialization and momentum in deep learning. *International Conference on Machine Learning*, 1139–1147.
- TankSafe South Africa. (2024). Home - tanksafe south africa [Accessed: 2024-03-14]. <https://tanksafe.co.za/>
- Tracking, L. (2023). *Fleet fuel management - landmark tracking*. Retrieved June 8, 2023, from <https://www.landmarktracking.co.za/fleet-fuel-management/>
- Van Vlasselaer, V., Bravo, C., Caelen, O., Eliassi-Rad, T., Akoglu, L., Snoeck, M., et al. (2015). Apaté: A novel approach for automated credit card transaction fraud detection using network-based extensions. *Decis Support Syst*, 75, 38–48.
- Vapnik, V. (1995). *The nature of statistical learning theory*. Springer.
- Ward Jr, J. H. (1963). Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301), 236–244.
- Wong, N., Ray, P., Stephens, G., & Lewis, L. (2012). Artificial immune systems for the detection of credit card fraud: An architecture, prototype and preliminary results. *Inf Syst J*, 22, 53–76.
- Xu, L., & Jordan, M. I. (1996). A robust em clustering algorithm for gaussian mixture models. *Pattern Recognition*, 29(7), 1033–1042.
- Xu, R., & Wunsch, D. (2005). Survey of clustering algorithms.
- Zaki, M. J., & Meira Jr, W. (2014). *Data mining and analysis: Fundamental concepts and algorithms*. Cambridge University Press.
- Zareapoor, M., & Shamsolmoali, P. (2005). Application of credit card fraud detection: Based on bagging ensemble classifier. *Procedia Comput Sci*, 48, 679–685.
- Zhang, H. (2004). The optimality of naive bayes. *AA*, 1(2), 3.
- Zhang, T., Ramakrishnan, R., & Livny, M. (1996). Birch: An efficient data clustering method for very large databases. *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data*, 103–114.
- Zhang, T., Ramakrishnan, R., & Livny, M. (1997). Birch: A new data clustering algorithm and its applications. *Data Mining and Knowledge Discovery*, 1(2), 141–182.

Appendix A

Other Modelling Methods

A.1 Naive Bayes Model

This section is mainly based on a paper by Domingos and Pazzani (1997).

A.1.1 Introduction

The Naive Bayes classifier is a probabilistic machine learning model rooted in Bayes' theorem. It excels in classification tasks due to its simplicity and surprising effectiveness across various domains. At its core lies the central assumption of conditional independence between features.

A.1.2 Underlying Assumptions

Conditional Independence:

The central premise of the Naive Bayes model is that all features within a given class are conditionally independent of each other. In other words, the presence or absence of a particular feature does not influence the presence or absence of another feature, given the class label. Mathematically, this assumption can be expressed as:

$$P(x_i|y, x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n) = P(x_i|y) \quad (\text{A.1})$$

where x_i is the i -th feature, y is the class label, and $x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n$ are the other features.

In practice, this strong assumption of conditional independence rarely holds perfectly in real-world datasets. Features often exhibit some level of correlation or dependence within a given class. However, despite this violation of the assumption, the Naive Bayes classifier has been shown to be remarkably robust and effective in many practical applications (Domingos & Pazzani, 1997).

The robustness of the Naive Bayes classifier can be attributed to several factors. Firstly, even when the conditional independence assumption is violated, the classifier can still perform well if the dependencies

among features are not too strong or if the dependencies cancel each other out on average (Rish, 2001). Secondly, the Naive Bayes classifier's decision boundary is determined by the relative probabilities of the classes, rather than the absolute probabilities. As a result, even if the probability estimates are biased due to the violated independence assumption, the classifier can still make correct predictions as long as the relative probabilities are reasonably accurate (Zhang, 2004).

Furthermore, the Naive Bayes classifier has been shown to have a lower asymptotic error rate compared to other classifiers when the conditional independence assumption holds, and it can still be competitive even when the assumption is violated (Domingos & Pazzani, 1997). The simplicity of the Naive Bayes model also helps to avoid overfitting, especially when dealing with high-dimensional data or limited training samples (Hand & Yu, 2001).

A.1.3 Algorithm and Mechanism

1. **Data Representation:** A dataset is represented where each data point (instance) comprises a set of features $(x_1, x_2, \dots, x_n)$ and an associated class label y .
2. **Prior Probability Estimation:** For each class y , compute the prior probability, $P(y)$, representing the likelihood of encountering that class within the dataset.
3. **Likelihood Calculation:** For each feature x_i and class y , calculate the likelihood, $P(x_i|y)$, denoting the probability of observing the feature value x_i given that the instance belongs to class y .
4. **Posterior Probability Computation:** Given a new data point x , apply Bayes' theorem to calculate the posterior probability for each class:

$$P(y|x) = \frac{P(y) \prod_{i=1}^n P(x_i|y)}{P(x)} \quad (\text{A.2})$$

where $P(x)$ is the evidence, a normalisation factor.

5. **Classification:** Assign the data point x to the class with the highest posterior probability.

Mathematical Formulation

The Naive Bayes classifier is based on Bayes' theorem, which states:

$$P(y|x_1, \dots, x_n) = \frac{P(y)P(x_1, \dots, x_n|y)}{P(x_1, \dots, x_n)} \quad (\text{A.3})$$

where y is the class variable and $x_1, \dots, x_n$ are the features.

The naive assumption of conditional independence allows the simplification of the joint probability $P(x_1, \dots, x_n|y)$ as:

$$P(x_1, \dots, x_n|y) = \prod_{i=1}^n P(x_i|y) \quad (\text{A.4})$$

Substituting this into Bayes' theorem gives the Naive Bayes classifier:

$$P(y|x_1, \dots, x_n) = \frac{P(y) \prod_{i=1}^n P(x_i|y)}{P(x_1, \dots, x_n)} \quad (\text{A.5})$$

Since $P(x_1, \dots, x_n)$ is constant for a given input, the following classification rule can be u:

$$\hat{y} = \underset{y}{\operatorname{argmax}} P(y) \prod_{i=1}^n P(x_i|y) \quad (\text{A.6})$$

where $\hat{y}$ is the predicted class label.

The prior probability $P(y)$ and the likelihood $P(x_i|y)$ are estimated from the training data. The method for estimating these probabilities depends on the type of feature (discrete or continuous) and the assumed distribution.

For discrete features, the likelihood is often estimated using the Multinomial distribution:

$$P(x_i|y) = \frac{\operatorname{count}(x_i, y) + \alpha}{\operatorname{count}(y) + \alpha n} \quad (\text{A.7})$$

where $\operatorname{count}(x_i, y)$ is the number of times feature x_i appears in class y , $\operatorname{count}(y)$ is the total count of class y , n is the number of possible values for feature x_i , and α is a smoothing parameter (usually 1, for Laplace smoothing).

For continuous features, the Gaussian distribution is commonly used:

$$P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right) \quad (\text{A.8})$$

where μ_y and σ_y^2 are the mean and variance of feature x_i in class y , estimated from the training data.

These mathematical formulations underpin the mechanism of the Naive Bayes classifier, which combines the prior probabilities and feature likelihoods to make predictions based on the maximum a posteriori (MAP) rule.

A.1.4 Python Implementation

For the Python implementation, the **GaussianNB** function in the **sklearn** package was used (Pedregosa et al., 2011). More details on the exact implementation can be found on the dissertation GitHub repository.

A.2 Linear Support Vector Machine Model

This section is based on a paper by Burges (1998).

A.2.1 Introduction

Linear Support Vector Machines (SVMs) are powerful supervised learning algorithms designed for binary classification problems. The key idea is to find a hyperplane in the feature space that optimally separates the two classes of data points, maximising the margin between the hyperplane and the closest data points (support vectors). SVMs were originally developed by Vladimir Vapnik and colleagues (Vapnik, 1995). While this section focuses on linear SVMs, it is worth noting that SVMs can be extended to handle non-linear decision boundaries using kernel functions (Schölkopf & Smola, 2002).

A.2.2 Underlying Assumptions

1. **Linear Separability:** The fundamental assumption in linear SVMs is that the data is linearly separable. This means a hyperplane can be drawn to clearly divide the data points belonging to different classes. In some real-world cases, perfect linear separability might not be achievable. In such cases, SVMs can still find a hyperplane that minimises the classification error by allowing some data points to be misclassified or fall within the margin, known as the "soft margin" approach (Cortes & Vapnik, 1995).
2. **Large Margin:** SVMs strive to find the hyperplane that not only separates the classes but does so with the widest possible margin. This margin principle grants SVMs better generalisation capabilities.

A.2.3 Algorithm and Mechanism

1. **Problem Formulation:** Consider a dataset with data points $(\mathbf{x}_i, y_i)$, where $\mathbf{x}_i$ is the feature vector and $y_i \in -1, +1$ is the class label. The goal is to find a hyperplane defined by the equation:

$$\mathbf{w}^T \mathbf{x} + b = 0 \tag{A.9}$$

where $\mathbf{w}$ is the weight vector perpendicular to the hyperplane and b is the bias term. This equation represents the decision boundary (hyperplane) in the feature space, and the goal is to find the values of $\mathbf{w}$ and b that define the optimal hyperplane.

2. **Margin Maximisation:** To maximise the margin, we want to ensure that for all data points:

$$y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1 \tag{A.10}$$

This constraint ensures that the data points are correctly classified and are at least a distance of 1 from the decision boundary. The points exactly on the margin (i.e., $y_i(\mathbf{w}^T \mathbf{x}_i + b) = 1$) are called support vectors (Burges, 1998).

-
3. **Optimisation:** The SVM optimisation problem is formulated as a quadratic programming problem seeking to minimise $|\mathbf{w}|^2$ (the magnitude of the weight vector) under the margin constraints. The primal form of the optimisation problem is:

$$\min_{\mathbf{w}, b} \frac{1}{2} |\mathbf{w}|^2 \quad (\text{A.11})$$

$$\text{subject to: } y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1, \forall i \quad (\text{A.12})$$

Solving the dual form of this optimisation problem using Lagrangian multipliers leads to a quadratic programming problem (Burges, 1998). This can be solved using techniques like Lagrangian multipliers and introducing the concept of "soft margins" to handle slight non-separability.

4. **Decision Rule:** For a new data point $\mathbf{x}$, the classification decision is made as follows:

$$f(\mathbf{x}) = \text{sign}(\mathbf{w}^T \mathbf{x} + b) \quad (\text{A.13})$$

SVMs offer several advantages, including their ability to handle high-dimensional data, their robustness to outliers, and their good generalisation performance (Vapnik, 1995).

A.2.4 Python Implementation

For the Python implementation, the **LinearSVC** function in the **sklearn** package was used. More details on the exact implementation can be found on the dissertation GitHub repository.

A.3 Stochastic Gradient Descent Classifier

This section is based on a paper by Bottou (2010).

A.3.1 Introduction

Stochastic Gradient Descent (SGD) is a simple yet effective optimisation algorithm commonly used for training linear classifiers, such as logistic regression and support vector machines. SGD iteratively updates the model parameters based on the gradients computed from individual training examples, making it efficient for large-scale learning problems. Compared to batch gradient descent methods, SGD offers impressive scalability advantages, particularly for large datasets (Bottou, 2010). It is widely used in various machine learning frameworks and libraries.

A.3.2 Underlying Assumptions

1. **Convexity:** SGD assumes that the objective function being optimised is convex. This ensures that the algorithm converges to a global minimum.
2. **Stochastic Approximation:** SGD relies on the assumption that the stochastic gradients computed from individual examples provide an unbiased estimate of the true gradient of the objective function. The stochastic gradients introduce noise into the update process but converge to the true gradient on average (Robbins & Monro, 1951).

A.3.3 Algorithm and Mechanism

1. **Initialisation:** Initialise the model parameters (weights) $\mathbf{w}$ randomly or to zero.
2. **Iterative Updates:** For each training example $(\mathbf{x}_i, y_i)$ or mini-batch of examples, perform the following steps:
 - Compute the gradient $\mathbf{g}_i$ (or average gradient for mini-batch) of the loss function with respect to the model parameters $\mathbf{w}$ for the current example(s).
 - Update the model parameters using the gradient and learning rate η :

$$\mathbf{w} \leftarrow \mathbf{w} - \eta \mathbf{g}_i \tag{A.14}$$

Mini-batch SGD, where the gradients are computed based on a subset (mini-batch) of examples, provides a balance between the stability of batch updates and the efficiency of stochastic updates (Bottou et al., 2018).

3. **Learning Rate Scheduling:** Adjust the learning rate η according to a predefined schedule or adaptive strategy. Common learning rate schedules include constant, decaying, or adaptive learning rates (e.g., AdaGrad, RMSprop) (Bottou et al., 2018).
4. **Iteration and Convergence:** Repeat steps 2-3 for a fixed number of iterations or until a convergence criterion is met. Common convergence criteria include a maximum number of epochs, a minimum change in the objective function, or a validation-based early stopping.

-
5. **Prediction:** To make predictions for a new instance $\mathbf{x}$, compute the dot product between the learned weights $\mathbf{w}$ and the instance features, and apply the appropriate decision function (e.g., sign function for binary classification).

Mathematical Formulation

Given a training dataset $(\mathbf{x}_i, y_i)_{i=1}^N$, the objective of SGD is to minimise the empirical risk:

$$R(\mathbf{w}) = \frac{1}{N} \sum_{i=1}^N L(y_i, f(\mathbf{x}_i; \mathbf{w})) \quad (\text{A.15})$$

where L is the loss function (e.g., hinge loss for SVM, logistic loss for logistic regression), and $f(\mathbf{x}_i; \mathbf{w})$ is the prediction function parameterised by weights $\mathbf{w}$.

The update rule for the model parameters at iteration t using mini-batch SGD is given by:

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \frac{\eta^{(t)}}{m} \sum_{i \in \mathcal{B}^{(t)}} \nabla L(y_i, f(\mathbf{x}_i; \mathbf{w}^{(t)})) \quad (\text{A.16})$$

where $\eta^{(t)}$ is the learning rate at iteration t , $\mathcal{B}^{(t)}$ is the mini-batch of examples at iteration t , m is the mini-batch size, and ∇L denotes the gradient of the loss function (Bottou et al., 2018).

Variants of SGD, such as SGD with momentum (Qian, 1999) or Nesterov accelerated gradient (Sutskever et al., 2013), incorporate additional techniques to improve convergence speed and stability.

A.3.4 Python Implementation

For the Python implementation, the **SGDClassifier** function in the **sklearn** package was used (Pedregosa et al., 2011). More details on the exact implementation can be found on the dissertation GitHub repository.

Appendix B

Other Predictive Modelling Results

B.1 Linear SVM

The Linear SVM model's performance in detecting fraudulent fuel transactions in the Eastern Cape government fleet can be contrasted with the results obtained by the Multi-layer Perceptron (MLP) model. While both models demonstrate strong capabilities in identifying fraudulent activities, there are notable differences in their performance across the different classes of fraud indicators.

In terms of accurately identifying transactions with no fraud indicators (class 0), the Linear SVM model achieves an accuracy of 89.11%, which is lower than the MLP model's accuracy of 95.57%. This suggests that the MLP model is more effective in minimising false positives and maintaining a high true positive rate for legitimate transactions. The precision, recall, and F1-score for class 0 are also higher for the MLP model (0.90, 0.96, and 0.93, respectively) compared to the Linear SVM model (0.87, 0.89, and 0.88, respectively), further highlighting the MLP model's superior performance in this regard.

When it comes to identifying transactions with a single fraud indicator (class 1), the Linear SVM model's accuracy of 47.51% is lower than the MLP model's accuracy of 54.54%. This indicates that the MLP model is more effective in accurately identifying these anomalous transactions, although both models have room for improvement. The precision, recall, and F1-score for class 1 are also higher for the MLP model (0.72, 0.55, and 0.62, respectively) compared to the Linear SVM model (0.48, 0.48, and 0.48, respectively), emphasising the MLP model's better performance in handling transactions with a single fraud indicator.

The most striking difference between the two models is observed in their ability to identify transactions with 2 fraud indicators (class 2). The Linear SVM model accurately predicts only 1.65% of these instances, while the MLP model achieves a markedly higher accuracy of 49.41%. This substantial gap in performance is further reflected in the precision, recall, and F1-score for class 2, with the MLP model (0.64, 0.49, and 0.56, respectively) greatly outperforming the Linear SVM model (0.26, 0.02, and 0.03, respectively). These results underscore the MLP model's superior ability to identify the most suspicious transactions, which is crucial for effective fraud detection and prevention.

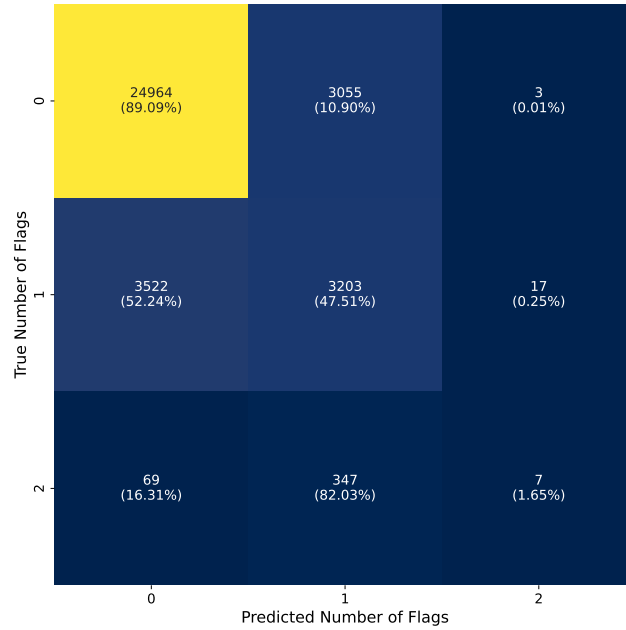


Figure B.1: The confusion matrix with percentages of total observations for the Linear SVM model on the test set.

The ROC curves and AUC values provide additional insights into the performance differences between the two models. While the Linear SVM model achieves a good AUC of 0.80 for class 0, the MLP model's AUC of 0.92 indicates its excellent ability to distinguish between legitimate transactions and those with at least one fraud indicator. Similarly, the MLP model's AUC for class 1 (0.90) is higher than that of the Linear SVM model (0.78), demonstrating the MLP model's stronger ability to differentiate between transactions with 1 fraud indicator and those with either no indicators or 2 indicators. The most noticeable difference is observed for class 2, where the MLP model achieves an exceptional AUC of 0.99, indicating near-perfect performance in distinguishing between transactions with 2 fraud indicators and those with fewer indicators. In contrast, the Linear SVM model's AUC for class 2 is 0.94, which, while still excellent, is lower than that of the MLP model.

Both models face challenges due to the imbalanced nature of the dataset, with class 2 having substantially fewer instances than class 0 and class 1. However, the MLP model demonstrates greater robustness in handling this imbalance, as evidenced by its superior performance across all classes and the smaller discrepancies between the accuracy values in the confusion matrix and the high AUC values.

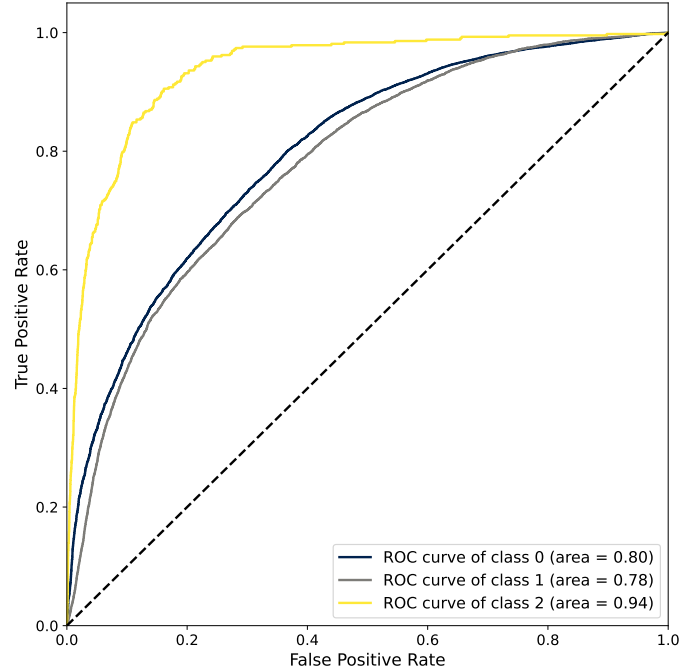


Figure B.2: The ROC curves for the Linear SVM model for each class on the test set using a one-vs-all approach, where the class of interest is treated as the positive class, and all other classes are combined to form the negative class.

In summary, while the Linear SVM model exhibits strong performance in detecting fraudulent fuel transactions, particularly for class 0 and class 2, the Multi-layer Perceptron model consistently outperforms it across all classes and evaluation metrics. The MLP model's superior ability to accurately identify transactions with no fraud indicators, 1 fraud indicator, and 2 fraud indicators, as well as its robustness in handling the imbalanced dataset, make it the more effective choice for detecting fraudulent activities in the Eastern Cape government fleet fuel management system.

The comparison of the Linear SVM model's performance with the findings from credit card fraud detection studies using SVM models further highlights the potential and challenges of this approach in various fraud detection contexts. The strong performance of the Linear SVM model in accurately identifying transactions with no fraud indicators (class 0) and its excellent ability to distinguish between transactions with 2 fraud indicators (class 2) and those with fewer indicators are consistent with the results reported by Sahin and Duman (2011). Their study found that while decision tree models initially outperformed SVM models in terms of accuracy and the number of fraudulent transactions correctly identified, the performance gap decreased as the training dataset size increased. This observation aligns with the Linear SVM model's performance in the Eastern Cape government fleet fuel management system, suggesting that SVM models can be effective in fraud detection tasks, particularly with larger datasets.

However, the moderate performance of the Linear SVM model in identifying transactions with a single fraud indicator (class 1) is similar to the challenges faced by SVM models in accurately classifying instances from imbalanced datasets, as reported by Lu and Ju (2011). Their study proposed techniques

such as class weighting to improve accuracy and handle data imbalance, which could potentially enhance the performance of the Linear SVM model in identifying transactions with a single fraud indicator. These findings underscore the importance of addressing the imbalanced nature of fraud detection datasets to optimise the performance of SVM models across all classes of fraudulent activities.

The comparison of the Linear SVM model's results with those of the Multi-layer Perceptron (MLP) model and the findings from credit card fraud detection studies using SVM models emphasises the need for a comprehensive evaluation of different machine learning techniques in various fraud detection contexts. While the MLP model consistently outperforms the Linear SVM model in the Eastern Cape government fleet fuel management system, the SVM model's performance is still noteworthy and comparable to the results reported in credit card fraud detection studies. This highlights the importance of considering the specific characteristics of the dataset, the nature of the fraudulent activities, and the strengths and limitations of each machine learning approach when selecting the most appropriate model for a given fraud detection task.

B.2 Naive Bayes

The Naive Bayes model, a probabilistic classifier based on applying Bayes' theorem with strong independence assumptions between features, demonstrates varying levels of effectiveness in detecting fraudulent fuel transactions in the Eastern Cape government fleet. Its performance can be evaluated in comparison to the Multi-layer Perceptron (MLP) model across different classes of fraud indicators.

When examining the model's accuracy in identifying transactions with no fraud indicators (class 0), the Naive Bayes model achieves a modest 64.11

Shifting focus to transactions with a single fraud indicator (class 1), the Naive Bayes model's accuracy of 30.36% reveals its challenges in accurately identifying these suspicious transactions. In comparison, the MLP model achieves a notably higher accuracy of 54.54%, indicating its superior performance in this regard. The Naive Bayes model's precision (0.28), recall (0.30), and F1-score (0.29) for class 1 further emphasize its limited effectiveness compared to the MLP model.

Interestingly, the Naive Bayes model demonstrates a notable ability to identify transactions with 2 fraud indicators (class 2), achieving an accuracy of 69.03%, surpassing the MLP model's accuracy of 49.41%. However, it is essential to consider the Naive Bayes model's precision (0.04) and F1-score (0.07) for class 2. These low values indicate that the Naive Bayes model tends to misclassify many transactions from other classes as class 2, leading to a high number of false positives. In contrast, the MLP model exhibits a more balanced performance for class 2, with higher precision (0.64), recall (0.49), and F1-score (0.56).

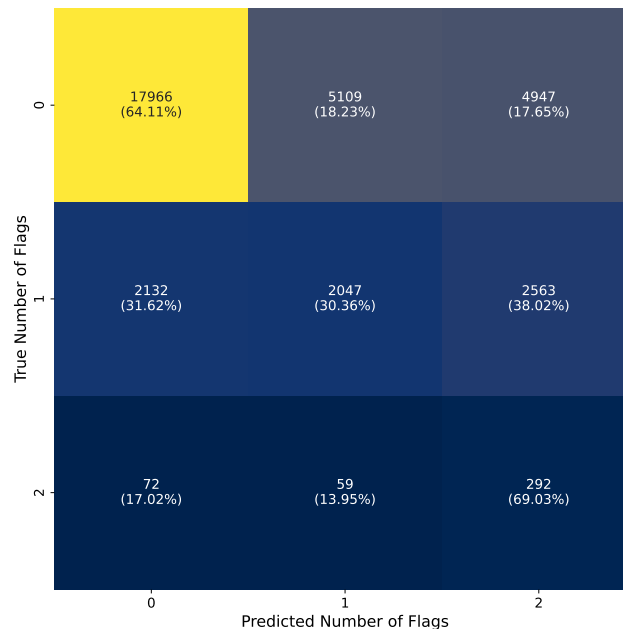


Figure B.3: The confusion matrix with percentages of total observations for the Naive Bayes model on the test set.

The ROC curves and AUC values offer additional insights into the Naive Bayes model's performance in

comparison to the MLP model. With an AUC of 0.73 for class 0, the Naive Bayes model demonstrates a moderate ability to distinguish between legitimate transactions and those with at least one fraud indicator. However, this performance is overshadowed by the MLP model's AUC of 0.92, indicating the MLP model's superior discriminatory power. Similarly, the Naive Bayes model's AUC for class 1 (0.64) falls short of the MLP model's AUC (0.90), highlighting its limited ability to differentiate between transactions with 1 fraud indicator and those with either no indicators or 2 indicators. For class 2, the Naive Bayes model's AUC of 0.81 is lower than the MLP model's exceptional AUC of 0.99, suggesting that while the Naive Bayes model can distinguish between transactions with 2 fraud indicators and those with fewer indicators to some extent, it is not as effective as the MLP model.

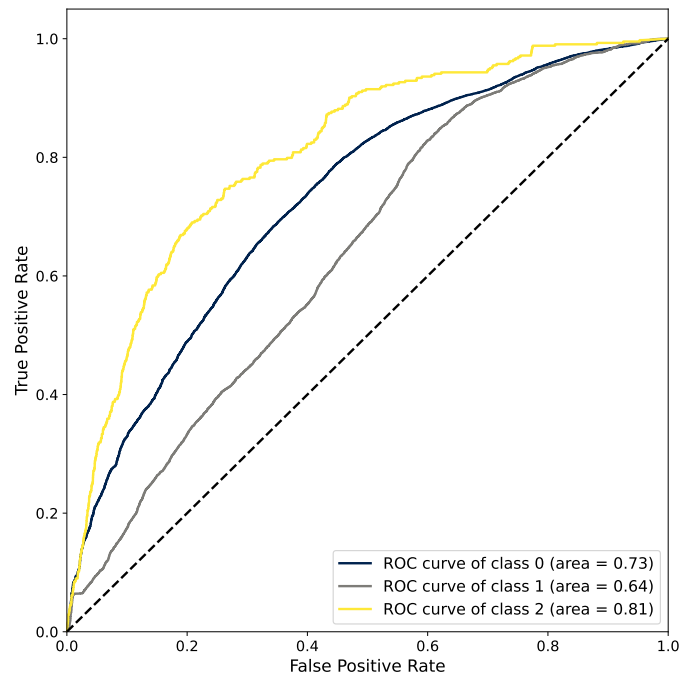


Figure B.4: The ROC curves for the Naive Bayes model for each class on the test set using a one-vs-all approach, where the class of interest is treated as the positive class, and all other classes are combined to form the negative class.

The Naive Bayes model's performance in detecting fraudulent fuel transactions in the Eastern Cape government fleet can be further contextualised by examining its effectiveness in relation to findings from credit card fraud detection studies employing probabilistic methods. While the Naive Bayes model demonstrates a moderate ability to identify transactions with no fraud indicators (class 0) and a notable ability to identify transactions with 2 fraud indicators (class 2), its overall effectiveness falls short of the results reported in some studies.

For instance, Pun (2011) investigated the use of a meta-learning strategy that incorporated a Naive Bayes classifier as one of the base classifiers, showcasing its potential to contribute to improved fraud detection performance when combined with other algorithms. However, the Naive Bayes model's standalone performance in the Eastern Cape government fleet fuel management system does not reach

the same level of effectiveness as the meta-learning approach, indicating that the Naive Bayes classifier alone may not be sufficient to address the intricacies of this specific fraud detection task.

Moreover, the Naive Bayes model's limited performance in accurately identifying transactions with a single fraud indicator (class 1) aligns with the challenges faced by probabilistic models in handling imbalanced datasets and distinguishing between classes with similar characteristics, as observed in studies such as Stolfo et al. (1997) and Sen and Dash (2013). These findings underscore the necessity for more advanced techniques, such as ensemble methods or data resampling, to enhance the performance of Naive Bayes classifiers in fraud detection tasks with imbalanced datasets.

In summary, the Naive Bayes model's performance in detecting fraudulent fuel transactions in the Eastern Cape government fleet is characterised by its moderate ability to identify transactions with no fraud indicators, limited effectiveness in identifying transactions with a single fraud indicator, and a notable ability to identify transactions with 2 fraud indicators, albeit with a high number of false positives. While the model demonstrates some potential in certain aspects of fraud detection, its overall performance is surpassed by the Multi-layer Perceptron model across most classes and evaluation metrics.

B.3 SGD Classifier

The SGD Classifier model's performance in detecting fraudulent fuel transactions in the Eastern Cape government fleet can also be evaluated in comparison to the Multi-layer Perceptron (MLP) model. While both models demonstrate effectiveness in identifying fraudulent activities, there are notable differences in their performance across the different classes of fraud indicators.

When considering the accuracy of identifying transactions with no fraud indicators (class 0), the SGD Classifier model achieves an accuracy of 89.04%, which is lower than the MLP model's impressive accuracy of 95.57%. This suggests that the MLP model is more effective in minimising false positives and maintaining a high true positive rate for legitimate transactions. The precision (0.88), recall (0.89), and F1-score (0.88) for class 0 are also slightly lower for the SGD Classifier model compared to the MLP model (0.90, 0.96, and 0.93, respectively), further highlighting the MLP model's superior performance in this regard.

Focusing on transactions with a single fraud indicator (class 1), the SGD Classifier model's accuracy of 39.29% falls short of the MLP model's accuracy of 54.54%. This indicates that the MLP model is more effective in accurately identifying these suspicious transactions, although both models have room for improvement. The precision (0.49), recall (0.39), and F1-score (0.44) for class 1 are also lower for the SGD Classifier model compared to the MLP model (0.72, 0.55, and 0.62, respectively), emphasising the MLP model's better performance in handling transactions with a single fraud indicator.

The SGD Classifier model's ability to identify transactions with 2 fraud indicators (class 2) is notable, with an accuracy of 52.72%, slightly higher than the MLP model's accuracy of 49.41%. However, the SGD Classifier model's precision (0.17), recall (0.53), and F1-score (0.25) for class 2 are lower than those of the MLP model (0.64, 0.49, and 0.56, respectively). This suggests that while the SGD Classifier model can identify a larger proportion of transactions with 2 fraud indicators, it does so at the cost of a higher number of false positives compared to the MLP model.

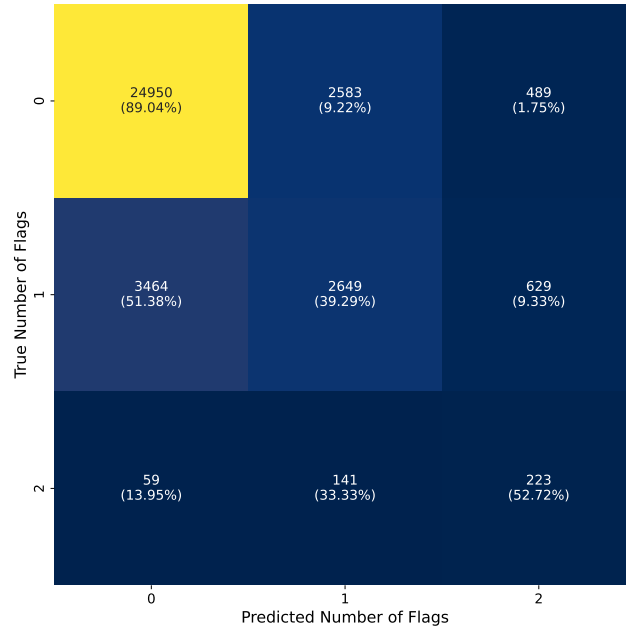


Figure B.5: The confusion matrix with percentages of total observations for the SGD Classifier model on the test set.

The ROC curves and AUC values provide further insights into the performance differences between the two models. The SGD Classifier model's AUC for class 0 (0.80) is lower than that of the MLP model (0.92), indicating its weaker ability to distinguish between legitimate transactions and those with at least one fraud indicator. Similarly, the SGD Classifier model's AUC for class 1 (0.76) is considerably lower than that of the MLP model (0.90), demonstrating its limited ability to differentiate between transactions with 1 fraud indicator and those with either no indicators or 2 indicators. For class 2, the SGD Classifier model's AUC (0.94) is lower than that of the MLP model (0.99), suggesting that while it can distinguish between transactions with 2 fraud indicators and those with fewer indicators effectively, it is not as accurate as the MLP model.

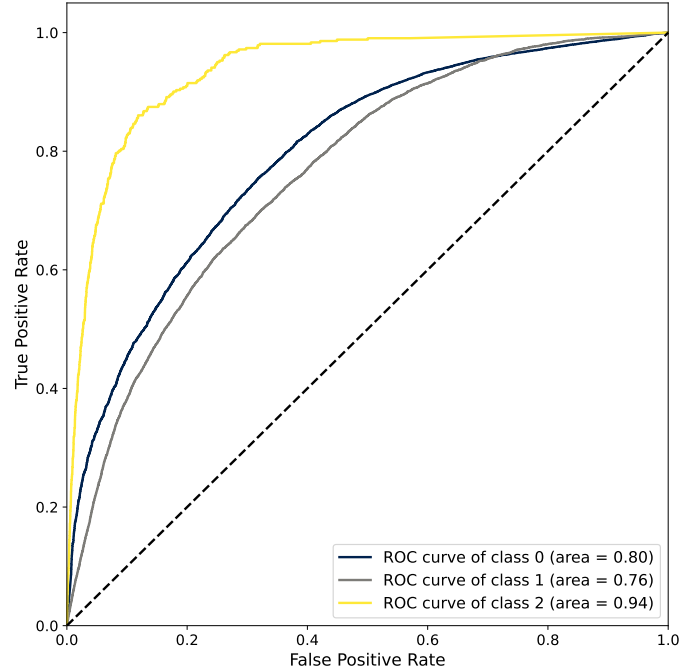


Figure B.6: The ROC curves for SGD Classifier for each class on the test set using a one-vs-all approach, where the class of interest is treated as the positive class, and all other classes are combined to form the negative class.

In conclusion, while the SGD Classifier model demonstrates effectiveness in detecting fraudulent fuel transactions in the Eastern Cape government fleet, particularly for transactions with no fraud indicators and those with 2 fraud indicators, its overall performance is inferior to that of the Multi-layer Perceptron model. The MLP model consistently outperforms the SGD Classifier model in terms of accuracy, precision, recall, and F1-scores across all classes, as well as in its ability to handle the imbalanced dataset. These findings underscore the potential of neural network-based approaches, such as the MLP model, in tackling complex fraud detection challenges and highlight the importance of comparing and evaluating different machine learning techniques to identify the most suitable model for a given fraud detection task.

The comparison of the SGD Classifier model's performance with the findings from credit card fraud detection studies using gradient-based optimisation techniques provides additional context for evaluating its effectiveness in the Eastern Cape government fleet fuel management system. While the SGD Classifier model demonstrates a strong ability to identify transactions with no fraud indicators (class 0) and an excellent ability to distinguish between transactions with 2 fraud indicators (class 2) and those with fewer indicators, its performance is not always directly comparable to the results reported in the literature due to differences in dataset characteristics and evaluation metrics.

Studies such as Pun (2011) and Stolfo et al. (1997) have explored the use of gradient-based optimisation techniques, such as decision tree classifiers and ensemble methods, in credit card fraud detection tasks. While the SGD Classifier model's performance in the Eastern Cape government fleet fuel management system aligns with the general findings of these studies in terms of its effectiveness in identifying

fraudulent activities, the specific performance metrics and dataset characteristics may differ, making a direct comparison challenging.

Moreover, the moderate performance of the SGD Classifier model in identifying transactions with a single fraud indicator (class 1) is consistent with the challenges faced by gradient-based optimisation models in accurately classifying instances from imbalanced datasets, as observed in studies like Sen and Dash (2013). However, the specific techniques employed in these studies, such as data resampling or cost-sensitive learning, may differ from the approach used in the SGD Classifier model, leading to potential variations in performance.

The comparison of the SGD Classifier model's results with those of the Multi-layer Perceptron (MLP) model and the findings from credit card fraud detection studies using gradient-based optimisation techniques highlights the importance of a comprehensive evaluation of different machine learning techniques in various fraud detection contexts. While the SGD Classifier model demonstrates some potential in identifying fraudulent transactions, particularly those with no fraud indicators or 2 fraud indicators, its overall performance is inferior to that of the MLP model in the Eastern Cape government fleet fuel management system. This suggests that the choice of machine learning approach should be based on a thorough analysis of the specific fraud detection task, dataset characteristics, and performance requirements, with neural network-based approaches like the MLP model showing promise in tackling complex fraud detection challenges.