Model performance optimisation in credit card fraud detection using class imbalance techniques, feature engineering and feature selection techniques

School of Computer Science & Applied Mathematics
University of the Witwatersrand

Joseph Assabil
1832028

Main Supervisor: Dr. Ibidun Christiana Obagbuwa
Co-supervisor: Dr J. Kariv

June 1, 2025

Master of Science - Research Report (e-Science)

ORCID #: 0000-0002-2523-5143

Abstract

Fraud detection in financial datasets, particularly in credit card transactions, presents a significant challenge due to the prevalence of irrelevant features and class imbalances. Addressing these issues is crucial for optimizing model performance and accurately identifying fraudulent activities. This research focuses on the application of feature engineering, class imbalance handling techniques alongside a comparative analysis of feature selection techniques such as Chi-Square, ANOVA, (Recursive Feature Elimination) RFE, and (Information Gain) IG all in bid to find the best combination of techniques that enhance model accuracy in (Credit Card Fraud Detection) CCFD. To mitigate class imbalances, (Synthetic Minority Oversampling Technique) SMOTE , (Synthetic Minority Oversampling Technique With Edited Nearest-Neighbours) SMOTE-EEN, and simple oversampling were employed. These methods aimed to balance the class distribution, improving the models' ability to detect fraud. Popular classification models, including Decision Trees, KNN, AdaBoost, and XGBoost, were trained on datasets that had undergone feature engineering, class imbalance techniques and feature selection all in bid to produce optimized model performances. The study utilized evaluation metrics like F1-score, Balanced Accuracy and ROC-AUC to assess model performance and the results demonstrated how feature engineering, combined with specifically SMOTE-EEN as the class imbalance handling technique alongside strategic ANOVA or Chi-Square Test, significantly improved the accuracy and robustness of the fraud detection models with accuracy scores of over 90% across the four classifiers on the four datasets. These findings will thus help provide valuable insights for industry researchers in selecting the most effective techniques for optimizing model performance in fraud detection studies.

Keywords: SMOTE, SMOTE-EEN, RFE, CHI-SQUARE, ANOVA, IG, ROC-AUC, F1-SCORE.

Declaration

I **Joseph Junior Assabil**, hereby declare the contents of this research report to be my own work. This report is submitted for the degree of Master of Science in e-Science at the University of the Witwatersrand. This work has not been submitted to any other university, or for any other degree.

06/06/2025

Acknowledgements

My first and foremost thanks goes to the NEPTTP foundation for their enormous assistance towards funding this research and studies. Secondly, I'd like to thank the Manager of the e-science program, Mrs. Casey Sparkes for her unwavering support and assistance. Lastly, my profound gratitude goes to my supervisors, Dr. C. Ibidun Obagbuwa and Dr. J. Kariv for their supervision through out this project.

Contents

List of Figures

List of Tables

0.1 List of Abbreviations

- **Adaboost** - Adaptive Boosting
- **AI** - Artificial Intelligence
- **AUC** - Area Under Curve
- **AUPRC** - Area Under Precision Recall Curve
- **CCF** - Credit Card Fraud
- **CCFD** - Credit Card Fraud Detection
- **CPF** - Card Present Fraud
- **CNPF** - Card Not Present Fraud
- **EDA** - Exploratory Data Analysis
- **PCA** - Principal Component analysis
- **SMOTEENN** - Sythetic Minority Oversampling Technique With Edited Nearest Neighbours
- **ML** - Machine Learning
- **ROC_AUC-Score** - Receiver Operating Characteristic & Area Under Curve
- **K-NN** - K- Nearest Neighbour
- **Xgboost** - Extreme Gradient Boost
- **Lr** - Logistic Regression
- **DTC** - Decision Tree Classifier
- **SMOTE** - Synthetic Minority Oversampling Technique
- **Unnamed:** - Unique index
- **trans_date:** - Transaction date
- **trans_time:** - Transaction time
- **cc_num:** - Credit Card number
- **merchant:** - Type of Merchant
- **category:** - Category of the transaction
- **amt:** - Amount of Transaction
- **first:** - Individual First name
- **last:** - Individual Surname
- **amt:** - Amount of Transaction
- **unix_time:** - Time lapse for Transaction

- **is_fraud:** - The fraudulent target class denoted by 1 and 0 otherwise.

Chapter 1

Introduction

High-dimensional datasets often have redundant and peripheral features that often hinders the performance of classification models. The issue is particularly prevalent in Credit Card Fraud Detection (CCFD), where errors resulting from such features can lead to overfitting, increased training time, difficulty in interpreting the model's behavior, and costly erroneous results. Credit Card Fraud (CCF), which involves unauthorized transactions made using someone else's credit card information, poses significant financial risks for companies and individuals alike. Companies can suffer substantial financial losses, damage to their reputation, and a decrease in customer trust if fraud is not effectively detected and prevented.

For individuals, CCF can result in financial distress, anxiety, and the burden of disputing fraudulent charges. On a broader scale, rampant fraud can erode public trust in financial systems and digital commerce, hindering economic progress. Feature engineering and feature selection help address these challenges by identifying and creating a smaller set features that are apposite to the target variable. Feature engineering involves the creation of new features or modification of existing ones to better capture the underlying patterns in the data, thus enhancing the machine learning models' performances. By focusing on informative features and engineering new ones, model performance can be significantly improved, enhancing both accuracy and generalizability. Eliminating irrelevant features reduces the likelihood of overfitting, as the model is less prone to learning patterns specific to the training data. Moreover, a smaller, more relevant feature set simplifies model explanation and provides a clearer understanding of how features influence predictions, thereby enhancing model interpretability.

Beyond feature engineering and selection, addressing class imbalances is another critical component in improving a model's performance in CCF detection. Models often struggle due to the class imbalances inherent in CCF datasets. Synthetic Minority Oversampling Technnique (SMOTE), Synthetic Minority Oversampling Technnique With Edited Nearest-Neighbour (SMOTE-ENN) and traditional oversampling techniques will be particularly adopted to address this issue. SMOTE-EEN and SMOTETomek further refine the dataset by combining oversampling with data cleaning methods, removing noisy or borderline instances to create clearer decision boundaries. Despite these efforts, challenges such as the evolving nature of fraud patterns, data privacy concerns, and regulatory compliance remain. Effective feature engineering requires domain expertise to identify variables that can capture subtle indicators of fraudulent behavior, such as transaction frequency, merchant categories, and geolocation patterns. Nonetheless, advancements in feature selection, engineering, and class imbalance techniques have the potential to significantly improve fraud detection systems. These enhancements can lead to substantial

financial savings for businesses and consumers, a more secure financial ecosystem, and greater trust in digital payment methods. On a societal level, robust fraud detection mechanisms can help prevent criminal activities related to financial fraud, contributing to overall economic stability and growth.

As the digital economy continues to expand, the ability to effectively detect and prevent fraud will become increasingly vital. By integrating advanced feature selection, feature engineering, and class imbalance techniques, we can develop more resilient models capable of combating fraudulent activities, thereby protecting financial interests and enhancing the integrity of financial transactions worldwide. This report explores some feature engineering techniques relating to credit card fraud datasets, then looks to select the best class imbalance technique for handling the class imbalance inherent in fraudulent datasets. It also goes on to conduct a comparative analysis on a few lists of feature selection techniques with hopes of finding the optimal technique, which when combined with the best class imbalance technique, would deliver an excellent performance for machine learning models which will detect fraudulent instances within the datasets.

1.1 Problem Statement

Recent advancements Asha and KR [2021]; Dhal and Azad [2022]; Kumar *et al.* [2021] in fraud detection have seen significant research efforts leveraging machine learning algorithms to identify fraudulent activities in financial datasets, particularly in credit card transactions. However, much of the existing literature predominantly focuses on training classification models using selected datasets, often with minimal emphasis on employing comprehensive strategies such as feature selection, feature engineering, and addressing class imbalances. Most studies utilize a single feature selection or class imbalance technique and fail to compare the effectiveness of multiple approaches. Furthermore, these studies rarely explore how the combination of multiple techniques— feature engineering, feature selection, and class balancing—can holistically enhance model performance.

This research aims to address these gaps by conducting a comparative analysis of several widely studied feature selection techniques, like Chi-Square, Analysis of Variance (ANOVA), Recursive Feature Elimination (RFE), and Information Gain (IG), in the context of CCFD. These techniques were chosen for their effectiveness in processing both numerical and categorical variables, particularly for binary classification problems inherent in fraud detection datasets. By engineering and selecting the most relevant features across four fraudulent datasets, this research seeks to reduce irrelevant noise, improve model accuracy, and ultimately enhance fraud detection capabilities.

In addition to feature selection and engineering, this study highlights the explicative role of addressing class imbalances, a common challenge in fraud detection due to the disproportionate ratio of fraudulent to non-fraudulent instances. To this end, the research incorporates and compares the effectiveness of various class imbalance techniques, including SMOTE, SMOTE-EEN, and simple oversampling. These methods aim to balance the class distribution, reducing model bias towards the majority class and improving the detection of minority fraudulent cases.

Thus by combining feature engineering, feature selection, and class imbalance techniques, this study will provide a holistic framework for optimizing machine learning models in CCFD. It will further rank and compare the performance of popular classification algorithms such as Decision Trees, (K-Nearest Neighbors) KNN, (Adaptive Boosting) AdaBoost, and (Extreme Gradient

Boosting) XGBoost on the engineered datasets. The study will symbolize a valuable resource for researchers and practitioners by offering practical guidelines and recommendations for selecting the best techniques under varying conditions.

Ultimately, this work seeks to advance the field of CCFD by providing a nuanced understanding of the integrated impact of feature engineering, feature selection, and class imbalance handling on model performance, contributing to more robust and accurate fraud detection solutions.

1.2 Motivation for the Study

Fraud detection, particularly in financial datasets such as credit card transactions, is a critical area of research due to its impact on global financial systems. Fraudulent activities cost financial institutions billions annually and affect consumer trust in digital payment systems. Despite advancements in machine learning, the persistent challenges of class imbalances, irrelevant features, and the lack of interpretable results have hindered optimal fraud detection outcomes. The motivation for this study stems from the need to address these challenges comprehensively by:

- Investigating the effectiveness of multiple feature engineering and selection techniques to isolate the most relevant variables, reduce noise, and improve model accuracy.
- Addressing the class imbalance issue using advanced techniques such as SMOTE, SMOTE-EEN, and oversampling to improve fraud detection for minority classes.
- Offering a comparative analysis of different methodologies, thereby providing researchers and practitioners with actionable insights into the best approaches for fraud detection.

Specifically, it examines datasets containing transactional records, where fraud cases represent the minority class. The research evaluates feature engineering, selection methods, and class balancing techniques in this context. This study area is highly relevant as:

- Credit card fraud remains one of the most significant threats in the financial sector.
- Datasets in this domain exhibit common challenges such as high dimensionality, class imbalance, and noisy features, making it an ideal testbed for validating the proposed methodologies.
- By exploring these challenges in CCFD, the findings of this research will be generalizable to other fraud detection systems, such as insurance fraud, healthcare fraud, and online retail fraud detection.

The justification of this research is founded by the pressing challenges that persist in the field of fraud detection, particularly in financial datasets such as credit card transactions. Financial fraud remains a global issue, causing substantial economic losses and necessitating the development of advanced solutions to mitigate its impact. Most fraud detection datasets suffer from severe class imbalances, which lead to biased models that overlook minority classes. By focusing on class balancing techniques such as SMOTE and SMOTE-EEN, this study ensures that models effectively identify fraudulent cases. Additionally, the prevalence of irrelevant and noisy features in such datasets often hampers model performance. Through a systematic evaluation of feature engineering and selection techniques, this research emphasizes the importance of retaining only the most relevant features to enhance accuracy. Additionally, the study evaluates the performance of popular classification algorithms, including Decision Trees, KNN, AdaBoost,

and XGBoost, on engineered datasets, contributing to a deeper understanding of their effectiveness in fraud detection tasks. By offering clear recommendations and actionable guidelines, this research equips practitioners with tools to optimize fraud detection models, making it highly relevant for real-world applications. In conclusion, the research addresses critical gaps in the domain of CCFD through the integration of feature engineering, feature selection, and class imbalance techniques into a cohesive framework. By evaluating the effectiveness of multiple methods, the study enhances the accuracy and robustness of fraud detection models. The findings of this research will not only advance academic understanding but also provide practical tools and guidelines for developing effective fraud detection systems. This work, therefore, represents a pivotal step in addressing the challenges of financial fraud detection in machine learning.

1.3 Objectives of the Research

The focus of this research is to explore the interaction between feature engineering techniques, feature selection techniques, class imbalance techniques and model types across four (4) different CCF dataset. This will be achieved through the following sub-objectives:

1. Study the datasets and engineer extra numerical/categorical features to enhance fraud detection and gain general insights into the importance of feature engineering in CCF detection.
2. Understand the overall impact of the three (3) class imbalance techniques and rank their performances.
3. Evaluate and compare the performance of five different feature selection techniques across the four (4) different datasets.

1.4 Research Questions

1. How does the engineering of relevant features help further unravel fraud patterns within a dataset?
2. Which of the class imbalance techniques between SMOTE, SMOTE-EEN and traditional oversampling offers the most optimal performance for ML classifiers?
3. Which of the feature selection techniques between Chi-Squared score, L1 regularisation, Information Gain, ANOVA, RFE scores is the most effective for optimal model performance in CCF detection?

Chapter 2

Literature Review

2.1 Overview of feature selection and feature Engineering

While numerous classification models exist, achieving optimal performance can be hindered by the presence of irrelevant or redundant features. This research will aim to firstly engineer extra features that can help unravel fraud patterns within the dataset and then evaluate the usefulness of different feature selection techniques in improving the performance of the chosen classification models. The publication by [Chandrashekar and Sahin 2014], has confirmed the rising need for feature selection techniques in the new era as load of machine learning and pattern recognition tasks have emerged more in the last decade than the times past. As such, their publication has added its voice to the existing calls for performance augmenting feature selection techniques for more accurate processing of complex datasets given the data era that we find ourselves in. The work of Mamdouh Farghaly and Abd El-Hafeez [2023] reports how widely feature selection has been used in enhancing performances of models in tasks such as genomic analysis, information retrieval, and text categorization. Büyükkeçeci and Okur [2022] in their publication, delved deeper into the concept of feature selection stability, a crucial aspect for ensuring the reliability of selected features across different datasets.

It also explores various metrics used to assess the stability of feature selection classifiers. Other recent studies have rifled deeply through feature selection techniques for classification tasks. Feature selection's impact on model performance and interpretability is a growing area of study, with works like Liu *et al.* [2019] exploring information gain and correlation coefficient combinations, and Mamdouh Farghaly and Abd El-Hafeez [2023] investigating chi-square tests alongside minimum redundancy approaches for optimal feature subset selection. Additionally, studies like Kumar *et al.* [2021] analyze the merits of merging feature selection with machine learning algorithms, while works like Hasan and Bao [2021] explore works on comparing different feature selection methods for cardiovascular diseases detections.

Mustaqim *et al.* [2021] in their work, evaluated the effectiveness of RFE with Cross-Validation toward classifier performances in CCF Detection. In their study, the RFE technique appeared to have been an enhancer for a more delightful performance for the classifiers than the instances where it was not used. The work of [Brownlee 2019], also highlighted the importance of careful feature selection in machine learning, providing a comprehensive overview of methods and practical guidance for implementation when working with selecting relevant features for model fitting. These advancements highlight the continuous development and refinement of feature selection techniques for achieving superior classification model performance.

2.2 Combined approach of feature selection and class imbalance methods

[Rtayli and Enneya 2020] reported that CCF has become a threat with the recent rise of the e-commerce industry, leading to increased research in automatic and real-time fraud detection. [Rtayli and Enneya 2020] went on further to show that recent studies leverage Machine Learning (ML) techniques to develop models capable of identifying fraudulent transactions effectively. Their paper introduces a novel hybrid model for CCF detection that outperforms previous approaches. The model's robustness was obtained by integrating RFE for selecting predictive features, GridSearchCV for optimising the hyper-parameters SMOTE to address the data imbalance. Experimental results obtained from their study demonstrated the model's superior efficiency and effectiveness on multiple real-world datasets. Emphasis on the importance of addressing the class imbalance problem as a prerequisite for developing robust fraud detection systems was presented by [Bekmaganbet 2021]. Hybrid sampling techniques, including oversampling methods, have been proposed as effective preprocessing strategies to balance the dataset before applying classification algorithms.

This approach aimed to enhance the model's ability to pinpoint fraudulent transactions by mitigating the bias towards the majority class. The effectiveness of such preprocessing techniques had been evaluated using various performance metrics in comparison with existing popular algorithms. The findings suggested that with highly imbalanced data, the model's ability to detect fraud improves significantly after application of hybrid sampling techniques. These results underscored the immense need to address data imbalance. [Kumar *et al.* 2021] in their study also focused on comparing different oversampling techniques, including SMOTE, (ADASYN) Adaptive Synthetic Sampling, Borderline-SMOTE, and Safe-Level SMOTE. These techniques were evaluated by applying various classifiers to the problem and analyzing their performance across several metrics.

The comparative analysis provided insights into the effectiveness of each oversampling method in improving classification accuracy, particularly for the minority classes, thereby contributing to the broader effort to mitigate the challenges posed by class imbalances in machine learning. Further emphasis was also given by [Krawczyk 2016] on how class imbalance can severely impact the performance of machine learning models, particularly those that prioritize overall accuracy, as they tend to favor the majority class. To mitigate this, other recent studies have explored various data-level solutions such as resampling techniques. [Nguyen *et al.* 2019] introduced an innovative variant of the SMOTE, known as Borderline-SMOTE, which focuses on generating synthetic instances close to the decision boundary, thus enhancing the classifier's ability to learn from difficult instances. This technique improves classifier generalization, particularly in highly imbalanced datasets. In addition to data-level approaches, cost-sensitive learning methods have also gained momentum in the field. [Buda *et al.* 2018] analyzed deep learning models and their response to class imbalance, recommending cost-sensitive loss functions as an effective method for increasing minority class accuracy. The introduction of higher penalties for misclassifying minority class samples has shown promise in improving recall rates in domains such as healthcare diagnostics, where missing a rare event like disease detection could have critical consequences.

Another trend in recent literature involves ensemble learning methods. [García and Herrera 2020] extended the classic Random Forest algorithm to incorporate class weights and developed new ensemble techniques like Balanced Random Forests (BRF) and EasyEnsemble, both of which have been shown to outperform traditional resampling techniques. By adjusting the

sample size or assigning weights to classes within the ensemble framework, these methods provide a robust solution to class imbalance while preserving the generalization capability of the model. Despite these recent advancements, the challenge of selecting appropriate strategies for handling imbalanced data remains an active area of research. [Wang *et al.* 2022], in their comprehensive review, emphasized the non-linear relationship between class imbalance and classifier performance, urging further exploration of hybrid methods that combine data-level, algorithmic, and cost-sensitive techniques. This current study contributes to this growing body of knowledge by analyzing the convex relationship between class imbalance, feature engineering techniques and classifier accuracy across a range of datasets, providing insights that are crucial for fine-tuning models in real-world scenarios.

Chapter 3

Methodology

In this chapter, we look at the different methods adopted for the purposes of this research diving through the experimental set up as well as where the datasets were acquired from and dataset descriptions the preprocessing techniques involved to clean the datasets. It only looks at the situation specific feature engineering techniques employed for the suitability of this study, delving deeper into the technicalities of the different feature selection techniques used in the study. It glances briefly at the class imbalance techniques used in the study for optimising model performance in CCF detection and rounds up the chapter on model deployments and the chosen evaluation metrics for the study.

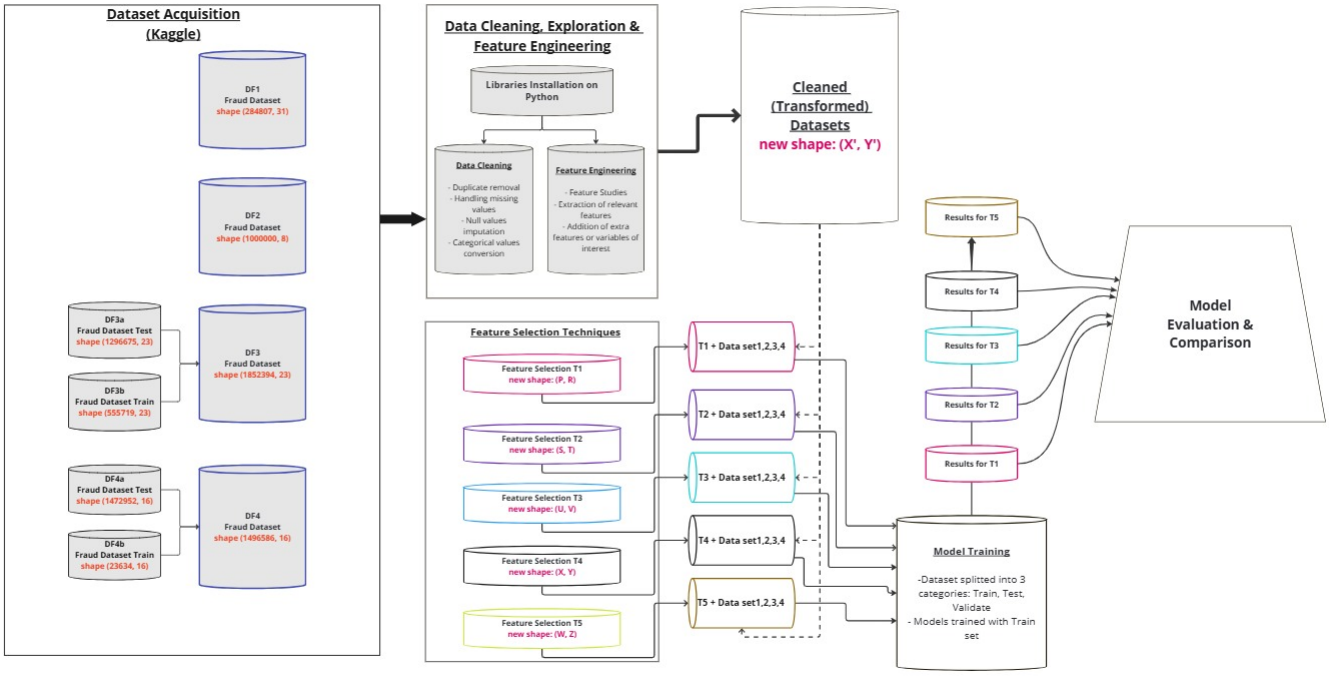

Figure 3.1: Architectural diagram for methodology design

3.1 Experimental set-up

Hardware: The experiments were conducted on a standard personal computer and a cloud-based server equipped with sufficient processing power and memory, with 16GB RAM and a multi-core CPU/GPU.
Software: The software environment included the Python programming language, Jupyter Notebook as well as Google Colab notebook for the code execution and interactive computing, machine learning libraries such as scikit-learn, imbalanced-learn, sklearn.linear_model (for importing logistic regression classifier), sklearn.model_selection (for train, test and splitting dataset), sklearn.ensemble (for importing the Random Forest classifier), sklearn.metrics (for importing the evaluation metrices), sklearn.feature_selection (for importing the feature selection technique), imblearn.over_sampling (for importing Oversampling, SMOTE and SMOTE-EEN), TensorFlow/Keras, and XGBoost as well as data manipulation libraries including pandas and numpy, and visualization tools like matplotlib and seaborn.

3.2 Data Acquisition:

Four (4) datasets of which two (2) are simulated credit card transaction dataset containing legitimate and fraud transactions (see Figure 3.1 for dataset architecture) were utilised for the purpose of this study.

3.2.1 Dataset1 (DF1) info:

This dataset includes credit card transactions carried out by European cardholders in 2023. It contains over 550,000 records, with the data anonymized to safeguard the cardholders' identities. The dataset was collected from credit card transactions made by European cardholders in 2023, with sensitive information removed to ensure privacy and compliance with ethical guidelines.

Key Features:

id, V1-V28, Amount, Class.

3.2.2 Dataset2 (DF2) info:

This is a comprehensive dataset containing close to 1 million transactions, sourced by an unnamed institute to provide crucial information for developing fraud detection models. This dataset particularly is what is called a nearly perfect dataset as the entire dataset contained only numerical entities. The key features in this dataset paint a vivid picture of each transaction, allowing for a deeper understanding of potentially fraudulent behaviors.

Key Features:

distance_from_home, distance_from_last_transaction, ratio_to_median_purchase_price, repeat_retailer, used_chip, used_pin_number, online_order, fraud

3.2.3 Dataset3 (DF3) info:

This dataset is a simulated credit card transaction dataset containing legitimate and fraud transactions from the duration 1st Jan 2019 - 31st Dec 2020. This dataset covers credit cards of 1000

customers doing transactions with a pool of 800 merchants. The dataset was generated using Sparkov Data Generation-Github tool created by Brandon Harris (from Kaggle). The files will be combined and converted into a standard format.

Key Features

Unnamed, trans_date, trans_time, cc_num, merchant, category, amt, first, last, unix_time, is_fraud

3.2.4 Dataset4 (DF4) info:

This is a synthetic dataset containing Fraudulent E-Commerce Transactions. It was designed to simulate transaction data from an e-commerce platform with a focus on fraud detection. It contains several features commonly found in transactional data, with additional attributes specifically engineered to support the development and testing of fraud detection algorithms.

Dataset Overview

Number of Transactions in DF4A:	1,472,952
Number of Transactions in DF4B:	23,634
Number of Features before engineering:	16
Fraudulent Transactions:	Approximately 5%

Key Features:

Transaction ID, Customer ID, Transaction Amount, Transaction Date, Payment Method, Product Category, Quantity, Customer Age, Customer Location, Device Used, IP Address, Shipping Address, Billing Address, Is Fraudulent, Account Age Days, Transaction Hour

3.3 Data Preprocessing:

The preprocessing phase for the fraud detection dataset involved several crucial steps to ensure the data was clean, well-prepared, and suitable for the selected machine learning algorithms. The following techniques were adopted:

3.3.1 Data Cleaning

- *Duplicate Removal:* Luckily across all four datasets, there were no duplicate entries.
- *Handling Missing Values:* All four (4) datasets had no null values.
- *Outlier Detection and Treatment:* Identified and handled outliers that could skew the models' performances. The Z-score analysis was used to detect and handle outliers if any existed.

3.3.2 Feature engineering and transformation

Feature engineering is a vital step in CCFD as it helps uncover hidden patterns and relationships within transaction data. By deriving meaningful features, we can effectively differentiate between legitimate and fraudulent transactions. Each of the four datasets had their own context specific features engineered (kindly refer to *appendix 1* for list of engineered features). The next

sentences give an overview of the categorical systems taken into account for the engineering of the features. To start, **transaction time features** such as the hour of the day, day of the week, and elapsed time since the last transaction were extracted. These temporal patterns provided insights into unusual transaction timings often associated with fraud. Similarly, **transaction amount features**, including flags for large transactions, average transaction amount, and deviations from historical averages, were engineered. Features like percentile-based classifications helped capture variations in transaction value.

Additionally, **frequency-based features** such as the number of transactions within a specified timeframe, repeat merchant flags, and transaction frequency ratios provided useful behavioral insights. Moving on to behavioral features, patterns in cardholder spending, such as preferred transaction amounts, merchant categories, and anomaly scores, were analyzed. **Location-based features** such as distances between consecutive transaction locations, country-based flags for unusual regions, and time-location consistency checks were also engineered to detect spatial inconsistencies.

Aggregated features played a key role in capturing broader trends. **Rolling statistics**, such as averages and standard deviations over various time windows, were calculated. **Peer comparisons** benchmarked individual transactions against similar users or transactions at the same merchant. For categorical features, merchant and transaction type variables were encoded using frequency or target encoding, while high-risk categories and flags for incomplete metadata added further value. Finally, **anomaly detection features** derived from unsupervised models like autoencoders provided scores indicating unusual transactions.

The objective of these engineered features was to enhance the dataset's information richness, enabling the model to capture behavioral, temporal, and spatial anomalies typically seen in fraudulent transactions. As a result, the predictive model achieved significant improvements in accuracy, recall, and precision, effectively identifying fraudulent activities while minimizing false positives.

3.4 Feature Selection

A select group of feature selection techniques like Chi-Square test, ANOVA, RFE, IG and Lasso was compared and the suitable technique interms of offering optimal performances to the models used in the study was selected.

3.4.1 Chi-Squared Test

A statistical technique used to assess whether there is a significant relationship between variables . In feature selection, it evaluates the independence of each feature from the target variable Pudjihartono *et al.* [2022]; Chandrashekar and Sahin [2014]; Liu and Setiono [2022]. This method has been selected for this study due to its simplicity and effectiveness in identifying features with a strong association with the target variable. It is particularly well-suited for categorical data, making it an ideal choice for pinpointing critical factors in fraud detection. Chi-Squared statistic is calculated as:

$$\chi^2 = \sum_{i=1}^{n} \frac{(O_i - E_i)^2}{E_i} \tag{3.1}$$

- O_i - observed frequency
- E_i expected frequency of the feature value (see equation 3.1).
High Chi-Squared values correspond to more relevant features. This test is widely used for selecting relevant features in fraud detection, particularly when dealing with categorical features. It helps identify features that have a statistically significant association with fraudulent transactions Liu and Setiono [2022]. The only advantage to using the Chi Square test is, its simple to interpret, suitable for categorical data and the disadvantage is that it does not measure the strength of the relationship, which renders its suitability for continuous features practically useless, and sensitive to imbalanced datasets He *et al.* [2020].

3.4.2 Information Gain (IG)

IG evaluates reduction in entropy when a dataset is split based on a feature. It evaluates how much insight into the target variable is gained by knowing the value of the feature, making it a perfect fit for this study. Information Gain is selected for its ability to handle both categorical and continuous features. It helps in identifying features that provide the most information about the target variable, which is pertinent for refining the accuracy of fraud detection models Omuya *et al.* [2021]; Tangirala [2020]. The entropy H and Information Gain IG are defined as:

$$H(S) = -\sum_{i=1}^{n} p_i \log_2(p_i) \tag{3.2}$$

$$IG(S, A) = H(S) - \sum_{v \in \text{Values}(A)} \frac{|S_v|}{|S|} H(S_v) \tag{3.3}$$

p_i represents the probability of class i, S_v is the subset of S where feature A takes on the value v ((see equations 3.2 and 3.3).

IG is a popular technique for selecting relevant features in fraud detection tasks. Studies have shown its effectiveness in identifying informative features from transaction data, including transaction amount, location, time, and cardholder details Tangirala [2020]. IG offers easy implementation, offers computationally efficiency, and provides a clear ranking of features based on their informativeness. Its drawback lies in its sensitivity to imbalanced datasets, a frequent issue in fraud detection, which can hinder its ability in capturing suffiscated non-linear relationships between features and the target variable Omuya *et al.* [2021]; Tangirala [2020].

3.4.3 ANOVA (Analysis of Variance)

ANOVA is a statistical technique used to compare the means of multiple groups to determine if they are significantly different. In feature selection, it evaluates the variance between the means of different classes for each feature (see equation 3.4 for formula). ANOVA is chosen for its effectiveness in handling continuous features. It helps in identifying features that have significant differences between classes, making it useful for improving model performance in fraud detection. The F-statistic is calculated as:

$$F = \frac{\text{Variance Between Groups}}{\text{Variance Within Groups}} \tag{3.4}$$

Higher F-values indicate that the feature significantly contributes to the target variable.

3.4.4 L1 Regularization (Lasso)

L1 Regularization, also known as Lasso (Least Absolute Shrinkage and Selection Operator), adds a penalty equal to the absolute value of the magnitude of coefficients to the loss function Omuya *et al.* [2021]; Chandrashekar and Sahin [2014]. L1 Regularization is selected for its ability to handle multicollinearity and perform automatic feature selection Dhal and Azad [2022]. It is particularly useful in high-dimensional datasets, common in fraud detection, where it helps in reducing overfitting and improving model interpretability. This can be expressed as:

$$L(\beta) = \sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij})^2 + \lambda \sum_{j=1}^{p}|\beta_j| \tag{3.5}$$

where λ (see equation 3.5) is the regularization parameter. Lasso can shrink some coefficients to zero, effectively performing feature selection.

L1 regularization is a powerful technique for feature selection in fraud detection tasks involving logistic regression models. It not only selects relevant features but also helps prevent overfitting by reducing model complexity. It is able to perform feature selection and reduces overfitting simultaneously, thus providing interpretability by identifying features with non-zero coefficients. Unfortunately, the Lasso regression may not be as effective as other methods for highly correlated features, which means it requires careful tuning of the regularization parameter.

3.4.5 RFE (Recursive Feature Elimination)

RFE is a feature selection technique that recursively removes the least important features based on the model's performance. It involves the following steps:

1. Training a model on the current set of features.
2. Ranking features based on their importance.
3. Removing the least important feature.
4. Then repeating until the desired number of features is reached.

RFE has been chosen for its ability to select a subset of features that contribute the most to model performance. It is an iterative and model-agnostic approach, making it suitable for the choice of data in this study for fraud detection. RFE has also shown promising results in fraud detection tasks, particularly when used with SVM models. It leverages the strengths of SVMs in handling high-dimensional data and selects features directly relevant to the SVM's classification ability. RFE Works well with SVMs, performs feature selection based on model performance. It is also very computationally expensive compared to filter methods, and it offers limited interpretability of feature selection.

3.5 Handling Class Imbalance

Class imbalances occur when one class in a dataset significantly outnumbers the other(s), leading to biased model predictions. In fraud detection for example, the "fraudulent" class is usually

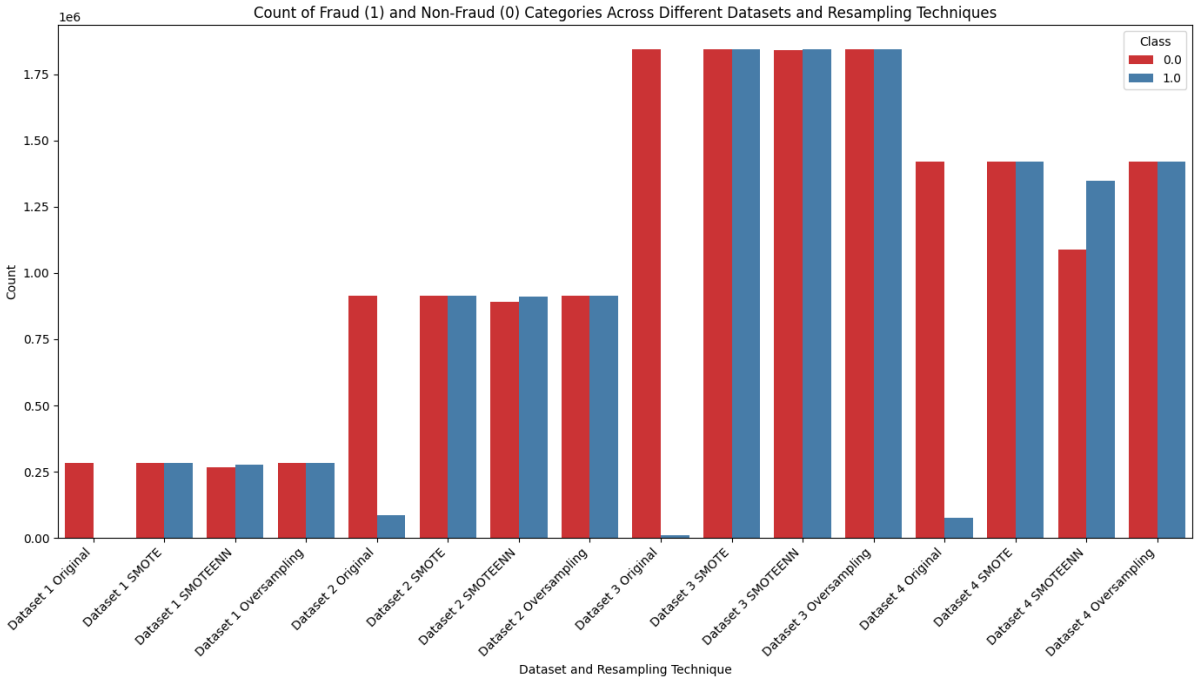

Figure 3.2: Dataset analysis with class imbalance techniques

much smaller than the ”non-fraudulent” class. Various techniques like oversampling, undersampling, SMOTE and SMOTE-EEN have been developed in this study to tackle the problem of class imbalance.

3.5.1 SMOTE

A popular method for handling class imbalance in datasets. Introduced by Chawla *et al.* [2021], SMOTE aims to create synthetic examples of the minority class to balance the class distribution without merely duplicating existing samples, as is the case in traditional oversampling. SMOTE is widely adopted due to its effectiveness and simplicity, especially in scenarios where the dataset has a large imbalance between majority and minority classes. Creates artificial samples for the underrepresented class by interpolating between current instances of that class. SMOTE identifies the k closest neighbors within the minority class for each instance and generates new examples by blending them Mamdouh Farghaly and Abd El-Hafeez [2023]. New synthetic instances are then created through the selection of a neighbor and generation of new instances along the line connecting the original instances and the neighbors. SMOTE reduces overfitting compared to simple random oversampling and it also generates diverse synthetic instances, which can help improve model generalization. The only disadvantage associated with the SMOTE technique is that it can introduce noise if the minority class instances are noisy or if clusters of different classes are close to each other.

This technique is a non-linear technique as it does not assume any relationship between the features. Instead, it generates synthetic samples by interpolating between existing samples of the minority class. For each minority class sample, SMOTE selects a few of its nearest neighbors (typically $k = 5$). It then randomly selects one of these nearest neighbors and generates a synthetic point along the line connecting the original sample to the neighbor. Mathematically,

this synthetic point is generated as:

$$\text{Synthetic Sample} = \text{Minority Sample} + \lambda \times (\text{Neighbor Sample} - \text{Minority Sample}) \quad (3.6)$$

Where $\lambda \in [0, 1]$.λ is a random number that controls the interpolation between the minority sample and its neighbor. This ensures that the new synthetic sample lies somewhere between the original sample and its neighbor. This process is repeated until the desired number of synthetic samples is generated Bekmaganbet [2021]. Even though SMOTE is considered non-linear, the synthetic samples are created by interpolating between the original samples in feature space without assuming any linearity or distribution for the data, it leverages the distances between the points to create new synthetic examples. Since this interpolation occurs along the distance (which is not constrained to linear relationships), it can capture non-linear boundaries between classes, which often exist in real-world datasets as seen in fraudulent transactions.

The SMOTE method, as illustrated in Figure 3.2 and Table 3.1, effectively balances the class distribution by generating synthetic examples for the minority class (fraud cases). This is particularly evident in Datasets 1, 2, 3, and 4, where the application of SMOTE increases the number of fraud cases to match the non-fraud ones, resulting in an almost equal distribution of both classes. The rationale behind SMOTE involves creating synthetic samples along the line segments joining each minority class instance and its k-nearest neighbors, which aids in expanding the decision boundary of the minority class, reducing the risk of overfitting that often arises with simple replication methods. This synthetic sampling approach has been demonstrated to enhance model performance and mitigate class imbalance, as emphasized by Chawla *et al.* [2021], who underscore its utility in improving predictive models' recall rates. In practice, SMOTE generates these new instances in the feature space, rather than duplicating actual cases, thereby introducing variability and making the synthetic samples less likely to be direct copies. This characteristic of SMOTE ensures that the classifiers trained on the balanced dataset are exposed to a broader distribution of the minority class, ultimately enhancing the model's ability to generalize.

3.5.2 SMOTE-EEN

It is a hybrid method that combines SMOTE with Edited Nearest Neighbors (ENN), a data cleaning technique. ENN removes samples that are misclassified by their nearest neighbors, effectively cleaning both the majority and minority classes after SMOTE has generated synthetic samples [Muntasir Nishat *et al.* 2022]. It often enhances the quality of the dataset by reducing noise thus leading to better decision boundaries by removing instances that could confuse the classifier. It also tends to be computationally more expensive than SMOTE alone and the removal of data points might lead to a loss of important instances. SMOTE first generates synthetic minority class samples, as explained earlier, and then ENN is applied to clean the dataset. ENN works by examining each instance in the dataset (both synthetic and real) and removing any sample whose class label differs from the majority of its nearest neighbors. Typically, ENN uses 3-nearest neighbors to evaluate whether a sample should be removed. The goal of ENN is to reduce noise by removing borderline and noisy instances that could confuse the classifier. By combining SMOTE and ENN, SMOTE-EEN effectively addresses two key problems with SMOTE: class overlap and noise. SMOTE's synthetic data generation can sometimes lead to minority class samples overlapping with the majority class, but ENN helps to eliminate these overlapping instances by removing ambiguous samples. This leads to a cleaner, more separable dataset. Additionally, the removal of noisy data through ENN can improve the quality of the dataset, resulting in better model performance Dhal and Azad [2022]. However, SMOTE-EEN

is computationally more expensive than SMOTE or traditional oversampling, as it involves both generating synthetic data and performing an additional cleaning step. The ENN step may also remove some valuable minority class instances, which can reduce the diversity of the minority class data if not carefully tuned. Despite these challenges, SMOTE-EEN is often preferred in practice because it balances oversampling and undersampling, leading to more robust classifiers [Saheed *et al.* 2022]. The SMOTE-EEN technique, which represents a hybrid approach that combines the oversampling power of SMOTE with the noise-reduction capabilities of the Edited Nearest Neighbor (ENN) algorithm, is also highlighted in the chart. The SMOTE component functions by generating synthetic samples for the minority class, which enhances class balance and helps mitigate the problem of a skewed class distribution. Specifically, SMOTE synthesizes new data points in the feature space by interpolating between existing minority class samples and their k-nearest neighbors. This method broadens the decision boundaries of the minority class, reducing the classifier's bias towards the majority class and enhancing the ability to detect minority class instances in future predictions. However, while SMOTE is effective in addressing class imbalance, it does not inherently handle the potential noise or overlap in class distributions, which can lead to less reliable models when dealing with real-world, noisy datasets.

To address this limitation, SMOTE-EEN integrates the ENN algorithm, which further refines the dataset. ENN works by removing data points from both classes that are misclassified when tested against their k-nearest neighbors. This step is crucial for cleaning the synthetic dataset and eliminating instances that may introduce ambiguity or complexity in the classification process. By removing overlapping or noisy samples, SMOTE-EEN enhances the separation between classes, leading to a cleaner and more well-defined boundary. This refinement process is particularly valuable in scenarios where the data contains substantial overlap or outliers, as it enhances model robustness and reduces the likelihood of overfitting. According to Fernández *et al.* [2020], the combination of SMOTE and ENN has been shown to significantly improve classifier performance on imbalanced datasets, especially in cases where simple oversampling or standalone SMOTE is insufficient.

The visual representation in Figure 3.2 confirms the impact of this method, showing that while SMOTE-EEN achieves class balance similar to SMOTE, the total count of samples is reduced. This reduction is a direct consequence of ENN's noise removal step, which enhances data quality but decreases the overall number of samples. Such data pruning ensures that the model training is focused on more representative and less confusing instances, improving the classifier's ability to generalize well to unseen data. The use of SMOTE-EEN has been validated in various studies, demonstrating its effectiveness in scenarios characterized by a high degree of class overlap or the presence of noise, as further supported by More and Rana [2021]. Thus, SMOTE-EEN emerges as a powerful and comprehensive method for addressing class imbalance while simultaneously enhancing data quality.

3.5.3 Oversampling

Traditional oversampling is the simplest method for handling class imbalance. This method involves replicating existing minority class samples to balance the dataset. There are no new data points generated; instead, existing minority class instances are sampled repeatedly, sometimes with replacement. This method is easy to implement and computationally inexpensive since it merely duplicates data points rather than generating new ones. However, the main drawback of traditional oversampling is the increased risk of overfitting. Since the classifier encounters the same instances multiple times, it may become overly specialized to the training data, limiting

its ability to generalize to unseen data [Nguyen *et al.* 2019; Wang *et al.* 2022]. Furthermore, this technique does not introduce any new information about the minority class, and thus, if the original minority class data is noisy or sparse, traditional oversampling cannot rectify these issues. While it can ensure that the class distribution is balanced, traditional oversampling often does little to improve the classifier's performance in practice, especially when dealing with non-linear decision boundaries or highly imbalanced datasets. The oversampling technique, which straightforwardly balances the dataset by replicating minority class instances, is also depicted in figure 3.2. This method is evident in each of the datasets presented, where the counts for the fraud class are increased to match those of the non-fraud class, resulting in a perfectly balanced distribution. The key advantage of oversampling lies in its simplicity and ease of implementation; it requires minimal computational overhead, making it highly practical for rapid prototyping or when computational resources are restricted. By duplicating existing minority class instances, oversampling ensures that every class receives equal representation during the training phase, potentially enhancing the performance of classifiers that otherwise struggle with imbalanced data. Nevertheless, this technique introduces several significant drawbacks. One of the primary concerns is overfitting, as repeatedly sampling the same minority class data points may lead the model to memorize these instances rather than learning generalized patterns. This risk of overfitting is particularly pronounced when the duplicated samples dominate the training set, resulting in a model that performs well on the training data but fails to generalize effectively to unseen instances.

Moreover, oversampling does not contribute any new information to the dataset, as it merely replicates existing data points without adding variability. This lack of synthetic diversity can be problematic in real-world applications where capturing the underlying distribution of the minority class is crucial for accurate predictions. The duplicated samples do not enhance the feature space and may even amplify any noise or anomalies present in the original minority class data. As pointed out by Han *et al.* [2020], this method does not have mechanisms to manage data noise or address overlapping regions between classes, which can further hinder the model's performance. figure 3.2 reflects these characteristics, illustrating how oversampling equalizes class distributions but fails to mitigate noise or improve the dataset's quality. Despite these limitations, oversampling remains a valuable technique in cases where more sophisticated resampling methods, such as SMOTE or SMOTE-EEN, are not feasible. It provides an accessible and computationally efficient way to improve model training by addressing class imbalance, albeit with a trade-off in terms of potential overfitting and lack of diversity in the training data. Additionally, oversampling may be preferable in scenarios where preserving the exact characteristics of the original data is necessary, or when working with datasets that have a relatively small minority class size. Its utility in imbalanced learning applications has been explored extensively in the literature, as it serves as a baseline against which more advanced methods are compared.

3.6 Model Deployment & Evaluation

3.6.1 (Extreme Gradient Boosting) XGBoost

XGBoost is a gradient boosting algorithm that builds an ensemble of decision trees sequentially. Each new tree focuses on correcting the errors made by the previous trees. The key steps involve ideally involve **Initialization,** which starts with an initial model $f_0(x)$. The model then adds new trees $f_m(x)$ by minimizing the objective function, which combines a loss function L and a

Table 3.1: Comparison of Class Imbalance Techniques: Traditional Oversampling, SMOTE, and SMOTE-EEN

Criterion	**Traditional Oversampling**	**SMOTE**	**SMOTE-EEN**
Methodology	Replicates existing minority class instances	Generates synthetic minority samples by interpolating between instances	Combines SMOTE's synthetic sample generation with ENN's noise filtering
Technicality	Linear, duplicates real samples, increasing overfitting risks	Non-linear, creates new points in feature space by calculating vector difference and scaling with a random factor λ	Hybrid approach, balances between overfitting and noise by reducing noisy samples after applying SMOTE
Advantages	Simple to implement, no change in data space topology	More robust as it introduces synthetic variability, reducing overfitting	Handles both imbalance and noise, yielding cleaner, more balanced data
Disadvantages	High risk of overfitting, does not add variability to the data	May introduce noisy samples as synthetic points can be unrealistic	More computationally intensive due to the dual process of oversampling and cleaning

regularization term Ω this phase is usually called the **Additive Learning** phase.

$$\min_{f_m} \sum_{i=1}^{n} L(y_i, \hat{y}_i^{(m-1)} + f_m(x_i)) + \Omega(f_m) \tag{3.7}$$

where $\hat{y}_i^{(m-1)}$ is the prediction of the ensemble model after $m-1$ trees (see equation 3.7). Another phase in the process known as **Gradient Descent** is later introduced into the system to update the model. XGBoost was chosen for its high performance and efficiency in handling large datasets and its ability to improve iteratively on the errors of previous models, making it highly effective in detecting complex patterns in fraudulent transactions.

3.6.2 AdaBoost

AdaBoost is an ensemble method that combines multiple weak learners (typically decision stumps) into a strong classifier. It works by assigning weights to each training instance, focusing more on the misclassified instances in subsequent rounds Bagga *et al.* [2020]. The steps are:

1. **Initialization**: Where equal weights are assigned to all training instances.
2. **Training**: In this stage, a weak learner is trained on the weighted dataset.
3. **Weight Update**: The weights of the misclassified instances are then increased and the weights of the rightly classified instances are then decreased.

4. **Combination**: Here, all the weak learners are combined using a weighted majority vote.

The final model then becomes:

$$F(x) = \sum_{m=1}^{M} \alpha_m h_m(x) \tag{3.8}$$

where α_m is the weight assigned to the m-th weak learner $h_m(x)$ (see equation 3.8).

AdaBoost has been chosen for its ability to improve the classification accuracy of difficult cases. By focusing on misclassified instances, it enhances the model's ability to identify challenging fraud cases.

3.6.3 K-Nearest Neighbors (K-NN)

A simple, instance-based learning algorithm. It classifies a data point based on the labels of its k nearest neighbors in the feature space (see equation 3.9). The key steps involved in the functionality of this classifier are:

1. **Distance Calculation**: The distance between the new data point and all points in the training set using a distance metric is computed.

$$d(x, x_i) = \sqrt{\sum_{j=1}^{n} (x_j - x_{ij})^2} \tag{3.9}$$

2. **Neighbor Identification**: Identification of the k closest points.
3. **Majority Vote**: Then assigns the class label based on the majority class among the k neighbors.

The K-NN classifier has been chosen for this study due to its simplicity and effectiveness in capturing local patterns in the data. As in fraud detection, it can identify transactions similar to known fraudulent cases, thus providing a straightforward method for classification.

3.6.4 Decision Trees

Decision Trees work by recursively partitioning the data into subsets based on the value of input features, creating a tree-like model of decisions Fayyomi *et al.* [2021]. The primary computation in this process involves calculating a metric called Gain, which helps determine the best feature to split the data at each node. The most commonly used metric for this is Information Gain, which is based on the concept of entropy. Entropy H is defined in equation 3.10 as:

$$H(S) = -\sum_{i=1}^{n} p_i \log_2(p_i) \tag{3.10}$$

where p_i is the proportion of instances belonging to class i in the dataset S. Information Gain IG for a feature A is calculated in equation 3.11 as follows:

$$IG(S, A) = H(S) - \sum_{v \in \text{Values}(A)} \frac{|S_v|}{|S|} H(S_v) \tag{3.11}$$

Here, S_v is the subset of S where feature A has value v.

Each internal node in the tree represents a decision point, and each leaf node represents a final prediction Husejinovic [2020]. The tree grows until it reaches a stopping criterion, such as a maximum depth or a minimum number of samples per leaf. Pruning techniques are often applied to reduce overfitting Hammed and Soyemi [2020]. The Decision Tree classifier will be used in this study mainly due to its ability to provide simplistic and interpretable results. It provides clear rules for decision-making, which is thus essential in our understanding of the factors contributing to fraud. Additionally, this classifier can handle both numerical and categorical data effectively which makes it flexible enough for the scope of this project as additional features will be engineered which has categorical data type.

3.7 Model Evaluation Metrics

3.7.1 AUC-ROC - Score

The ROC curve demonstrates the trade-off between the True Positive Rate (TPR) and the False Positive Rate (FPR) across different threshold values. TPR, also known as recall or sensitivity, calculates the ratio of correctly predicted positive instances relative to all actual positive instances, while FPR indicates the proportion of incorrectly predicted positive instances relative to all actual negative instances Singh *et al.* [2022]; TAGHIYEV *et al.* [2021]. The ROC-AUC score quantifies the area under the ROC curve, ranging from 0 to 1. Values closer to 1, reflects better model performance.

3.7.2 Balanced Accuracy:

Evaluates classification models, especially in situations where classes are imbalanced just as seen with fraud detection datasets. It works with the average of the recall (sensitivity (see equations 3.12 and 3.13)), calculated independently for each class. Balanced accuracy thus adjusts for the imbalance by giving equal weight to the sensitivity of each class (see equation 3.14), providing a more fair assessment of model performance across all classes Wong and Yeh [2019]; Wang *et al.* [2021].

$$\text{Sensitivity}_{class1} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \tag{3.12}$$

$$\text{Sensitivity}_{class2} = \frac{\text{True Negatives}}{\text{True Negatives} + \text{False Positives}} \tag{3.13}$$

$$\textbf{Balanced Accuracy} = \frac{\text{Sensitivity}_1 + \text{Sensitivity}_2}{2} \tag{3.14}$$

We have a perfect classification if the balanced accuracy of the two classes = 1 and a poor classification if the balanced accuracy of the two classes < 0.5. If the balanced accuracy for the two classes is = 0.5, model's performance is as good as random guessing.

3.7.3 F1-Score

This metric combines precision (see equation 3.15) and recall (see equation 3.16), offering a single value to reflect a model's overall performance in detecting positive cases while reducing

both false positives and false negatives. It is highly valuable when dealing with imbalanced class distributions. See equation 3.17 for F1-score Calculation.

$$Precision = \frac{(TruePositive)}{(TruePositive) + (FalsePositive)} \tag{3.15}$$

$$Recall = \frac{(TruePositive)}{(FalseNegative) + (TruePositive)} \tag{3.16}$$

$$F1 - score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{3.17}$$

In the subsequent chapter, we present a detailed discussion of the results obtained in this study, alongside the extraction of key analytical insights typically obscured within CCF datasets. Furthermore, we propose informed recommendations aimed at effectively mitigating the identified issues.

Chapter 4

Results and Discussion

In this chapter, we present and discuss the results obtained in the study and bring to light any specific patterns the study unveiled in the context of CCF detection ranging from the channels through which CCF is carried out throught to the various relationships between categories and the instances of fraud and possible recommendation systems for preventing fraud where possible. The chapter also gives an in depth discussion for the basis of the selection for the class imbalance technique utilised for the study and eventually rounds up on the best possible combinations of model enhancing techniques that hold the potential for optimizing model performances in CCF contexts.

4.1 Analysis into Fraud Instances Across the Datasets

4.1.1 Fraud and Product Category

It is important to highlight that among the three basic needs of life—shelter, food, and protection—shelter appears to be the most implicated in fraud-related activities. A significant portion of fraudulent instances, particularly in dataset df2, reveals that the average spend on clothing is notably high. This aligns with the fact that clothing is a recurring necessity, as humans constantly seek to replenish their wardrobe. Additionally, the rise in technological advancements has paralleled an increase in fraudulent transactions involving electronics. This trend, coupled with notable fraud in health and beauty products (see Figure 4.1 and Figure 4.2), underscores the shift toward non-essential, lifestyle-driven purchases. Interestingly, most product categories linked to fraud are far removed from basic human necessities. Instead, they reflect discretionary spending habits that, while desirable, are not critical for survival.

4.1.2 Payment Mode and Fraud

Both Figure 4.5 and Figure 4.6 show that transactions with credit cards exhibit the widest distribution of fraud and include many high-value transactions, suggesting that credit cards are frequently used for both small and large purchases in both fraud and non-fraud instances hence the possible reason for high incidents in this particular domain. However the total transaction amounts recorded for the fraudulent instances supersedes that of the non-fraudulent. Showing the need for alertness from individuals as well as the relevant institutions especially in this mode of payment. The distribution of transaction amounts using PayPal appeared to be relatively low compared to other payment methods in the fraudulent samples. This could indicate that PayPal is commonly used for smaller transactions even in the non-fraudulent instances. Transactions via bank transfer and debit card display similar trends showing decreased distribution compared to the credit card mode in the fraudulent samples. A note worthy fact from

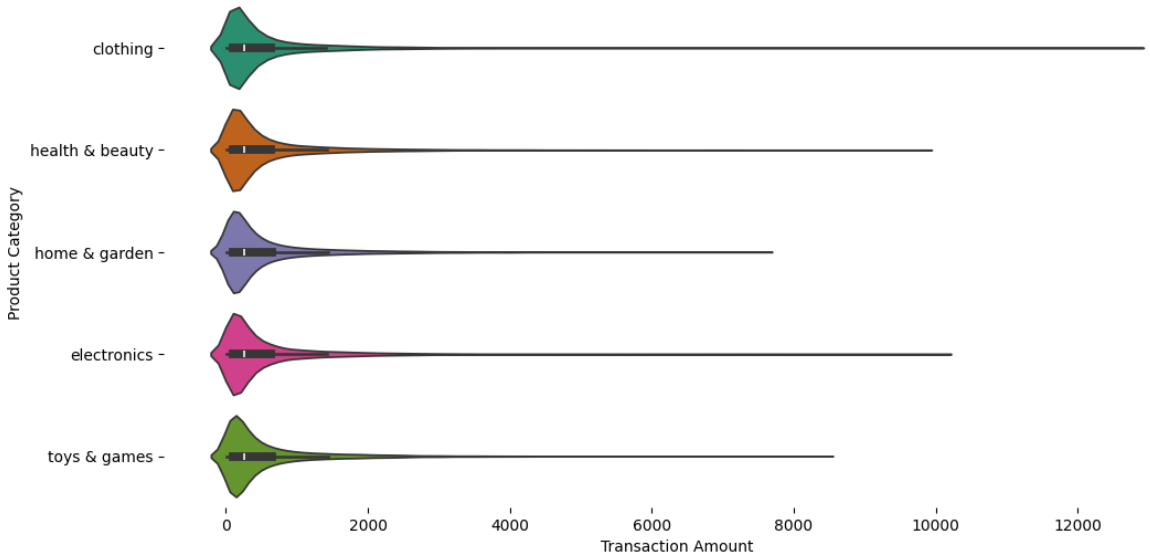

Figure 4.1: Average amount spent per Product Category on Fraudulent Instances

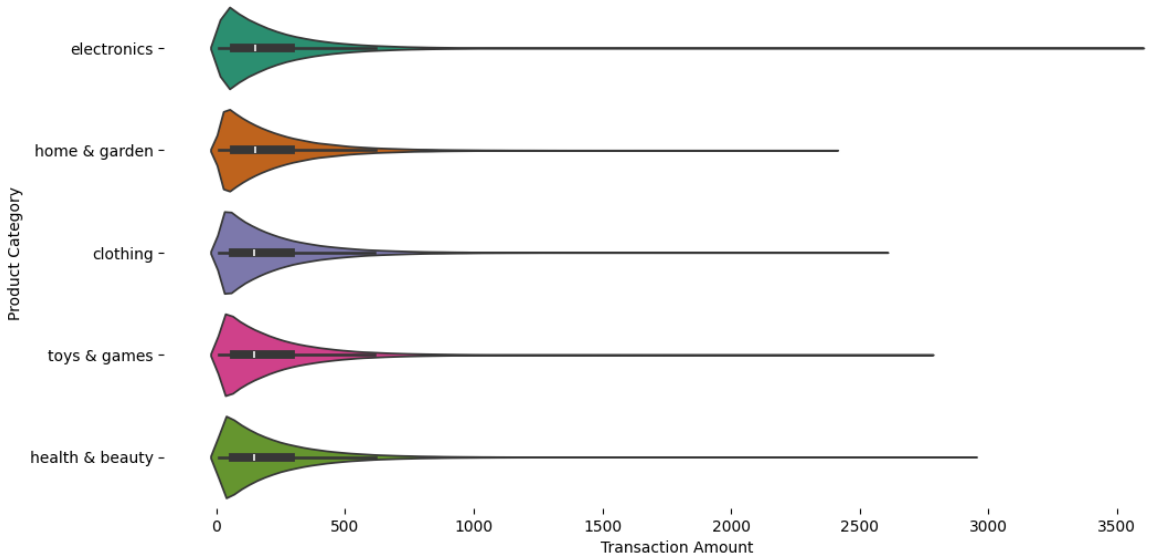

Figure 4.2: Average amount spent per Product Category on Non-Fraudulent Instances

this analysis nonetheless is the fact that criminals have become adept at exploring all modes of payment to achieve their goal as this is seen in higher transaction amounts recorded for the fraudulent samples in Figure 4.5 and Figure 4.7 (when weekday to weekend spending analysis is conducted) than in the non-fraudulent instances Figure 4.6. Overall, this is a cry for vigilance for all individuals to be cautious in their trading activities especially online.

4.1.3 Fraud instances and their channels

From Figure 4.3, we see that smaller gadgets are actually a more susceptible channel for fraudulent behaviours. The average transaction amount carried out on mobile phones is over $12000 which is more than 4 times the amount in the non-fraudulent instances as seen in Figure 4.4. This shows that mobile cellphones are an adequate means utilized by fraudsters in carrying out fraudulent transactions. This could also be related to the easiness associated with acquiring a cellphone than in acquiring other devices like the tablet or the desktop.

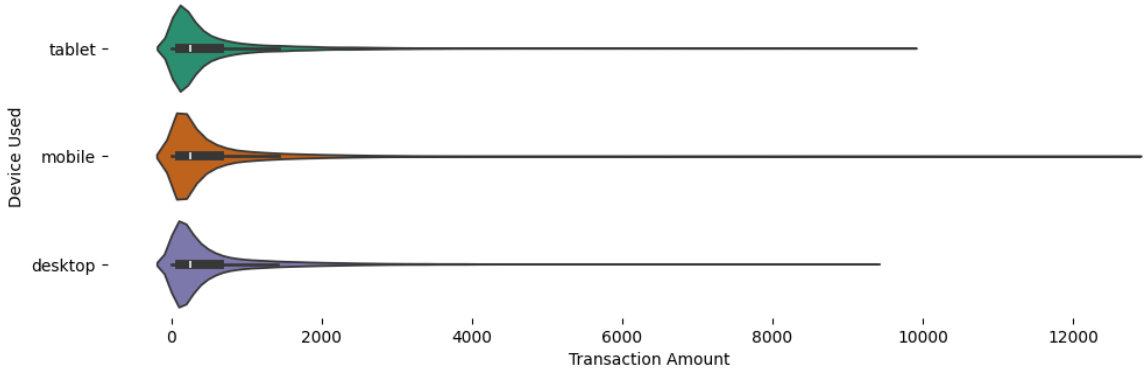

Figure 4.3: Fraud avenues devices - fraud

From Figure 4.8, one can build interesting scenarios for credit card fraud prevention strategies. This figure presents fraudulent instances based on the mode of transaction that is, either In-store, PIN Utilisation, Chip Used. When this figure is carefully analysed, we see high prevalence of fraudulent instances when a transaction is online, when the transaction is conducted without a PIN, and a Chip/card is used. Fraud instances are also very high anytime a PIN is not used. On the positive side, there was zero occurrence of fraud instances whenever a pin was used and the transaction was In-Store. That implies that the traditional route of transacting still holds as a more reliable way of transacting in our present age. As there is almost zero to at most 10 instances of fraudulent instances happening whenever a PIN is used to transact. These analysis could offer insights to understanding fraudulent behaviours and possibly working on a preventative means towards adjudicating against fraud in the financial industry.

4.2 Choice for class imbalance handling

The choice of SMOTE-EEN as the preferred method for addressing class imbalance in CCFD is driven by its unique ability to combine the strengths of both oversampling and undersampling techniques, thereby optimizing model performance in highly imbalanced datasets. SMOTE-EEN, effectively balances the data by generating synthetic samples for the minority class while also removing noisy and overlapping instances from the majority class. This dual approach addresses two significant challenges associated with class imbalance: underrepresentation of

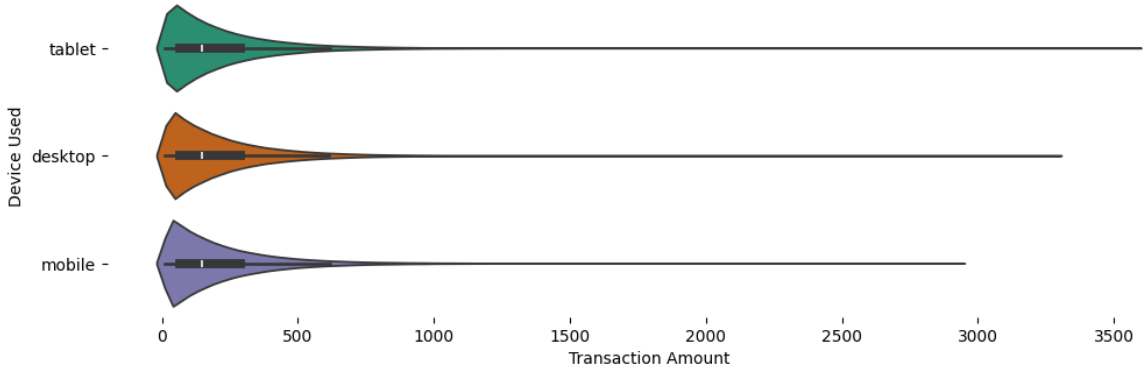

Figure 4.4: Fraud avenues devices - nonfraud

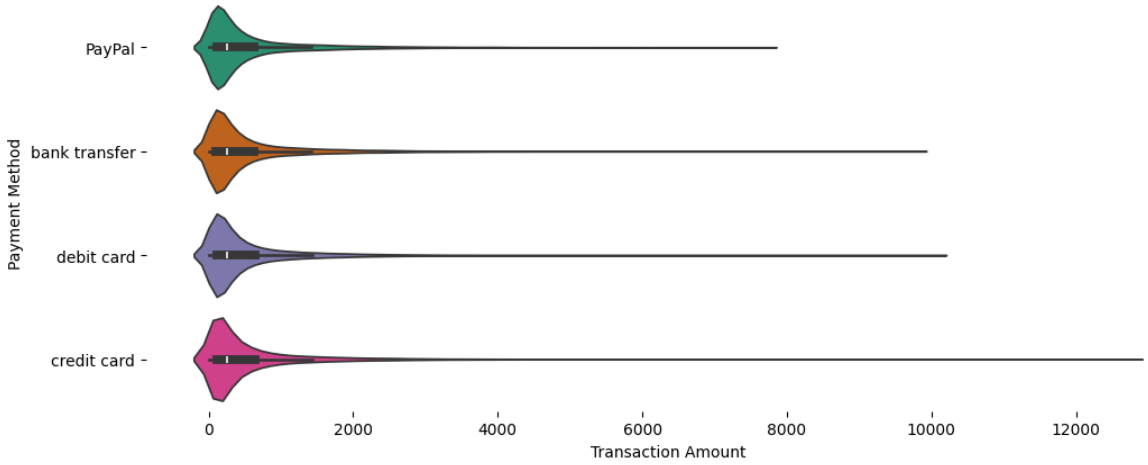

Figure 4.5: Payment Mode's Susceptibility to Fraud

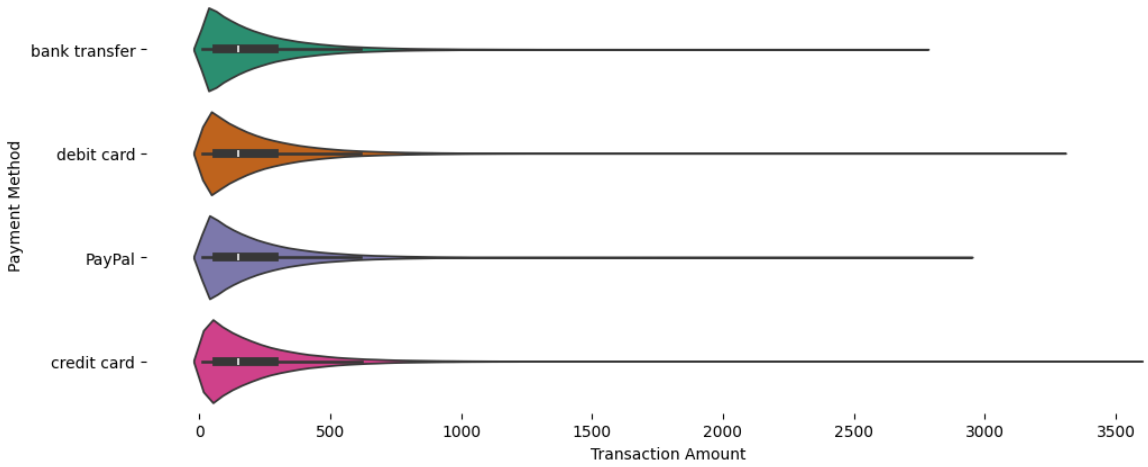

Figure 4.6: Payment Mode Non-Fraud

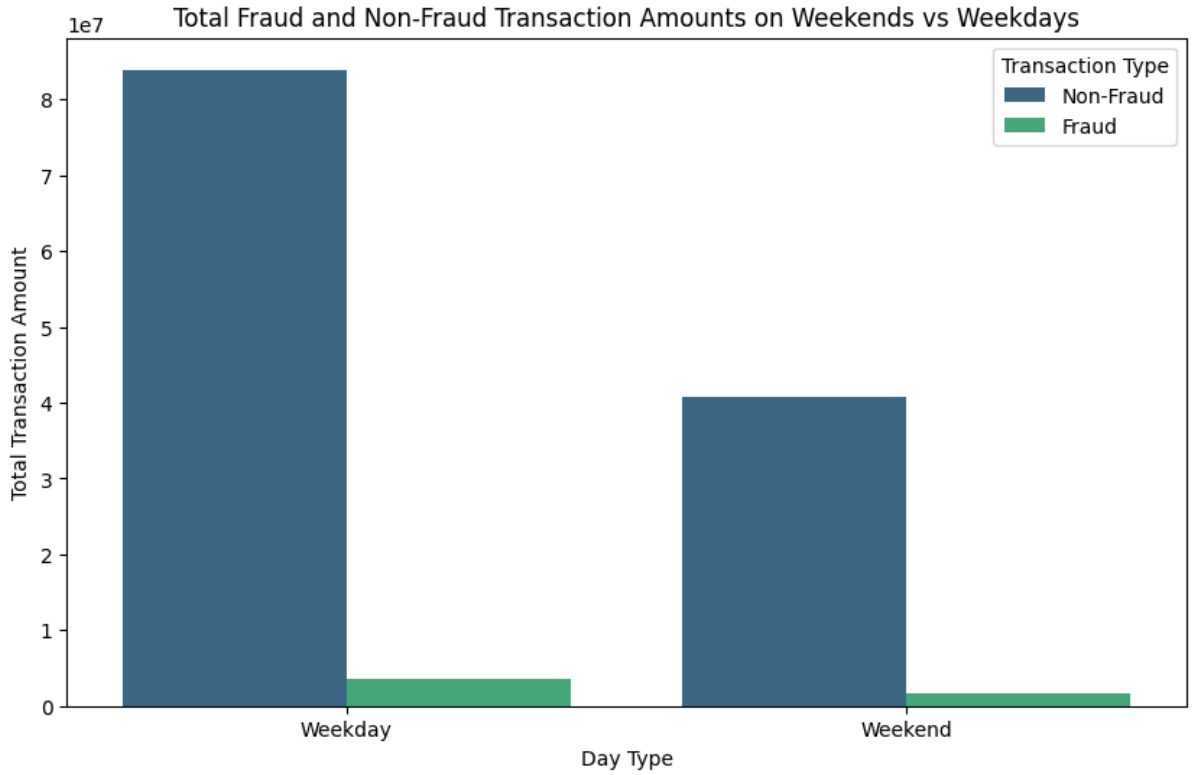

Figure 4.7: Fraud instances Over Weekends and Weekdays

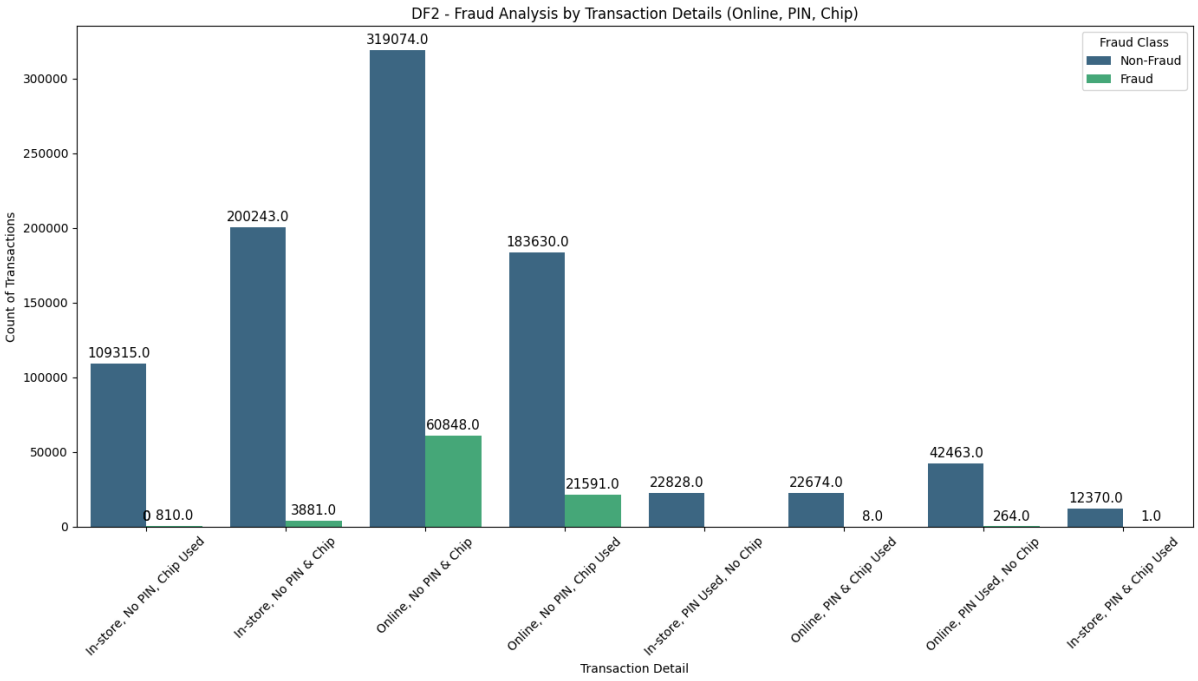

Figure 4.8: Fraud instances and their channels

Table 4.1: Comparison of Evaluation Metrics On Original Dataset and the Different Class Imbalance Techniques on (df1)

Split	Model	Technique	Balanced Accuracy	F1 Score	ROC AUC
Train	Decision Tree	Original	1.00	1.00	1.00
		SMOTE	1.00	1.00	1.00
		Oversampling	1.00	1.00	1.00
		SMOTE-EEN	1.00	1.00	1.00
	KNN	Original	0.58	0.57	0.58
		SMOTE	0.70	0.74	0.71
		Oversampling	0.71	0.74	0.73
		SMOTE-EEN	0.72	0.75	0.73
	AdaBoost	Original	1.00	1.00	1.00
		SMOTE	1.00	1.00	1.00
		Oversampling	1.00	1.00	1.00
		SMOTE-EEN	1.00	1.00	1.00
	XGBoost	Original	1.00	1.00	1.00
		SMOTE	1.00	1.00	1.00
		Oversampling	1.00	1.00	1.00
		SMOTE-EEN	1.00	1.00	1.00
Test	Decision Tree	Original	0.92	0.84	0.83
		SMOTE	0.91	0.87	0.85
		Oversampling	0.93	0.89	0.88
		SMOTE-EEN	0.91	0.88	0.86
	KNN	Original	0.58	0.59	0.58
		SMOTE	0.69	0.63	0.70
		Oversampling	0.70	0.65	0.71
		SMOTE-EEN	0.72	0.60	0.73
	AdaBoost	Original	0.83	0.85	0.98
		SMOTE	0.85	0.87	0.97
		Oversampling	0.85	0.88	0.97
		SMOTE-EEN	0.86	0.88	0.97
	XGBoost	Original	0.98	0.94	0.97
		SMOTE	0.97	0.93	0.96
		Oversampling	0.96	0.91	0.95
		SMOTE-EEN	0.89	0.77	0.87
Val	Decision Tree	Original	0.89	0.85	0.85
		SMOTE	0.90	0.88	0.86
		Oversampling	0.91	0.90	0.89
		SMOTE-EEN	0.89	0.91	0.87
	KNN	Original	0.60	0.59	0.59
		SMOTE	0.68	0.62	0.69
		Oversampling	0.69	0.64	0.70
		SMOTE-EEN	0.68	0.61	0.71
	AdaBoost	Original	0.83	0.85	0.98
		SMOTE	0.84	0.87	0.97
		Oversampling	0.85	0.89	0.97
		SMOTE-EEN	0.86	0.91	0.98
	XGBoost	Original	0.98	0.94	0.97
		SMOTE	0.97	0.92	0.95
		Oversampling	0.95	0.90	0.94
		SMOTE-EEN	0.89	0.77	0.87

the minority class and the presence of mislabeled or outlier data points that could degrade model performance. In the context of CCF detection, where fraudulent transactions are typically rare and often surrounded by complex data distributions, SMOTE-EEN enhances the classifier's ability to generalize to new, unseen data, thereby reducing the likelihood of both false negatives and false positives.

Table 4.2: Comparison of Different Class Imbalance Techniques on (df2) Using Evaluation Metrics

Split	Model	Technique	Balanced Accuracy	F1 Score	ROC AUC
Train	Decision Tree	Original	1.00	1.00	1.00
		SMOTE	1.00	1.00	1.00
		Oversampling	1.00	1.00	1.00
		SMOTE-EEN	1.00	1.00	1.00
	KNN	Original	0.97	0.97	1.00
		SMOTE	0.99	0.99	1.00
		Oversampling	0.99	0.99	1.00
		SMOTE-EEN	1.00	1.00	1.00
	AdaBoost	Original	1.00	1.00	1.00
		SMOTE	1.00	1.00	1.00
		Oversampling	1.00	1.00	1.00
		SMOTE-EEN	1.00	1.00	1.00
	XGBoost	Original	1.00	1.00	1.00
		SMOTE	1.00	1.00	1.00
		Oversampling	1.00	1.00	1.00
		SMOTE-EEN	1.00	1.00	1.00
Test	Decision Tree	Original	1.00	1.00	1.00
		SMOTE	1.00	1.00	1.00
		Oversampling	1.00	1.00	1.00
		SMOTE-EEN	1.00	1.00	1.00
	KNN	Original	0.95	0.95	0.99
		SMOTE	0.98	0.93	0.99
		Oversampling	0.98	0.93	0.99
		SMOTE-EEN	0.98	0.91	0.99
	AdaBoost	Original	1.00	1.00	1.00
		SMOTE	1.00	1.00	1.00
		Oversampling	1.00	1.00	1.00
		SMOTE-EEN	1.00	0.99	1.00
	XGBoost	Original	0.88	0.92	0.98
		SMOTE	0.90	0.91	0.99
		Oversampling	0.89	0.92	0.98
		SMOTE-EEN	0.89	0.88	0.98
Val	Decision Tree	Original	1.00	1.00	1.00
		SMOTE	1.00	1.00	1.00
		Oversampling	1.00	1.00	1.00
		SMOTE-EEN	1.00	1.00	1.00
	KNN	Original	0.95	0.95	0.99
		SMOTE	0.98	0.93	0.99
		Oversampling	0.98	0.93	0.99
		SMOTE-EEN	0.98	0.91	0.99
	AdaBoost	Original	1.00	1.00	1.00
		SMOTE	1.00	1.00	1.00
		Oversampling	1.00	1.00	1.00
		SMOTE-EEN	1.00	0.99	1.00
	XGBoost	Original	0.89	0.93	0.97
		SMOTE	0.92	0.93	0.98
		Oversampling	0.92	0.94	0.97
		SMOTE-EEN	0.90	0.91	0.98

The rationale for using SMOTE-EEN over simpler methods such as basic oversampling or SMOTE alone is grounded in the complexity of fraud detection scenarios. Credit card fraud data often contain overlapping regions between fraudulent and non-fraudulent transactions, as well as noise that can mislead traditional machine learning models. By applying SMOTE to create synthetic examples that capture the minority class distribution, SMOTE-EEN ensures that the

Table 4.3: Comparison of Different Class Imbalance Techniques on (df3) Using Evaluation Metrics

Split	Model	Technique	Balanced Accuracy	F1 Score	ROC AUC
Train	Decision Tree	Original	1.00	1.00	1.00
		SMOTE	1.00	1.00	1.00
		Oversampling	1.00	1.00	1.00
		SMOTE-EEN	1.00	1.00	1.00
	KNN	Original	0.97	0.95	1.00
		SMOTE	1.00	1.00	1.00
		Oversampling	1.00	1.00	1.00
		SMOTE-EEN	1.00	1.00	1.00
	AdaBoost	Original	0.53	0.55	0.96
		SMOTE	0.89	0.89	0.95
		Oversampling	0.89	0.89	0.96
		SMOTE-EEN	0.89	0.89	0.95
	XGBoost	Original	0.75	0.83	0.99
		SMOTE	0.96	0.96	0.99
		Oversampling	0.98	0.98	1.00
		SMOTE-EEN	0.96	0.96	0.99
Test	Decision Tree	Original	0.71	0.70	0.71
		SMOTE	0.85	0.65	0.85
		Oversampling	0.69	0.70	0.69
		SMOTE-EEN	0.85	0.65	0.85
	KNN	Original	0.94	0.91	1.00
		SMOTE	0.98	0.91	1.00
		Oversampling	0.99	0.91	1.00
		SMOTE-EEN	0.99	0.88	0.99
	AdaBoost	Original	0.53	0.55	0.96
		SMOTE	0.86	0.57	0.93
		Oversampling	0.89	0.51	0.96
		SMOTE-EEN	0.86	0.57	0.93
	XGBoost	Original	0.71	0.78	0.99
		SMOTE	0.90	0.61	0.96
		Oversampling	0.92	0.60	0.98
		SMOTE-EEN	0.90	0.61	0.96
Val	Decision Tree	Original	0.72	0.71	0.72
		SMOTE	0.86	0.65	0.86
		Oversampling	0.70	0.70	0.70
		SMOTE-EEN	0.86	0.65	0.86
	KNN	Original	0.94	0.91	1.00
		SMOTE	0.99	0.91	1.00
		Oversampling	0.99	0.92	1.00
		SMOTE-EEN	0.99	0.88	0.99
	AdaBoost	Original	0.53	0.55	0.96
		SMOTE	0.85	0.57	0.92
		Oversampling	0.88	0.51	0.96
		SMOTE-EEN	0.85	0.57	0.92
	XGBoost	Original	0.71	0.78	0.99
		SMOTE	0.91	0.61	0.96
		Oversampling	0.92	0.60	0.98
		SMOTE-EEN	0.90	0.61	0.96

model has sufficient exposure to diverse fraudulent patterns. However, it also takes the critical extra step of applying Edited Nearest Neighbors, which cleans the majority class by removing samples that are likely to confuse the model. This cleaning process is particularly valuable in CCF detection, where the quality of the data is paramount to achieving high performance. As noted by Fernández *et al.* [2020], the combination of oversampling and cleaning strategies can

Table 4.4: Comparison of Different Class Imbalance Techniques on (df4) Using Evaluation Metrics

Split	Model	Technique	Balanced Accuracy	F1 Score	ROC AUC
Train	Decision Tree	Original	1.00	1.00	1.00
		SMOTE	1.00	1.00	1.00
		Oversampling	1.00	1.00	1.00
		SMOTE-EEN	1.00	1.00	1.00
	KNN	Original	0.62	0.68	0.96
		SMOTE	0.93	0.93	1.00
		Oversampling	0.96	0.96	1.00
		SMOTE-EEN	0.99	0.99	1.00
	AdaBoost	Original	0.57	0.62	0.82
		SMOTE	0.77	0.77	0.85
		Oversampling	0.74	0.74	0.82
		SMOTE-EEN	0.81	0.81	0.89
	XGBoost	Original	0.58	0.62	0.84
		SMOTE	0.78	0.78	0.86
		Oversampling	0.76	0.78	0.86
		SMOTE-EEN	0.82	0.82	0.90
Test	Decision Tree	Original	0.60	0.60	0.60
		SMOTE	0.63	0.56	0.63
		Oversampling	0.60	0.60	0.60
		SMOTE-EEN	0.68	0.56	0.68
	KNN	Original	0.59	0.63	0.69
		SMOTE	0.66	0.55	0.70
		Oversampling	0.65	0.58	0.69
		SMOTE-EEN	0.68	0.53	0.72
	AdaBoost	Original	0.58	0.62	0.82
		SMOTE	0.73	0.55	0.79
		Oversampling	0.75	0.56	0.82
		SMOTE-EEN	0.74	0.52	0.81
	XGBoost	Original	0.58	0.62	0.82
		SMOTE	0.73	0.56	0.79
		Oversampling	0.74	0.56	0.79
		SMOTE-EEN	0.74	0.54	0.80
Val	Decision Tree	Original	0.60	0.59	0.60
		SMOTE	0.63	0.55	0.63
		Oversampling	0.59	0.60	0.59
		SMOTE-EEN	0.67	0.55	0.67
	KNN	Original	0.58	0.62	0.69
		SMOTE	0.66	0.55	0.70
		Oversampling	0.65	0.58	0.68
		SMOTE-EEN	0.68	0.52	0.71
	AdaBoost	Original	0.57	0.61	0.82
		SMOTE	0.72	0.55	0.79
		Oversampling	0.75	0.56	0.82
		SMOTE-EEN	0.73	0.52	0.80
	XGBoost	Original	0.57	0.61	0.81
		SMOTE	0.72	0.56	0.78
		Oversampling	0.74	0.56	0.78
		SMOTE-EEN	0.73	0.54	0.79

significantly boost model robustness and accuracy, especially in domains characterized by noisy or imbalanced datasets.

Additionally, SMOTE-EEN has been empirically validated to improve key evaluation metrics such as precision, recall, and the F1-score, which are crucial in fraud detection tasks where the cost of misclassification is high. The method's ability to balance sensitivity and specificity

makes it highly suitable for optimizing the performance of ML algorithms like Random Forests, SVM, and Gradient Boosting Machines, which are commonly used in fraud detection frameworks. More and Rana [2021] have highlighted that hybrid methods like SMOTE-EEN not only enhance model performance but also provide a more stable and interpretable classification outcome. Given the goal of maximizing model performance in CCF detection, the choice of SMOTE-EEN is compelling because it provides a comprehensive approach to handling class imbalance, effectively mitigating the risk of overfitting and enhancing the model's ability to detect fraudulent transactions with high reliability.

In conclusion, SMOTE-EEN offers a balanced and sophisticated solution that aligns well with the objectives of this study, which aims to optimize model performance in CCFD using ML algorithms. By addressing both the class imbalance and noise issues inherent in CCF datasets, SMOTE-EEN provides a robust framework for achieving high predictive accuracy while minimizing costly misclassifications. This makes it a strategic choice, grounded in both theoretical rationale and empirical evidence by tables 4.1 to 4.4, to enhance the effectiveness of ML models in fraud detection applications.

4.3 Selected Features for Various Datasets Across Feature Selection Techniques

Table 4.5: List of Selected Features for Various Datasets Across Techniques

Dataset	Lasso	Chi-Square	ANOVA	Information Gain (IG)	RFE
Dataset 1	V4, V5, V7, V8, V10, V12, V14, V16, V17	Amount, V3, V14, V17, V12, V10, V7, V1	V10, V12, V14, V16, V17, Amount	V10, V11, V12, V14, V17	V4, V10, V13, V14, V16
Dataset 2	distance_from_home, distance_from_last_transaction, ratio_to_median_purchase_price, repeat_retailer, used_chip, used_pin_number, online_order, transaction_density	distance_from_home, distance_from_last_transaction, ratio_to_median_purchase_price, repeat_retailer, used_chip, used_pin_number, online_order, transaction_density	distance_from_home, distance_from_last_transaction, ratio_to_median_purchase_price, repeat_retailer, used_chip, used_pin_number, online_order, transaction_density	ratio_to_median_purchase_price, repeat_retailer, used_chip, used_pin_number, online_order	distance_from_home, distance_from_last_transaction, ratio_to_median_purchase_price, used_pin_number, online_order
Dataset 3	amt, cc_num, unix_time, lat_diff, merch_lat, euclidean_distance, distance_to_kansas, trans_time_in_secs, day_of_week	amt, cc_num, unix_time, euclidean_distance, trans_time_in_secs, city_pop, haversine_distance_km, distance_to_kansas	amt, unix_time, distance_to_kansas, trans_time_in_secs, day_of_week	amt, cc_num, unix_time, trans_time_in_secs, day_of_week	amt, unix_time, euclidean_distance, haversine_distance_km, distance_to_kansas
Dataset 4	Transaction Amount, Quantity, Customer Age, Account Age Days, Transaction Hour, trans_time_in_secs, account_maturity, ratio_accn_day_to_amt	Transaction Amount, Quantity, Customer Age, Account Age Days, Transaction Hour, trans_time_in_secs, account_maturity, ratio_accn_day_to_amt	Transaction Amount, Account Age Days, Transaction Hour, account_maturity, ratio_accn_day_to_amt	Transaction Amount, Quantity, Account Age Days, day_of_week, ratio_accn_day_to_amt	Transaction Amount, Quantity, Account Age Days, day_of_week, ratio_accn_day_to_amt

4.4 Feature Selection Comparison

This section compares the performances of the different chosen feature selection methods with the SMOTE-EEN technique to decipher the best combinations for delivering optimised performances for the classifiers; K-NN, Decision Tree, Xgboost and Adaboost.
From tables 4.6 to 4.9, (kindly refer to Appendix 2 for the graph of the model performances) we analyse the performances of the different models under various feature selection techniques applied to datasets (df1, df2, df3, df4) using the SMOTE-EEN method for handling class imbalance. The analysis was organized by model performance on the training, testing, and validation sets. Across all the datasets, most models exhibit perfect or near-perfect metrics (Balanced Accuracy, F1 Score, ROC AUC) on the training set, irrespective of the feature selection technique. The Decision Tree, KNN, and XGBoost classifiers consistently achieved 100% scores across all metrics with techniques like ANOVA, Chi-Square, IG, Lasso, and RFE. Whiles AdaBoost occasionally demonstrated slight variability. For instance, in df1, the IG method resulted in lower training performance compared to other techniques, indicating sensitivity to feature selection. The training results suggested potential overfitting, particularly with the simpler models like Decision Trees and KNN, as their performance is unnaturally flawless across all feature selection methods.

Model performance discrepancies emerged during testing which offered insights into the generalizability of the models and the effectiveness of feature selection techniques. The Decision Tree classifier performed better with **RFE** and **ANOVA** across the datasets. In df1 for instance, RFE achieved a Balanced Accuracy of 0.91 compared to Lasso's 0.75. Lasso frequently underperformed, highlighting its limitations in capturing relevant features for Decision Trees. The KNN classifier also consistently showed high performance with **Chi-Square** and **ANOVA**, achieving balanced accuracies above 0.90 in most cases across the datasets. Lasso, however, led to a significant drop in performance once again, especially in df1, with a Balanced Accuracy of 0.64. On the other hand, the AdaBoost classifier exhibited strong performance with **ANOVA** and **Chi-Square** but fluctuated when using IG (e.g., df1 Balanced Accuracy: 0.92) and Lasso (df1 Balanced Accuracy: 0.79) (see Appendix 2 for list of figures for the model performances). This indicated AdaBoost's dependency on feature selection techniques that prioritize interpretability and variance. The XGBoost algorithm demonstrated its robustness across the different feature selection methods, with Chi-Square often outperforming others in terms of Balanced Accuracy and F1 Score. Even Lasso showed respectable performance, though slightly lagging behind RFE and ANOVA. For AdaBoost, robust performance is observed across all splits, even on the original dataset, with F1 and ROC AUC nearing perfection. SMOTE-EEN provides marginal gains in balanced accuracy (from 0.83 to 0.86) while preserving excellent test and validation ROC AUC (0.98), indicating it can fine-tune already strong classifiers. Similarly, XGBoost stands out with near-perfect scores on the original dataset, achieving 0.98 on test and validation F1 and ROC AUC. However, applying SMOTE-EEN slightly reduces its performance (e.g., ROC AUC drops to 0.87 on test and validation sets), suggesting that SMOTE-EEN may introduce noise or artifacts in an already optimal model.

The validation results largely corroborated testing trends where the Chi-Square and ANOVA techniques consistently emerged as top-performing techniques across all models, datasets, and metrics. RFE was a close contender, especially for Decision Tree and XGBoost models, offering comparable metrics to ANOVA and Chi-Square. Lasso however continued to underperform in most cases, likely due to its strong regularization bias, which may discard features critical to class separability in imbalanced datasets.

4.4.1 Choice of Feature Selection

From the results recorded in the tables (see tables 4.6 to 4.9) and the discussions raised, the best feature selection techniques under the SMOTE-EEN class imbalance handling method in CCF detection were **Chi-Square** and **ANOVA**, with RFE as a strong alternative. Chi-Square excelled relatively due to its simplicity and focus on categorical and discrete variable associations which was crucial in imbalanced datasets like CCF where class separability is paramount. Whereas ANOVA ensured high performance by prioritizing features with statistically significant variance between classes, RFE balanced interpretability and predictive power, especially in tree-based models as seen in the likes of Decision Tree and XGBoost. Lasso on the contrary, while valuable for linear models, underperformed in the CCF context due to its aggressive feature reduction ability.

4.5 Comparing results of this study to similar works in literature

4.5.1 SMOTE-EEN in other CCFD studies versus this study

Recent studies by Damanik and Liu [2025]; Isangediok and Gajamannage [2022] have emphasized the effectiveness of hybrid sampling techniques like SMOTE-EEN in addressing class imbalance in CCFD. This challenge was also reported as critical as fraudulent transactions constituted only a small fraction of the dataset, often leading to biased models favoring the majority class. In the case of Damanik and Liu [2025], SMOTE-EEN worked towards producing remarkable AUPRC score of 96%, F1-Score of 92% and ROC_AUC score of 100% for their choice model. This attested to the proved efficacy of SMOTE-EEN towards delivering optimised model performances in cases where it would be lacking without its implementation. Just as we saw in tables 4.1 and 4.4, SMOTE-EEN consistently outperformed other techniques like traditional Oversampling and SMOTE in almost all the datasets even in the case where no technique was employed, SMOTE-EEN once again proved its fidelity in optimising model performances. compared SMOTE-EEN with traditional oversampling techniques, such as SMOTE and ADASYN, on a publicly available credit card fraud dataset.

On the contrary, the study by Isangediok and Gajamannage [2022] reported a subpar performance for the SMOTE-EEN techinque where it consistently underperformed when compared with other techniques for the classifiers Random Forest, Xgboost and Logistic Regression with AUPRC scores of (68.4%, 81.7% and 71.2%) respectively. These results however would make sense as SMOTE-EEN was utilised as a standalone technique and no feature selection or engineering was performed on the dataset. The underperforming result of SMOTE-EEN albeit in this nature, still showcases its potential to unlock better results when combined with other techniques like feature engineering and feature selection. For the mere fact that the results reported by Isangediok and Gajamannage [2022] was from a standalone investigation of the SMOTE-EEN classifier, it is with much conviction that when combined with other techniques as seen in this study and that of Damanik and Liu [2025], the technique has the potential of unlocking better models performances.

4.5.2 Feature Selection in other studies versus this study

Feature selection plays a pivotal role in building efficient fraud detection systems by focusing on the most predictive variables, reducing computational overhead, and enhancing model interpretability. Recent research by Kumar and Mehta [2023]; Singh and Sharma [2024] has

showcased the effectiveness of statistical feature selection techniques such as Chi-Square and ANOVA (Analysis of Variance) in identifying critical fraud-related attributes.

In a particular study by Kumar and Mehta [2023], the Chi-Square test was employed to analyze categorical features, such as transaction type and merchant category. The method effectively ranked features by their dependency with the target variable (fraud status), leading to a 9% improvement in accuracy and a 14% increase in precision when the reduced feature set was used with logistic regression and decision tree classifiers. Another study by Singh and Sharma [2024] demonstrated the power of ANOVA in evaluating numerical features like transaction amounts and time intervals. By selecting features with statistically significant variance between fraudulent and non-fraudulent transactions, the study achieved a 10% gain in the AUC-ROC and an 8% improvement in F1-score when applied to SVM and neural network models. The authors emphasized that ANOVA-based feature selection not only improved predictive performance but also highlighted key patterns of fraudulent behavior, such as unusually high transaction values. These results do attest with that presented in this study which revealed ANOVA and Chi-Square selection techniques as the better candidates at optimizing model performances in CCFD studies.

Table 4.6: Comparison of Feature Selection Techniques with the SMOTE-EEN Class Imbalance Technique on (df1) for four classifiers

Split	Model	Technique	Balanced Accuracy	F1 Score	ROC AUC
Train	Decision Tree	ANOVA	1.00	1.00	1.00
		Chi-Sq	1.00	1.00	1.00
		IG	1.00	1.00	1.00
		Lasso	1.00	1.00	1.00
		RFE	1.00	1.00	1.00
	KNN	ANOVA	1.00	1.00	1.00
		Chi-Sq	1.00	1.00	1.00
		IG	1.00	1.00	1.00
		Lasso	0.99	0.99	1.00
		RFE	1.00	1.00	1.00
	AdaBoost	ANOVA	0.96	0.96	0.99
		Chi-Sq	0.96	0.96	0.99
		IG	0.94	0.94	0.98
		Lasso	0.85	0.84	0.91
		RFE	0.95	0.95	0.99
	XGBoost	ANOVA	1.00	1.00	1.00
		Chi-Sq	1.00	1.00	1.00
		IG	1.00	1.00	1.00
		Lasso	0.90	0.89	0.95
		RFE	1.00	1.00	1.00
Test	Decision Tree	ANOVA	0.89	0.69	0.89
		Chi-Sq	0.88	0.72	0.88
		IG	0.90	0.64	0.90
		Lasso	0.75	0.62	0.75
		RFE	0.91	0.66	0.91
	KNN	ANOVA	0.91	0.66	0.92
		Chi-Sq	0.92	0.68	0.92
		IG	0.90	0.60	0.90
		Lasso	0.64	0.50	0.67
		RFE	0.90	0.61	0.91
	AdaBoost	ANOVA	0.91	0.54	0.95
		Chi-Sq	0.91	0.55	0.95
		IG	0.92	0.55	0.95
		Lasso	0.79	0.64	0.83
		RFE	0.92	0.53	0.94
	XGBoost	ANOVA	0.91	0.75	0.94
		Chi-Sq	0.91	0.78	0.95
		IG	0.90	0.62	0.93
		Lasso	0.80	0.67	0.84
		RFE	0.90	0.63	0.95
Val	Decision Tree	ANOVA	0.92	0.71	0.92
		Chi-Sq	0.91	0.73	0.91
		IG	0.93	0.67	0.93
		Lasso	0.75	0.62	0.75
		RFE	0.91	0.66	0.91
	KNN	ANOVA	0.92	0.66	0.93
		Chi-Sq	0.93	0.68	0.94
		IG	0.93	0.61	0,93
		Lasso	0.64	0.50	0.66
		RFE	0.92	0.63	0.93
	AdaBoost	ANOVA	0.93	0.55	0.96
		Chi-Sq	0.93	0.55	0.96
		IG	0.93	0.55	0.95
		Lasso	0.79	0.63	0.83
		RFE	0.93	0.53	0.95
	XGBoost	ANOVA	0.92	0.75	0.95
		Chi-Sq	0.93	0.82	0.96
		IG	0.92	0.62	0.94
		Lasso	0.80	0.67	0.84
		RFE	0.92	0.64	0.95

Table 4.7: Comparison of Feature Selection Techniques with the SMOTE-EEN Class Imbalance Technique on (df2) for four classifiers

Split	Model	Technique	Balanced Accuracy	F1 Score	ROC AUC
Train	Decision Tree	ANOVA	1.00	1.00	1.00
		Chi-Sq	1.00	1.00	1.00
		IG	1.00	1.00	1.00
		Lasso	1.00	1.00	1.00
		RFE	1.00	1.00	1.00
	KNN	ANOVA	1.00	1.00	1.00
		Chi-Sq	1.00	1.00	1.00
		IG	1.00	1.00	1.00
		Lasso	1.00	1.00	1.00
		RFE	1.00	1.00	1.00
	AdaBoost	ANOVA	1.00	1.00	1.00
		Chi-Sq	1.00	1.00	1.00
		IG	0.96	0.96	0.99
		Lasso	1.00	1.00	1.00
		RFE	1.00	1.00	1.00
	XGBoost	ANOVA	1.00	1.00	1.00
		Chi-Sq	1.00	1.00	1.00
		IG	0.96	0.96	0.99
		Lasso	1.00	1.00	1.00
		RFE	1.00	1.00	1.00
Test	Decision Tree	ANOVA	1.00	1.00	1.00
		Chi-Sq	1.00	1.00	1.00
		IG	0.85	0.75	0.85
		Lasso	1.00	1.00	1.00
		RFE	0.99	0.96	0.99
	KNN	ANOVA	0.97	0.88	0.98
		Chi-Sq	0.97	0.88	0.98
		IG	0.85	0.74	0.88
		Lasso	0.97	0.88	0.98
		RFE	0.98	0.90	0.98
	AdaBoost	ANOVA	1.00	1.00	1.00
		Chi-Sq	1.00	1.00	1.00
		IG	0.86	0.87	0.95
		Lasso	1.00	1.00	1.00
		RFE	0.99	0.96	1.00
	XGBoost	ANOVA	1.00	0.99	1.00
		Chi-Sq	1.00	0.99	1.00
		IG	0.86	0.88	0.95
		Lasso	1.00	0.99	1.00
		RFE	0.99	0.95	1.00
Val	Decision Tree	ANOVA	1.00	1.00	1.00
		Chi-Sq	1.00	1.00	1.00
		IG	0.84	0.74	0.85
		Lasso	1.00	1.00	1.00
		RFE	0.99	0.96	0.99
	KNN	ANOVA	0.97	0.88	0.98
		Chi-Sq	0.97	0.88	0.98
		IG	0.84	0.74	0.87
		Lasso	0.97	0.88	0.98
		RFE	0.98	0.90	0.98
	AdaBoost	ANOVA	1.00	1.00	1.00
		Chi-Sq	1.00	1.00	1.00
		IG	0.86	0.87	0.95
		Lasso	1.00	1.00	1.00
		RFE	0.99	0.96	1.00
	XGBoost	ANOVA	1.00	0.99	1.00
		Chi-Sq	1.00	0.99	1.00
		IG	0.86	0.88	0.95
		Lasso	1.00	0.99	1.00
		RFE	0.99	0.95	1.00

Table 4.8: Comparison of Feature Selection Techniques with the SMOTE-EEN Class Imbalance Technique on (df3) for four classifiers

Split	Model	Technique	Balanced Accuracy	F1 Score	ROC AUC
Train	Decision Tree	ANOVA	1.00	1.00	1.00
		Chi-Sq	1.00	1.00	1.00
		IG	1.00	1.00	1.00
		Lasso	1.00	1.00	1.00
		RFE	1.00	1.00	1.00
	KNN	ANOVA	1.00	1.00	1.00
		Chi-Sq	1.00	1.00	1.00
		IG	1.00	1.00	1.00
		Lasso	1.00	1.00	1.00
		RFE	1.00	1.00	1.00
	AdaBoost	ANOVA	0.93	0.93	0.98
		Chi-Sq	0.89	0.89	0.96
		IG	0.90	0.90	0.96
		Lasso	0.90	0.90	0.96
		RFE	0.90	0.90	0.96
	XGBoost	ANOVA	0.96	0.96	0.99
		Chi-Sq	0.95	0.95	0.99
		IG	0.95	0.95	0.99
		Lasso	0.95	0.95	0.99
		RFE	0.93	0.93	0.98
Test	Decision Tree	ANOVA	0.85	0.56	0.85
		Chi-Sq	0.84	0.58	0.84
		IG	0.84	0.57	0.84
		Lasso	0.83	0.57	0.83
		RFE	0.83	0.55	0.85
	KNN	ANOVA	0.71	0.52	0.73
		Chi-Sq	0.98	0.88	0.99
		IG	0.98	0.88	0.99
		Lasso	0.98	0.88	0.99
		RFE	0.76	0.51	0.80
	AdaBoost	ANOVA	0.89	0.54	0.96
		Chi-Sq	0.88	0.54	0.95
		IG	0.89	0.54	0.95
		Lasso	0.88	0.54	0.95
		RFE	0.86	0.57	0.93
	XGBoost	ANOVA	0.90	0.56	0.96
		Chi-Sq	0.92	0.59	0.97
		IG	0.91	0.58	0.97
		Lasso	0.91	0.59	0.97
		RFE	0.87	0.55	0.93
Val	Decision Tree	ANOVA	0.85	0.56	0.85
		Chi-Sq	0.84	0.58	0.84
		IG	0.83	0.57	0.83
		Lasso	0.82	0.57	0.82
		RFE	0.83	0.55	0.83
	KNN	ANOVA	0.71	0.52	0.73
		Chi-Sq	0.99	0.89	0.99
		IG	0.99	0.89	0.99
		Lasso	0.99	0.89	0.99
		RFE	0.76	0.51	0.80
	AdaBoost	ANOVA	0.89	0.54	0.96
		Chi-Sq	0.88	0.54	0.93
		IG	0.88	0.54	0.95
		Lasso	0.88	0.54	0.95
		RFE	0.85	0.57	0.92
	XGBoost	ANOVA	0.89	0.56	0.96
		Chi-Sq	0.91	0.59	0.97
		IG	0.91	0.58	0.96
		Lasso	0.90	0.59	0.96
		RFE	0.86	0.55	0.93

Table 4.9: Comparison of Feature Selection Techniques with the SMOTE-EEN Class Imbalance Technique on (df4) for four classifiers

Split	Model	Technique	Balanced Accuracy	F1 Score	ROC AUC
Train	Decision Tree	ANOVA	1.00	1.00	1.00
		Chi-Sq	1.00	1.00	1.00
		IG	1.00	1.00	1.00
		Lasso	1.00	1.00	1.00
		RFE	1.00	1.00	1.00
	KNN	ANOVA	0.99	0.99	1.00
		Chi-Sq	0.99	0.99	1.00
		IG	0.98	0.98	1.00
		Lasso	0.99	0.99	1.00
		RFE	0.98	0.98	1.00
	AdaBoost	ANOVA	0.79	0.79	0.88
		Chi-Sq	0.85	0.84	0.91
		IG	0.78	0.77	0.85
		Lasso	0.85	0.84	0.91
		RFE	0.78	0.77	0.85
	XGBoost	ANOVA	0.81	0.81	0.90
		Chi-Sq	0.90	0.89	0.95
		IG	0.80	0.79	0.88
		Lasso	0.90	0.89	0.95
		RFE	0.80	0.79	0.88
Test	Decision Tree	ANOVA	0.67	0.55	0.67
		Chi-Sq	0.75	0.62	0.75
		IG	0.65	0.54	0.65
		Lasso	0.75	0.62	0.75
		RFE	0.65	0.54	0.65
	KNN	ANOVA	0.68	0.53	0.71
		Chi-Sq	0.64	0.50	0.67
		IG	0.66	0.51	0.68
		Lasso	0.64	0.50	0.67
		RFE	0.66	0.51	0.68
	AdaBoost	ANOVA	0.74	0.51	0.82
		Chi-Sq	0.79	0.64	0.83
		IG	0.73	0.58	0.76
		Lasso	0.79	0.64	0.83
		RFE	0.73	0.58	0.76
	XGBoost	ANOVA	0.74	0.53	0.87
		Chi-Sq	0.80	0.67	0.84
		IG	0.72	0.57	0.75
		Lasso	0.80	0.67	0.84
		RFE	0.72	0.57	0.75
Val	Decision Tree	ANOVA	0.67	0.55	0.67
		Chi-Sq	0.75	0.62	0.75
		IG	0.65	0.54	0.65
		Lasso	0.75	0.62	0.75
		RFE	0.65	0.54	0.65
	KNN	ANOVA	0.68	0.53	0.71
		Chi-Sq	0.64	0.50	0.66
		IG	0.65	0.51	0.68
		Lasso	0.64	0.50	0.66
		RFE	0.65	0.51	0.68
	AdaBoost	ANOVA	0.74	0.51	0.81
		Chi-Sq	0.79	0.63	0.83
		IG	0.72	0.58	0.75
		Lasso	0.79	0.63	0.83
		RFE	0.72	0.58	0.75
	XGBoost	ANOVA	0.73	0.53	0.81
		Chi-Sq	0.80	0.67	0.84
		IG	0.71	0.57	0.75
		Lasso	0.80	0.67	0.84
		RFE	0.71	0.57	0.75

Chapter 5

Conclusion

In the context of CCF detection studies, accuracy means everything and with modern day ML frameworks, optimal solutions are frequently researched to solve one of the biggest issues crippling the financial industry and the finances of vulnerable individuals. In this study, CCF datasets which had undergone domain-dependent feature engineering combined with SMOTE-EEN for handling class imbalance and either ANOVA or Chi-Square feature selection techniques provided a solid foundation for mitigating performance issues related to ML classifiers in the domain. The study also uncovered that the number of features engineered/available all fall at the mercy of the chosen feature selection technique with high dependence on their importance to the feature selection technique. The study overall, has highlighted the synergy between feature engineering, SMOTE-EEN and feature selection techniques, particularly ANOVA or Chi-Square Test which are crucial for optimizing model performance in imbalanced datasets. Chi-Square and ANOVA consistently offered robust and generalizable results, making them ideal for most scenarios. Additionally, advanced models like XGBoost consistently outperformed simpler models like KNN, emphasizing the importance of selecting models that align with the complexity of the feature selection process.

5.1 How the research answered the research questions

1. **The study overall, aimed to address the following research questions:**

   - **RQ1**: How does the engineering of relevant features help further unravel fraud patterns within a dataset?
   - **RQ2**: Which of the class imbalance techniques between SMOTE, SMOTE-EEN and traditional oversampling offers the most optimal performance for ML classifiers?
   - **RQ3**: Which of the feature selection techniques between Chi-Squared score, L1 regularisation, Information Gain, ANOVA, RFE scores is the most effective for opti-mal model performance in CCF detection

2. **Key Findings**

   For RQ1: The study revealed that the performance of the models was significantly influenced by the choice of feature selection techniques, as certain features were deemed unimportant while others were prioritized. Among the engineered features, some were selected for model evaluation by the feature selection methods, while others were excluded. This suggests that the impact of feature engineering on model performance is not absolute but context-dependent. Specifically, none of the engineered features in datasets df1 and

df2 were selected by the feature selection techniques, whereas almost all the engineered features in datasets df3 and df4 were chosen by one or more methods. The findings in general underscore the critical role of feature selection techniques in determining model performance, highlighting the nuanced interplay between feature engineering and model efficacy. The variation observed in the selection of the engineered features across the different datasets emphasized that the influence of feature engineering is highly context-specific and depends on the dataset characteristics and the feature selection method applied. These results reinforce the importance of carefully aligning feature engineering efforts with robust selection techniques to optimize model outcomes where necessary.

For RQ2: The study revealed the SMOTE-EEN class imbalance handler as the best option for optimising model performance. Although, it proved to be a computationally expensive process, it still proved its class as the best handler for dealing with CCFD in this study.

For RQ3: The study demonstrated that Chi-Square and ANOVA are the best feature selection techniques for optimising a model's performance. Thus a careful interplay between all these techniques in RQ1, RQ2 and RQ3 offer better model performances across all the four (4) popular classification models used in the study. Specifically, the Xgboost classifier came out top across almost all the datasets showing its efficacy in CCFD.

5.2 Limitations of the work

This study albeit its comprehensiveness, had several limitations that need to be acknowledged:

- **Dataset Dependency:** The effectiveness of the proposed methods was dependent on the quality and characteristics of the datasets used as we saw with the results for df2, that is something called nearly perfect dataset with entirely numerical features. In such circumstances model enhancing techniques play little to no role in improving performances rather in some cases, decreasing the performances of models that otherwise should be performing better. Such extreme cases inherent in a dataset, has the tendency of limiting the generalizability of the findings.
- **Computational Complexity:** Advanced techniques like SMOTE-EEN, Xgboost and Recursive Feature Elimination (RFE) required significant computational resources and time for fine-tuning, potentially limiting their application in resource-constrained environments.
- **Location specific dataset acquisition:** One of the major heartaches of this study was the lack of an african context-based datasets, as financial datasets are difficult to acquire taking into consideration the many privacy laws governing financial data.
- **Focus on Binary Classification:** The research primarily focused on binary classification in CCF detection. Multi-class fraud detection scenarios are not explored in depth.
- **Domain-dependent study**: This study focused mainly on CCFD. It would be interesting to see the applicability of the findings from this study to other domains like spam-filtering, tumor detection and other areas.
- **Real-time Applicability:** The study also focused heavily on evaluating the model enhancing techniques outside the confines of a real-time fraud detection system and did not address the potential challenges and fallback plan for deploying these methods in real-time fraud detection systems.

5.3 Future work

With all these said, future research will look to explore:

- hybrid approaches that integrate multiple feature selection techniques to leverage their combined strengths where singular techniques were lacking in bid to offer better model performances yet not captured by the advances in research.
- acquiring Southern African based datasets to test applicability of these findings in an african based context.
- deeper techniques that can prevent CCF related activities and also study other techniques that could strengthen fraud detection models against future adversarial attacks where fraudsters are growing manipulative means to bypass detection. This would ideally include the generation of adversarial examples and the implementation of robust training techniques to enhance model resilience.

References

[Adewale and Mbakwe 2022] Sikiru Ademola Adewale and Amarachi Blessing Mbakwe. Machine learning algorithms for credit card fraud detection. 2022.

[Alfaiz and Fati 2022] Noor Saleh Alfaiz and Suliman Mohamed Fati. Enhanced credit card fraud detection model using machine learning. *Electronics*, 11(4):662, 2022.

[Asha and KR 2021] RB Asha and Suresh Kumar KR. Credit card fraud detection using artificial neural network. *Global Transitions Proceedings*, 2(1):35–41, 2021.

[Ashraf *et al.* 2024] Kahakashan Ashraf, Sadia Nawar, Md Hamid Hosen, Mohammad Tanvirul Islam, and Mohammed Nazim Uddin. Beyond the black box: Employing lime and shap for transparent health predictions with machine learning models. In *2024 International Conference on Advances in Computing, Communication, Electrical, and Smart Systems (iCACCESS)*, pages 1–6. IEEE, 2024.

[Awati *et al.* 2019] Chetan Awati, Rashmi More, Sonam Patil, and Suresh Shirgave. A review on credit card fraud detection using machine learning. *International Journal of Scientific & technology research*, 8(10):1217–1220, 2019.

[Azhan and Meraj 2020] Mohammed Azhan and Shazli Meraj. Credit card fraud detection using machine learning and deep learning techniques. In *2020 3rd International Conference on Intelligent Sustainable Systems (ICISS)*, pages 514–518. IEEE, 2020.

[Bagga *et al.* 2020] Siddhant Bagga, Anish Goyal, Namita Gupta, and Arvind Goyal. Credit card fraud detection using pipeling and ensemble learning. *Procedia Computer Science*, 173:104–112, 2020.

[Batista *et al.* 2019] Gustavo E. A. P. A. Batista, Ronaldo C. Prati, and Maria Carolina Monard. A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD Explorations Newsletter*, 6(1):20–29, 2019.

[Bekmaganbet 2021] Galym Bekmaganbet. Crime prediction and forecasting: Feature selection and vulnerable region detection models. 2021.

[Bhattacharyya *et al.* 2011] Siddhartha Bhattacharyya, Sanjeev Jha, Kurian Tharakunnel, and J Christopher Westland. Data mining for credit card fraud: A comparative study. *Decision support systems*, 50(3):602–613, 2011.

[Bhavsar and Patil 2021] Archana Bhavsar and Tejashri Patil. Exploration of various machine learning algorithms. *PRATIBHA: INTERNATIONAL JOURNAL OF SCIENCE, SPIRITUALITY, BUSINESS AND TECHNOLOGY (IJSSBT)*, 2021.

[Brownlee 2019] Jason Brownlee. How to choose a feature selection method for machine learning. *Machine Learning Mastery*, 10, 2019.

[Buda *et al.* 2018] Mateusz Buda, Atsuto Maki, and Maciej A Mazurowski. A systematic study of the class imbalance problem in convolutional neural networks. *Neural Networks*, 106:249–259, 2018.

[Büyükkeçeci and Okur 2022] Mustafa Büyükkeçeci and Mehmet Cudi Okur. A comprehensive review of feature selection and feature selection stability in machine learning. *Gazi University Journal of Science*, 36(4):1506–1520, 2022.

[Chandrashekar and Sahin 2014] Girish Chandrashekar and Ferat Sahin. A survey on feature selection methods. *Computers & Electrical Engineering*, 40(1):16–28, 2014.

[Chawla *et al.* 2021] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, 2021.

[Chen and Lai 2021] Joy Iong-Zong Chen and Kong-Long Lai. Deep convolution neural network model for credit-card fraud detection and alert. *Journal of Artificial Intelligence*, 3(02):101–112, 2021.

[Cheng *et al.* 2020] Liang Cheng, Liang Wang, and Yu Zhang. Hybrid sampling with smote and borderline-smote for imbalanced data classification. *Applied Intelligence*, 50(2):394–405, 2020.

[Damanik and Liu 2025] Nurafni Damanik and Chuan-Ming Liu. Advanced fraud detection: Leveraging k-smoteenn and stacking ensemble to tackle data imbalance and extract insights. *IEEE Access*, 2025.

[Dhal and Azad 2022] Pradip Dhal and Chandrashekhar Azad. A comprehensive survey on feature selection in the various fields of machine learning. *Applied Intelligence*, 52(4):4543–4581, 2022.

[Dornadula and Geetha 2019] Vaishnavi Nath Dornadula and Sa Geetha. Credit card fraud detection using machine learning algorithms. *Procedia computer science*, 165:631–641, 2019.

[Fadaei Noghani and Moattar 2017] F Fadaei Noghani and M Moattar. Ensemble classification and extended feature selection for credit card fraud detection. *Journal of AI and data mining*, 5(2):235–243, 2017.

[Faraji 2022] Zahra Faraji. A review of machine learning applications for credit card fraud detection with a case study. *SEISENSE Journal of Management*, 5(1):49–59, 2022.

[Fayyomi *et al.* 2021] Aisha Mohammad Fayyomi, Derar Eleyan, and Amina Eleyan. A survey paper on credit card fraud detection techniques. *International Journal of Scientific & Technology Research*, 10(09), 2021.

[Fernández *et al.* 2020] Alberto Fernández, Salvador García, Mikel Galar, Ronaldo C. Prati, Bartosz Krawczyk, and Francisco Herrera. *Learning from imbalanced data sets*. Springer, 2020.

[García and Herrera 2020] Verónica García and Francisco Herrera. An adaptive ensemble method for class imbalance problem in imbalanced data streams. *Pattern Recognition*, 107:107474, 2020.

[Gnana *et al.* 2016] D Asir Antony Gnana, S Appavu Alias Balamurugan, and E Jebamalar Leavline. Literature review on feature selection methods for high-dimensional data. *International Journal of Computer Applications*, 136(1):9–17, 2016.

[Hammed and Soyemi 2020] Mudasiru Hammed and Jumoke Soyemi. An implementation of decision tree algorithm augmented with regression analysis for fraud detection in credit card. *International Journal of Computer Science and Information Security (IJCSIS)*, 18(2):79–88, 2020.

[Han *et al.* 2020] Hui Han, Wen Y. Wang, and Bing H. Mao. Borderline-smote: A new over-sampling method in imbalanced data sets learning. *Neurocomputing*, 70(1-3):196–206, 2020.

[Hasan and Bao 2021] Najmul Hasan and Yukun Bao. Comparing different feature selection algorithms for cardiovascular disease prediction. *Health and Technology*, 11(1):49–62, 2021.

[He *et al.* 2020] Shida He, Fei Guo, Quan Zou, et al. Mrmd2. 0: a python tool for machine learning with feature ranking and reduction. *Current Bioinformatics*, 15(10):1213–1221, 2020.

[Husejinovic 2020] Admel Husejinovic. Credit card fraud detection using naive bayesian and c4. 5 decision tree classifiers. *Husejinovic, A.(2020). Credit card fraud detection using naive Bayesian and C*, 4:1–5, 2020.

[Isangediok and Gajamannage 2022] Mary Isangediok and Kelum Gajamannage. Fraud detection using optimized machine learning tools under imbalance classes. In *2022 IEEE International Conference on Big Data (Big Data)*, pages 4275–4284. IEEE, 2022.

[Jain *et al.* 2019] Yashvi Jain, Namrata Tiwari, Shripriya Dubey, and Sarika Jain. A comparative analysis of various credit card fraud detection techniques. *Int J Recent Technol Eng*, 7(5S2):402–407, 2019.

[Ji 2021] Yingchao Ji. *Explainable AI methods for credit card fraud detection: Evaluation of LIME and SHAP through a User Study*, 2021.

[Keerthana *et al.* 2021] S Keerthana, B Khanitkar, M SrikanthRaju, and YB Varun. Machine learning algorithms for credit card fraud detection. 2021.

[Kovács 2019] György Kovács. An empirical comparison and evaluation of minority oversampling techniques on a large number of imbalanced datasets. *Applied Soft Computing*, 83:105662, 2019.

[Krawczyk 2016] Bartosz Krawczyk. Learning from imbalanced data: Open challenges and future directions. *Progress in Artificial Intelligence*, 5(4):221–232, 2016.

[Kumar and Mehta 2023] Ravi Kumar and Anil Mehta. Chi-square feature selection for fraudulent transaction detection in financial systems. *International Journal of Data Science*, 12(4):289–302, 2023.

[Kumar *et al.* 2021] Nishant Kumar, Kirti Soni, and Ravinder Agarwal. A comprehensive study of different feature selection methods and machine-learning techniques for sodar structure classification. *Modeling Earth Systems and Environment*, 7(1):209–220, 2021.

[Lebichot *et al.* 2020] Bertrand Lebichot, Yann-Aël Le Borgne, Liyun He-Guelton, Frederic Oblé, and Gianluca Bontempi. Deep-learning domain adaptation techniques for credit cards fraud detection. In *Recent Advances in Big Data and Deep Learning: Proceedings of the INNS Big Data and Deep Learning Conference INNSBDDL2019, held at Sestri Levante, Genova, Italy 16-18 April 2019*, pages 78–88. Springer, 2020.

[Lilhore and Patil 2018] Umesh Kumar Lilhore and Vipul Patil. A survey on different data mining & machine learning methods for credit card fraud detection. *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, 3(5):320–325, 2018.

[Liu and Setiono 2022] Huan Liu and Rudy Setiono. Feature selection and classification–a probabilistic wrapper approach. In *Industrial and engineering applications or artificial intelligence and expert systems*, pages 419–424. CRC Press, 2022.

[Liu *et al.* 2019] Haoyue Liu, MengChu Zhou, and Qing Liu. An embedded feature selection method for imbalanced data classification. *IEEE/CAA Journal of Automatica Sinica*, 6(3):703–715, 2019.

[Mamdouh Farghaly and Abd El-Hafeez 2023] Heba Mamdouh Farghaly and Tarek Abd El-Hafeez. A high-quality feature selection method based on frequent and correlated items for text classification. *Soft Computing*, 27(16):11259–11274, 2023.

[Maniraj *et al.* 2019] SP Maniraj, Aditya Saini, Shadab Ahmed, and Swarna Sarkar. Credit card fraud detection using machine learning and data science. *International Journal of Engineering Research*, 8(9):110–115, 2019.

[More and Rana 2021] Abhijeet More and Shashank Rana. Resampling techniques for imbalanced time series classification: A review. *Pattern Recognition Letters*, 138:85–95, 2021.

[Muntasir Nishat *et al.* 2022] Mirza Muntasir Nishat, Fahim Faisal, Ishrak Jahan Ratul, Abdullah Al-Monsur, Abrar Mohammad Ar-Rafi, Sarker Mohammad Nasrullah, Md Taslim Reza, and Md Rezaul Hoque Khan. A comprehensive investigation of the performances of different machine learning classifiers with smote-enn oversampling technique and hyperparameter optimization for imbalanced heart failure dataset. *Scientific Programming*, 2022(1):3649406, 2022.

[Mustaqim *et al.* 2021] Adi Zaenul Mustaqim, Sumarni Adi, Yoga Pristyanto, and Yuli Astuti. The effect of recursive feature elimination with cross-validation (rfecv) feature selection algorithm toward classifier performance on credit card fraud detection. In *2021 International conference on artificial intelligence and computer science technology (ICAICST)*, pages 270–275. IEEE, 2021.

[Nguyen *et al.* 2019] Huy Minh Nguyen, Eric W Cooper, and K Kamei. Borderline smote for imbalanced data classification. *International Journal of Machine Learning and Cybernetics*, 10(5):101–111, 2019.

[Omuya *et al.* 2021] Erick Odhiambo Omuya, George Onyango Okeyo, and Michael Waema Kimwele. Feature selection for classification using principal component analysis and information gain. *Expert Systems with Applications*, 174:114765, 2021.

[Pike 2019] Elizabeth R Pike. Defending data: toward ethical protections and comprehensive data governance. *Emory LJ*, 69:687, 2019.

[Pudjihartono *et al.* 2022] Nicholas Pudjihartono, Tayaza Fadason, Andreas W Kempa-Liehr, and Justin M O'Sullivan. A review of feature selection methods for machine learning-based disease risk prediction. *Frontiers in Bioinformatics*, 2:927312, 2022.

[Rtayli and Enneya 2020] Naoufal Rtayli and Nourddine Enneya. Enhanced credit card fraud detection based on svm-recursive feature elimination and hyper-parameters optimization. *Journal of Information Security and Applications*, 55:102596, 2020.

[Saheed *et al.* 2022] Yakub Kayode Saheed, Usman Ahmad Baba, and Mustafa Ayobami Raji. Big data analytics for credit card fraud detection using supervised machine learning models. In *Big Data Analytics in the Insurance Market*, pages 31–56. Emerald Publishing Limited, 2022.

[Salih *et al.* 2024] Ahmed M Salih, Zahra Raisi-Estabragh, Ilaria Boscolo Galazzo, Petia Radeva, Steffen E Petersen, Karim Lekadir, and Gloria Menegaz. A perspective on explainable artificial intelligence methods: Shap and lime. *Advanced Intelligent Systems*, page 2400304, 2024.

[Shaji *et al.* 2021] Anchana Shaji, Sumitra Binu, Akhil M Nair, and Jossy George. Fraud detection in credit card transaction using ann and svm. In *Ubiquitous Communications and Network Computing: 4th EAI International Conference, UBICNET 2021, Virtual Event, March 2021, Proceedings*, pages 187–197. Springer, 2021.

[Singh and Sharma 2024] Priya Singh and Rajesh Sharma. Anova-based feature selection for improving fraud detection models. *Transactions on Machine Learning and Data Analytics*, 8(1):56–70, 2024.

[Singh *et al.* 2022] Amit Singh, Ranjeet Kumar Ranjan, and Abhishek Tiwari. Credit card fraud detection under extreme imbalanced data: a comparative study of data-level algorithms. *Journal of Experimental & Theoretical Artificial Intelligence*, 34(4):571–598, 2022.

[TAGHIYEV *et al.* 2021] KHAYALADDIN R TAGHIYEV, TAMERLAN H RUSTAMOV, and ARAZ A HASANZADE. Analysis of payment cards fraud transactions and measures to prevent them. *Economic innovations*, 23(2 (79)):172–184, 2021.

[Tan 2007] Feng Tan. Improving feature selection techniques for machine learning. 2007.

[Tangirala 2020] Suryakanthi Tangirala. Evaluating the impact of gini index and information gain on classification using decision tree classifier algorithm. *International Journal of Advanced Computer Science and Applications*, 11(2):612–619, 2020.

[Trivedi *et al.* 2020] Naresh Kumar Trivedi, Sarita Simaiya, Umesh Kumar Lilhore, and Sanjeev Kumar Sharma. An efficient credit card fraud detection model based on machine learning methods. *International Journal of Advanced Science and Technology*, 29(5):3414–3424, 2020.

[van der Ploeg and Steyerberg 2016] Tjeerd van der Ploeg and Ewout W Steyerberg. Feature selection and validated predictive performance in the domain of legionella pneumophila: a comparative study. *BMC Research Notes*, 9:1–7, 2016.

[Wang *et al.* 2021] Pin Wang, En Fan, and Peng Wang. Comparative analysis of image classification algorithms based on traditional machine learning and deep learning. *Pattern Recognition Letters*, 141:61–67, 2021.

[Wang *et al.* 2022] Jiaqi Wang, Yan Zhang, and Yun Zhou. A comprehensive review on handling imbalanced data: Data-level and algorithm-level methods. *Pattern Recognition*, 124:108137, 2022.

[Wong and Yeh 2019] Tzu-Tsung Wong and Po-Yang Yeh. Reliable accuracy estimates from k-fold cross validation. *IEEE Transactions on Knowledge and Data Engineering*, 32(8):1586–1594, 2019.

[Zhao *et al.* 2019] Zhenyu Zhao, Radhika Anand, and Mallory Wang. Maximum relevance and minimum redundancy feature selection methods for a marketing machine learning platform. In *2019 IEEE international conference on data science and advanced analytics (DSAA)*, pages 442–452. IEEE, 2019.

[Zhou *et al.* 2021] Zhiyong Zhou, Lingyan Wu, and Qingwei Zhang. Adaptive cost-sensitive deep learning for imbalanced data classification. *Neural Processing Letters*, 53(2):1099–1115, 2021.

Appendix 1

This appendix contains list of features engineered for the various datasets.

- Dataset 1 was originally subjected to PCA thus having its features hidden. In such cases, it's difficult to decipher the context one is operating in. So product features and quotient features were created to study its potential effect on study.
  - V_div
  - V_prod
- Dataset 2 is what I termed a near perfect dataset. features like:
  - Transaction Frequency
  - Average Transaction Amount
  - Device Location Consistency

  These extra features were designed to help understand the behaviors surrounding fraudulent activities.
- Dataset 3 had the following features engineered:
  - trans_date & trans_time in seconds
  - day of week
  - day of month
  - lat_diff: this feature was engineered by finding latitudinal coordinate of the most central location in the specific country and subtracting it from the merch_lat feature.
  - lat_long: this feature was engineered by finding longitudinal coordinate of the most central location in the specific country and subtracting it from the merch_long feature.
  - Harvesian distance: The Haversine distance calculated the shortest path between two points using their latitudes and longitudes.

    $$d = 2r \arcsin\left(\sqrt{\sin^2\left(\frac{\phi_2 - \phi_1}{2}\right) + \cos(\phi_1)\cos(\phi_2)\sin^2\left(\frac{\lambda_2 - \lambda_1}{2}\right)}\right)$$

    Where:
    - r is the radius of the sphere (e.g., Earth's radius ≈ 6371 km).
    - ϕ_1, ϕ_2 are the latitudes of points 1 and 2 in radians.
    - λ_1, λ_2 are the longitudes of points 1 and 2 in radians.
  - Euclidean Distance: The Euclidean distance was calculated between two points in n-dimensional space.

    $$d = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

    Where:

        * x_i, y_i were the coordinates of the two latitudinal and longitudinal points in n-dimensional space.

- Dataset 4 had similar patterns and features as in dataset 3. So similar features were engineered for this dataset too. Other features engineered exclusively for dataset 4 were:
    - Account maturity
    - Account age days

Appendix 2

This appendix contains list of figures.

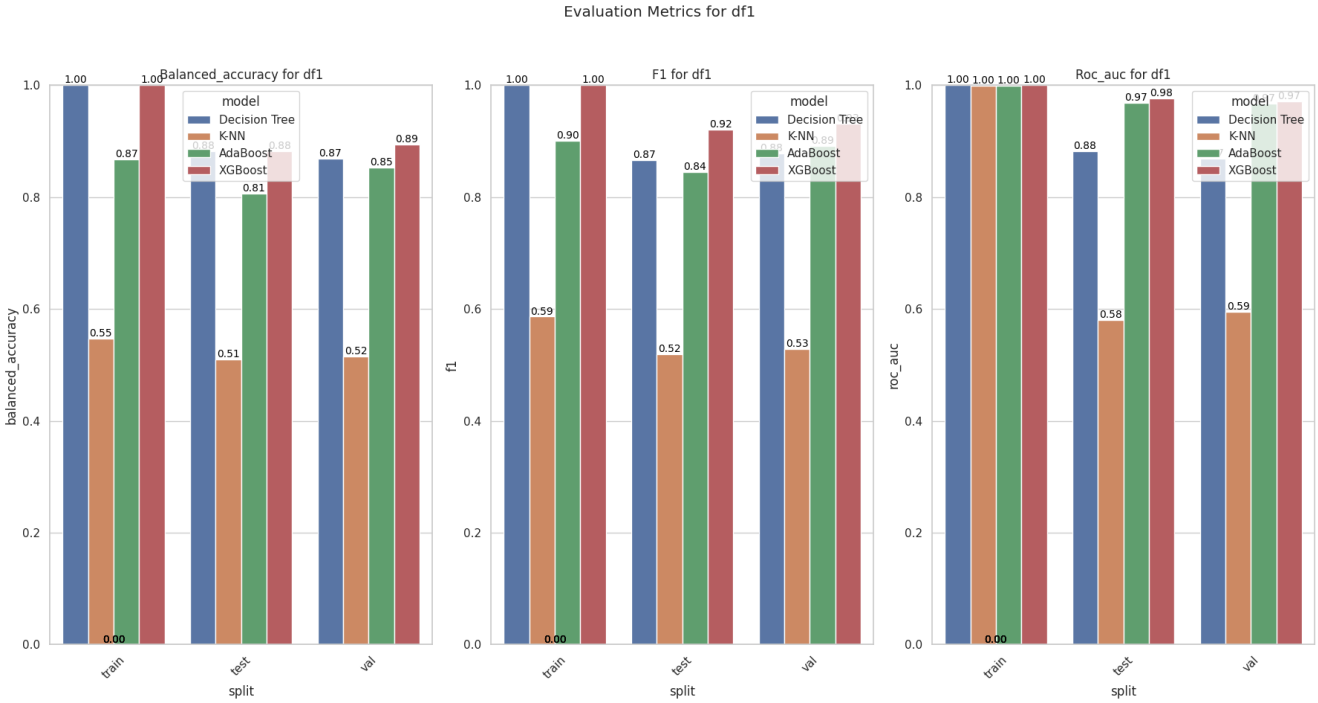

Figure 1: Model performances on raw dataset1

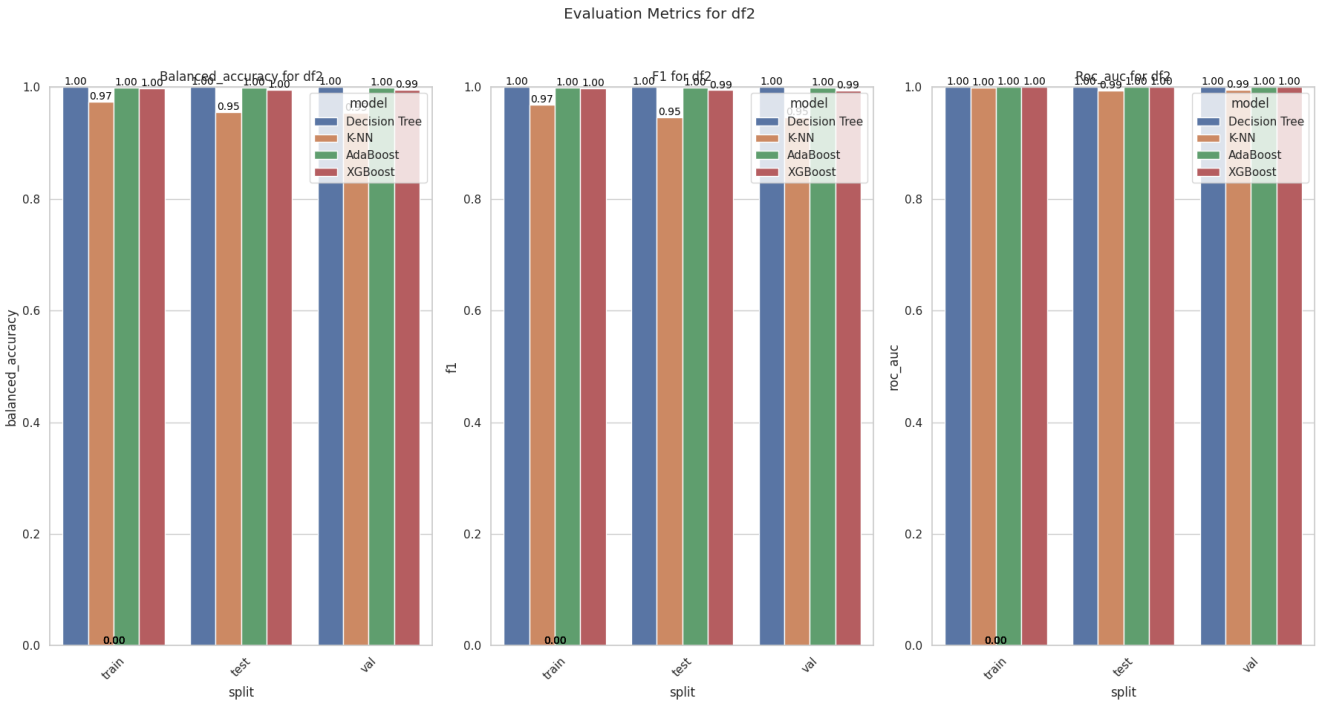

Figure 2: Model performances on raw dataset2

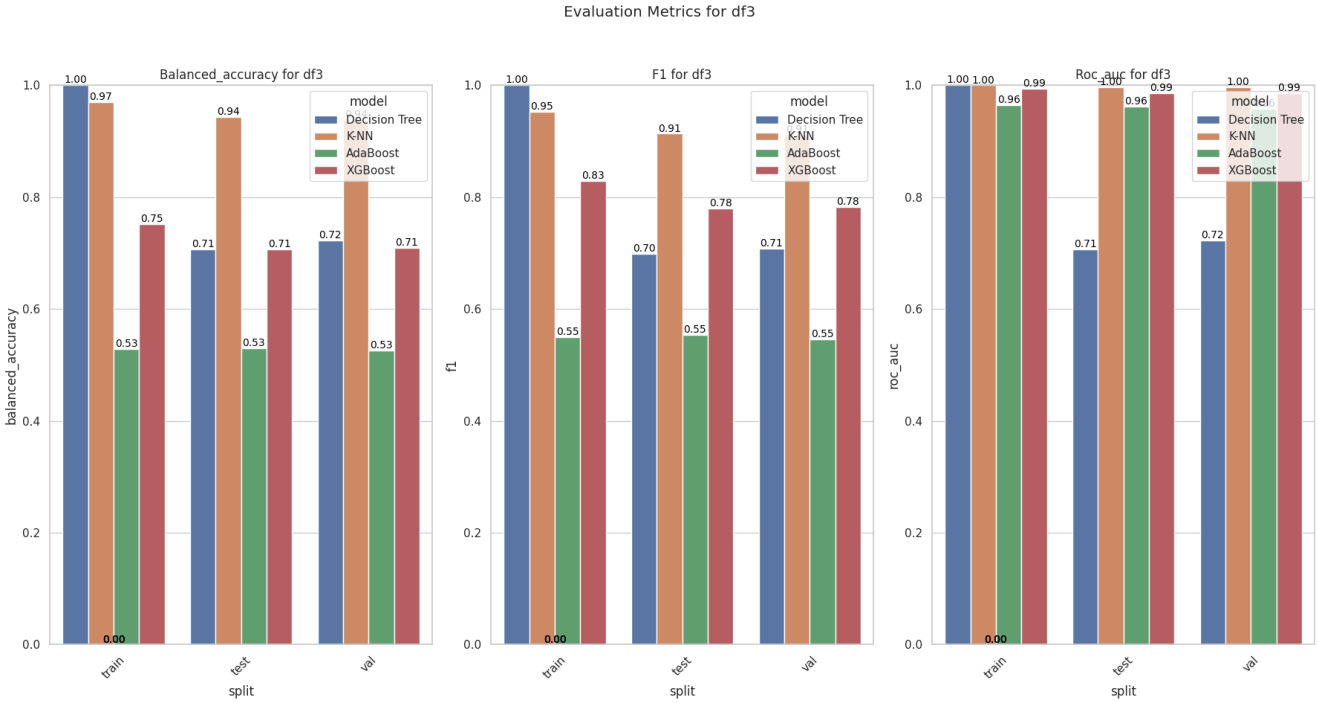

Figure 3: Model performances on raw dataset3

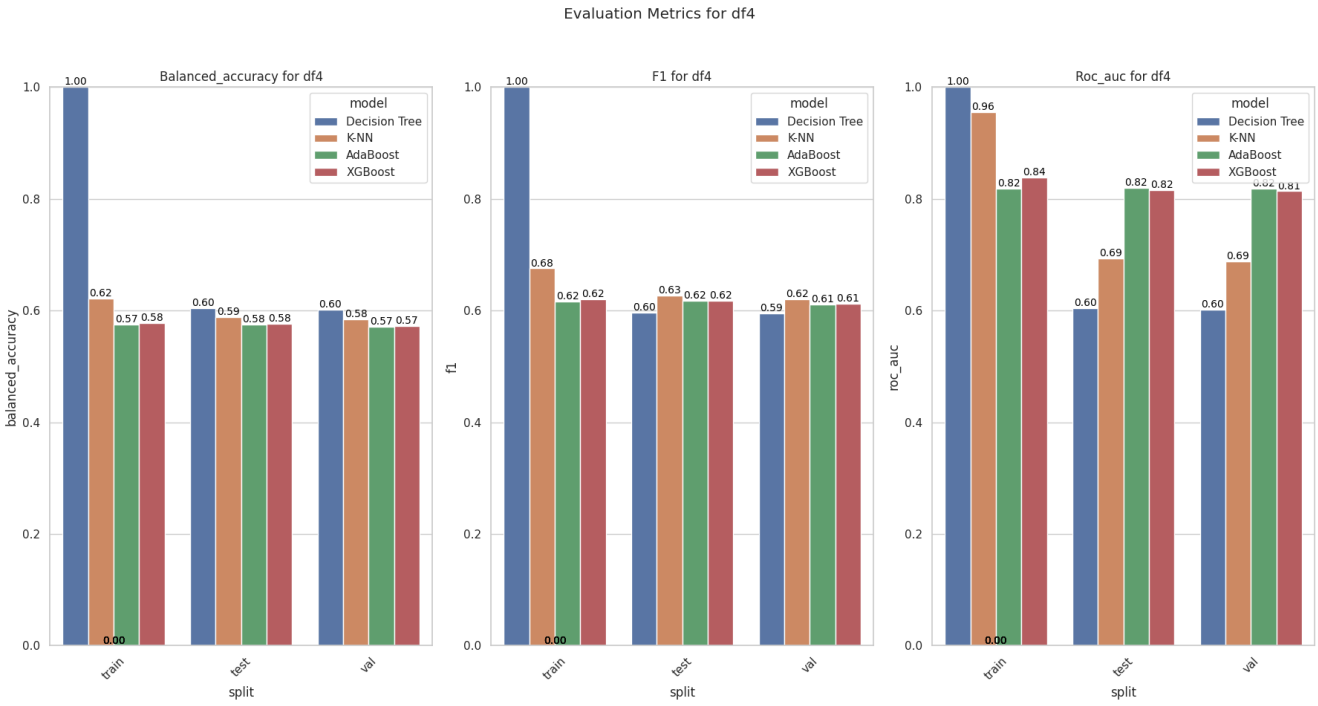

Figure 4: Model performances on raw dataset4

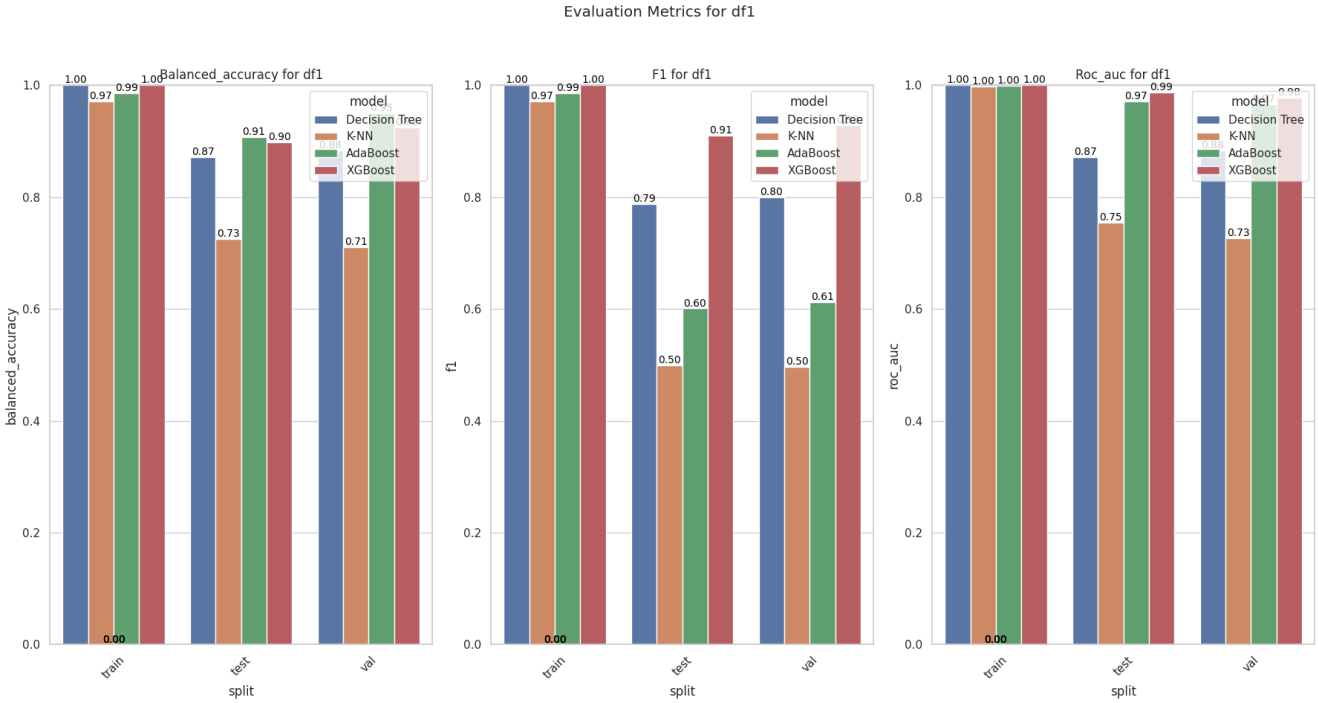

Figure 5: Model performances on SMOTE handled dataset1

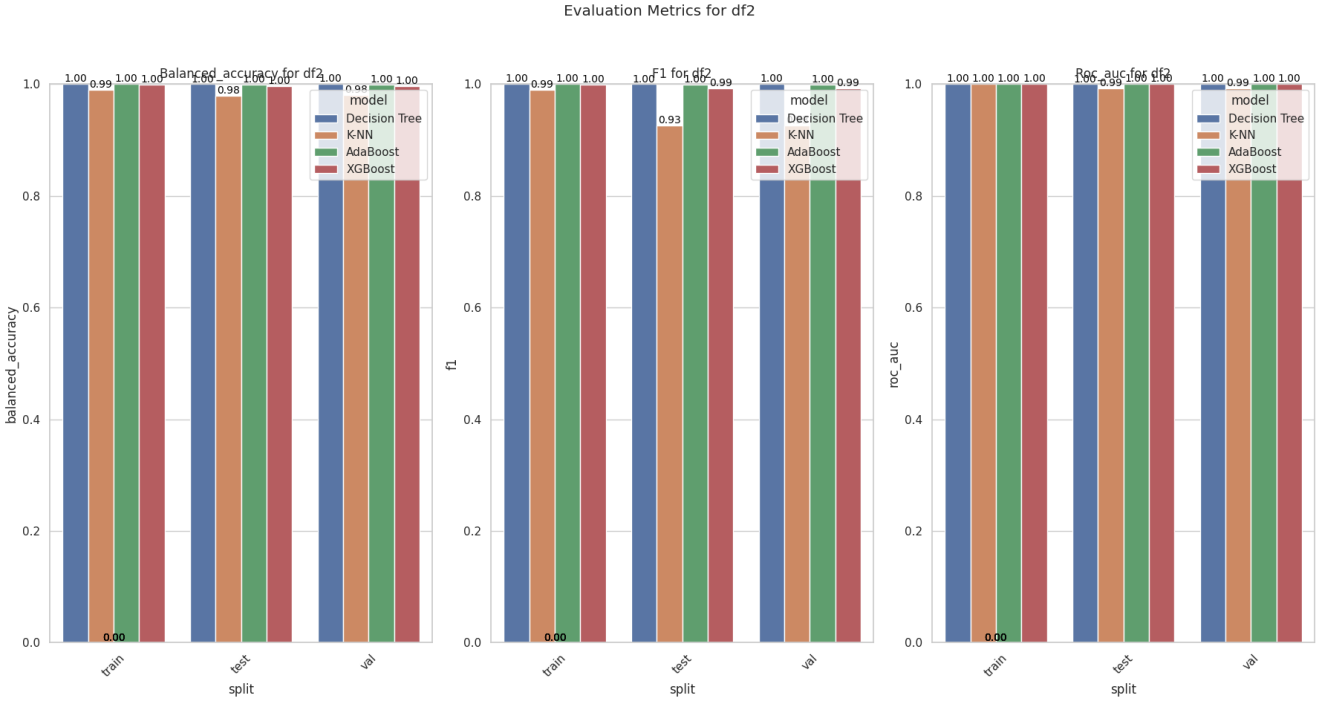

Figure 6: Model performances on SMOTE handled dataset2

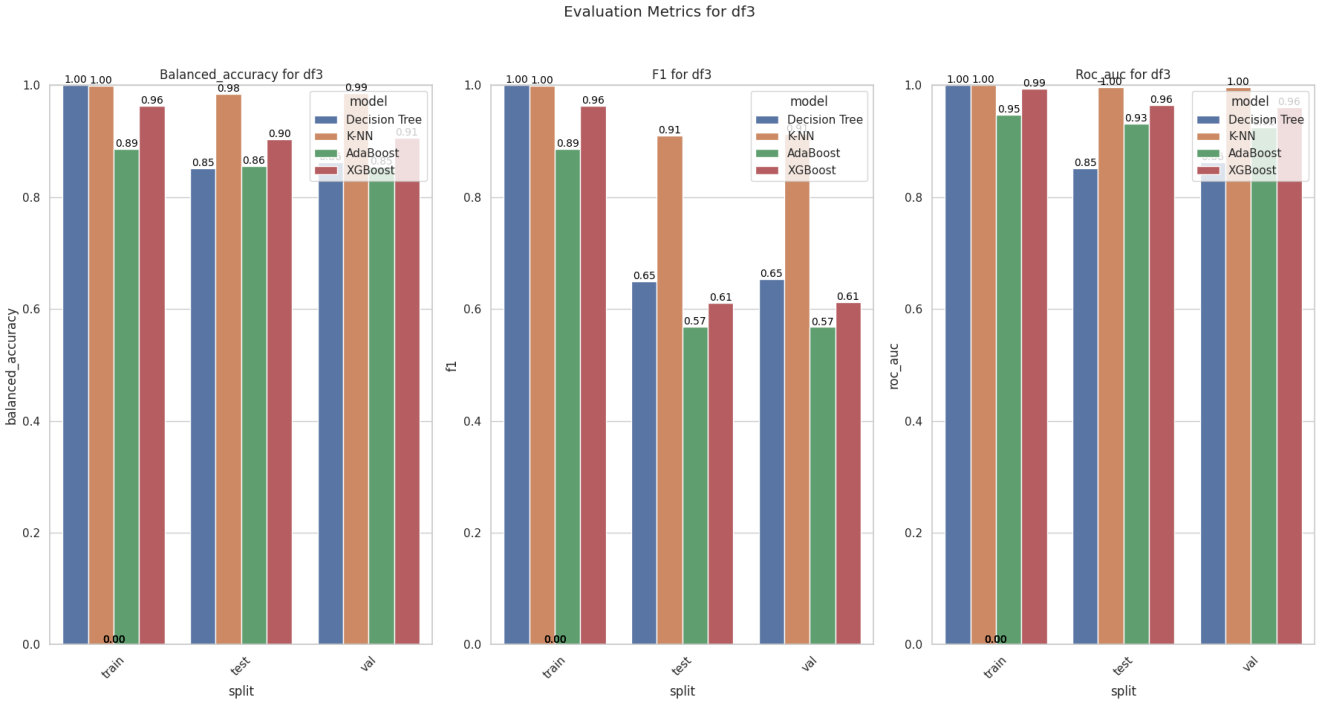

Figure 7: Model performances on SMOTE handled dataset3

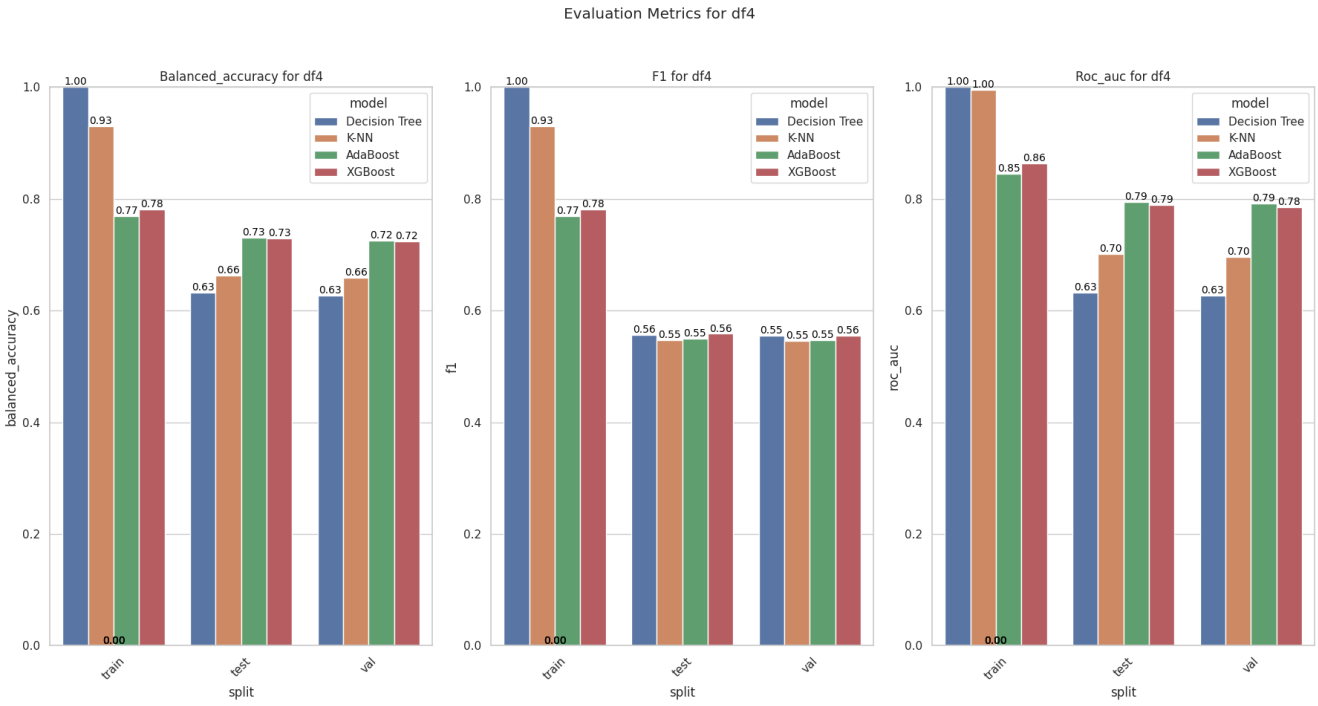

Figure 8: Model performances on SMOTE handled dataset4

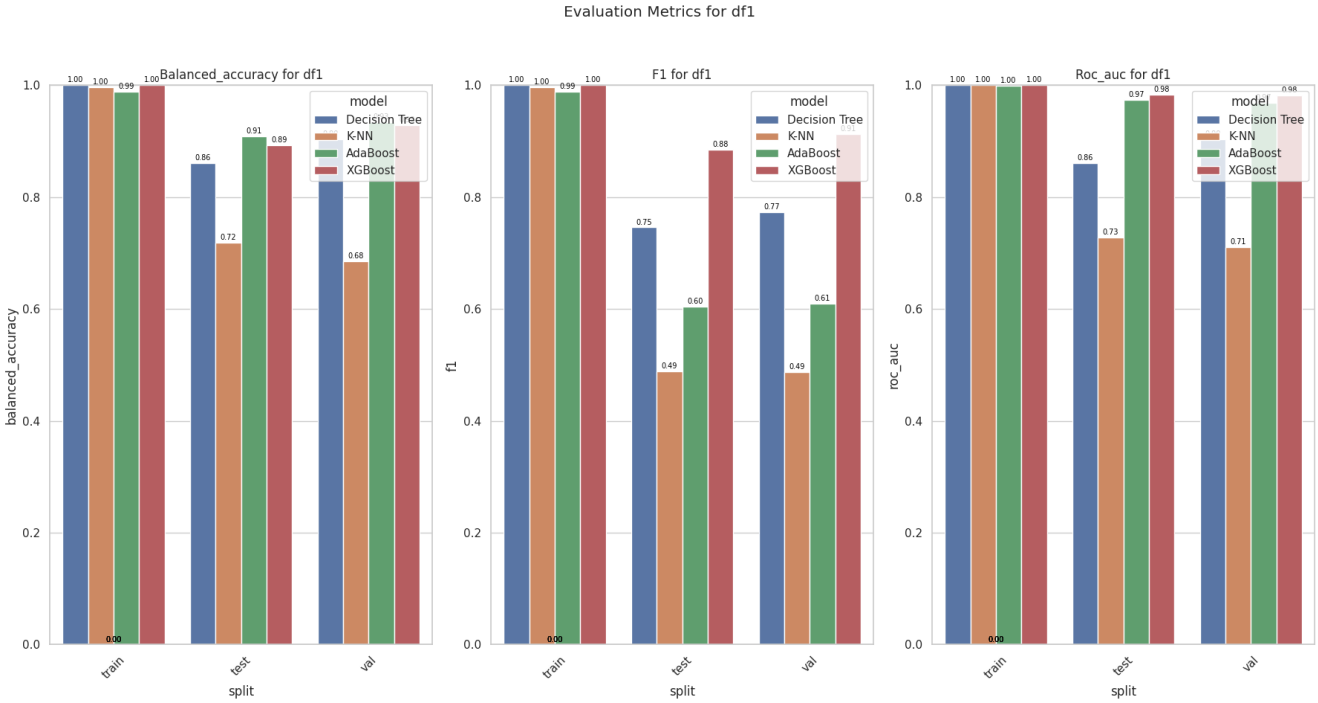

Figure 9: Model performances on SMOTE-EEN handled dataset1

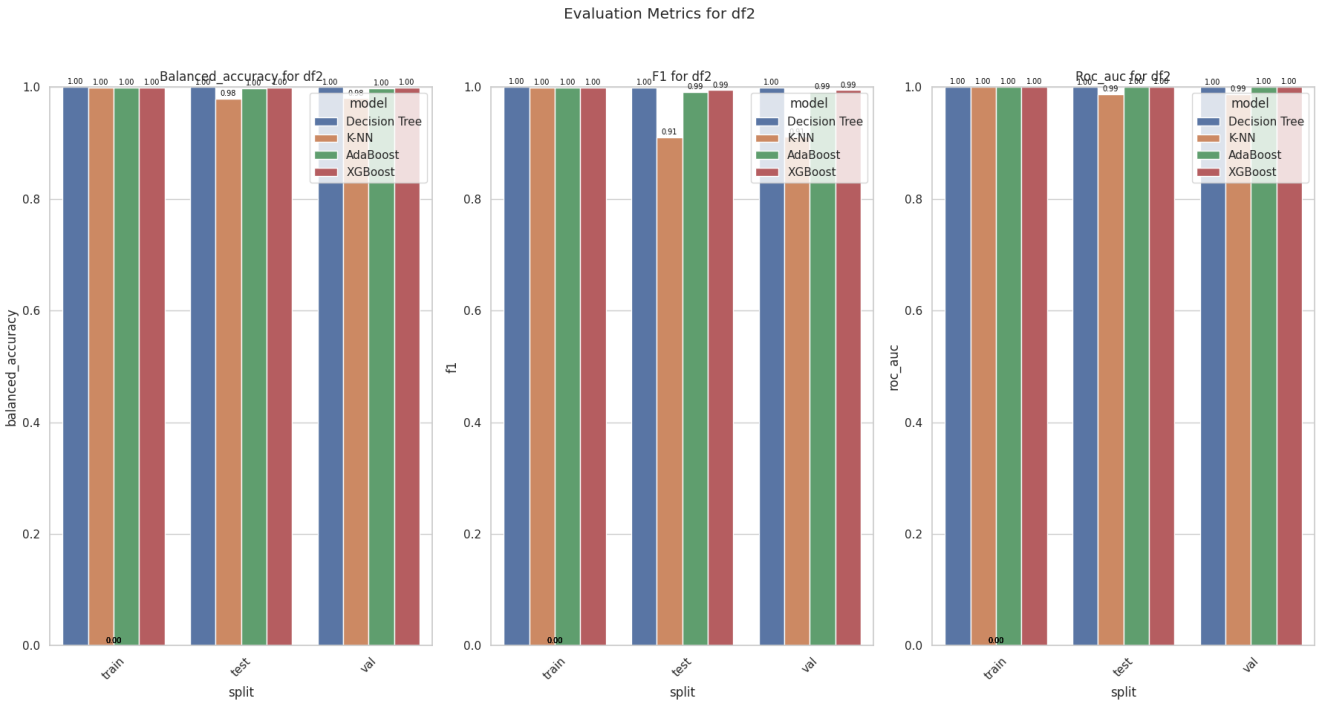

Figure 10: Model performances on SMOTE-EEN handled dataset2

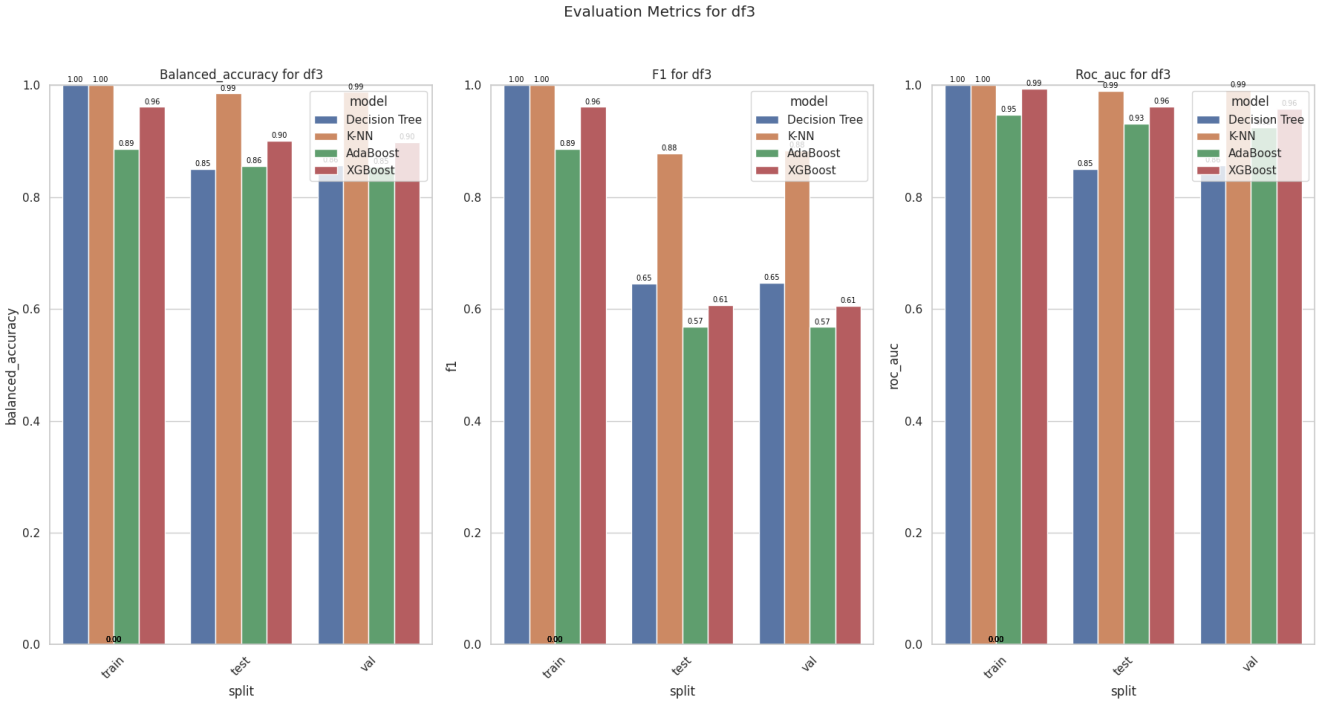

Figure 11: Model performances on SMOTE-EEN handled dataset3

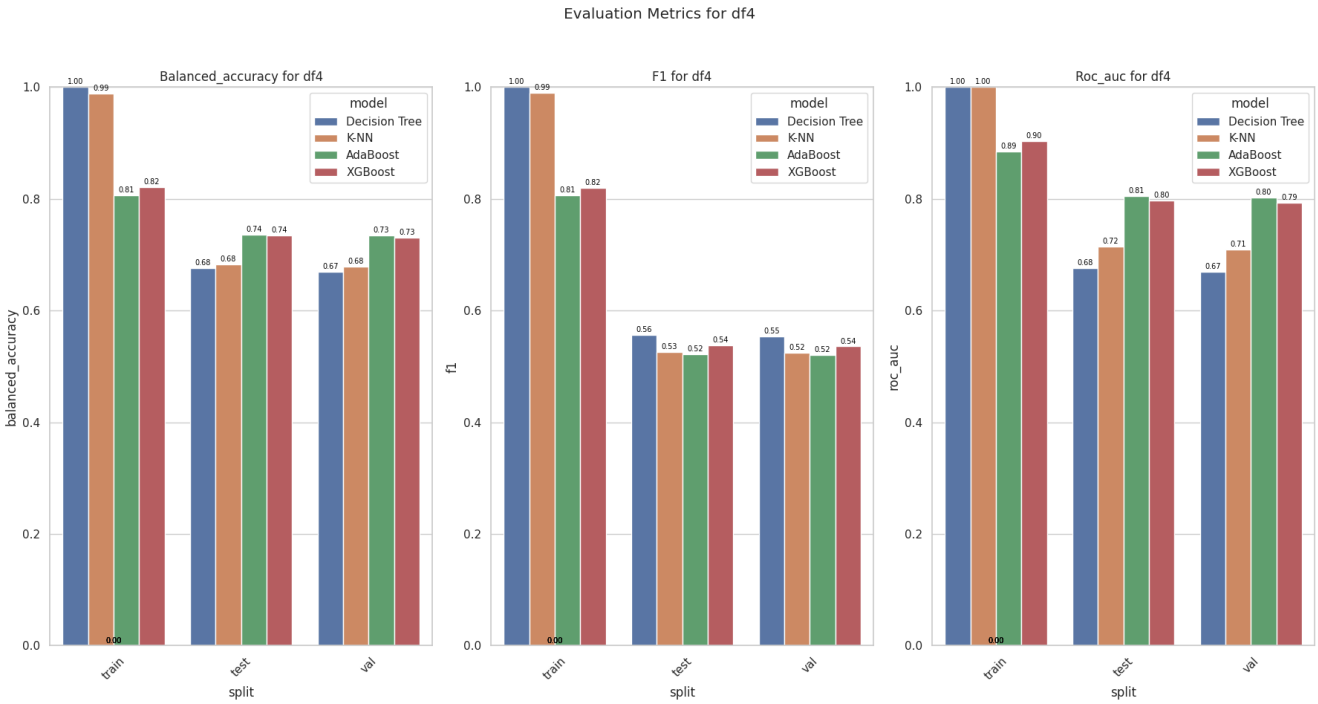

Figure 12: Model performances on SMOTE-EEN handled dataset4

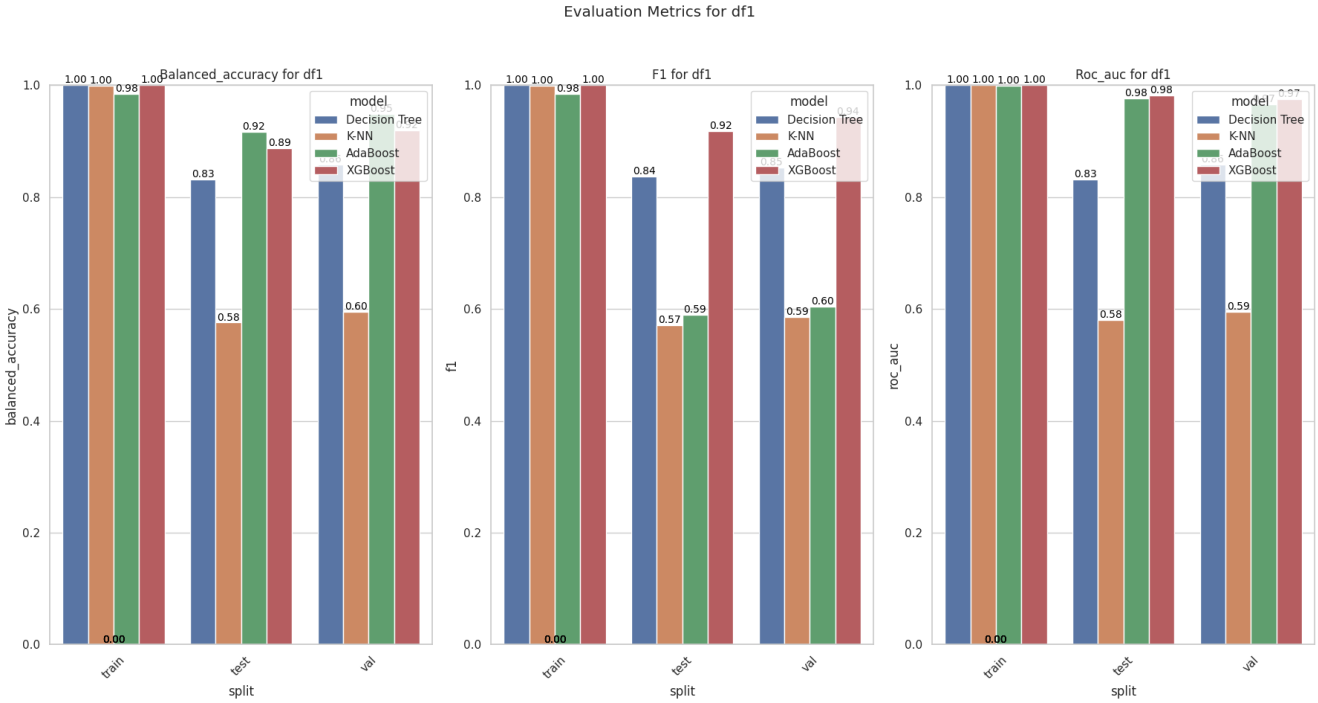

Figure 13: Model performances on Oversampling handled dataset1

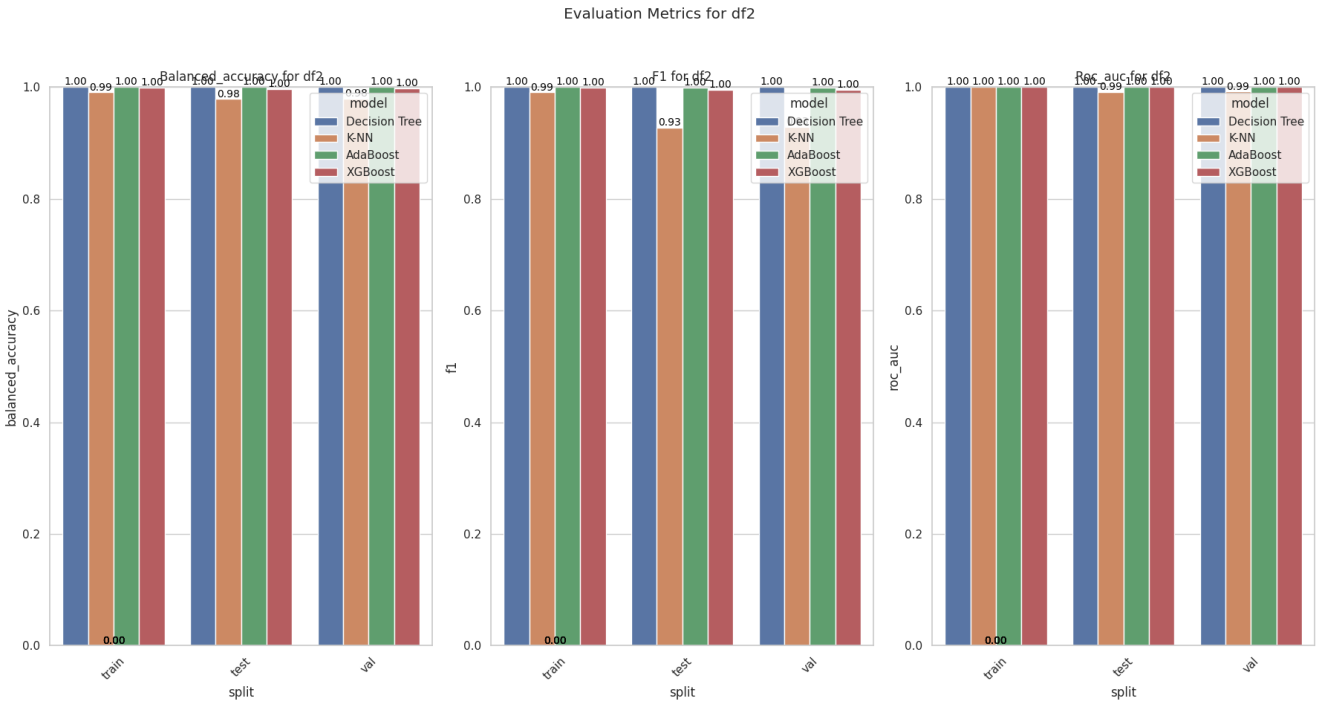

Figure 14: Model performances on Oversampling handled dataset2

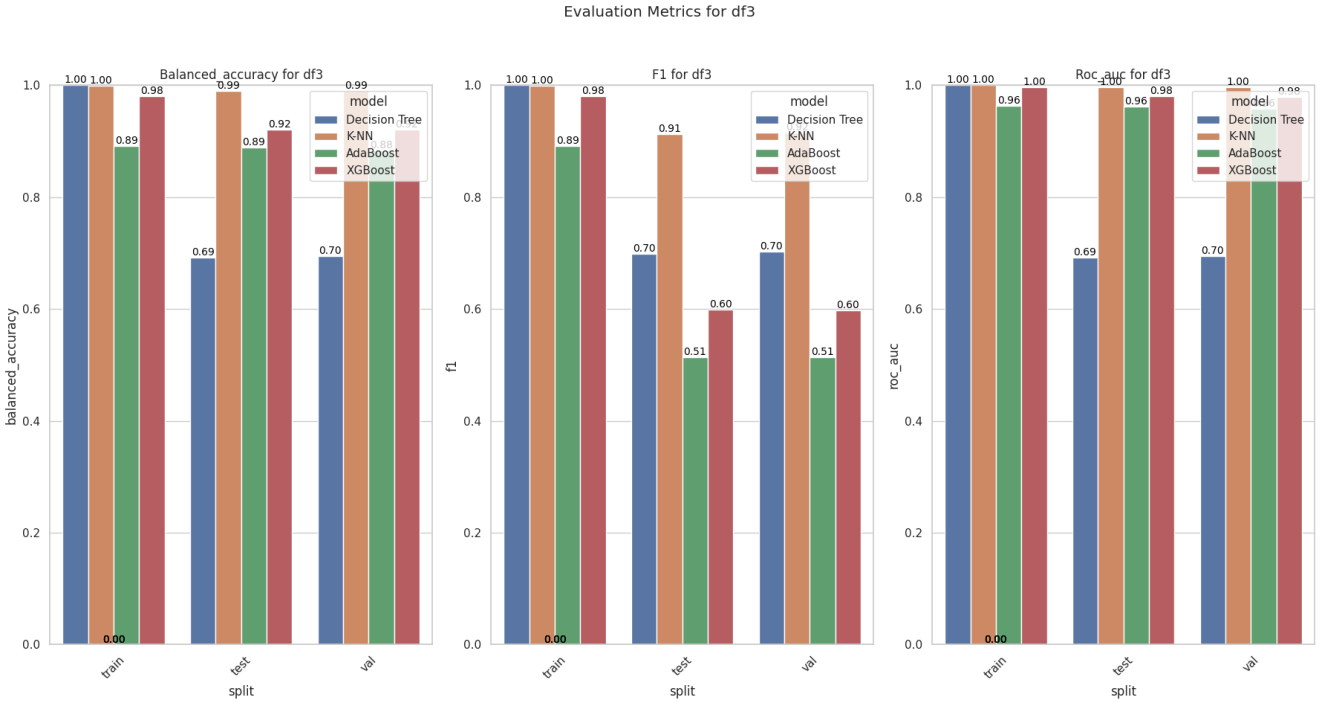

Figure 15: Model performances on Oversampling handled dataset3

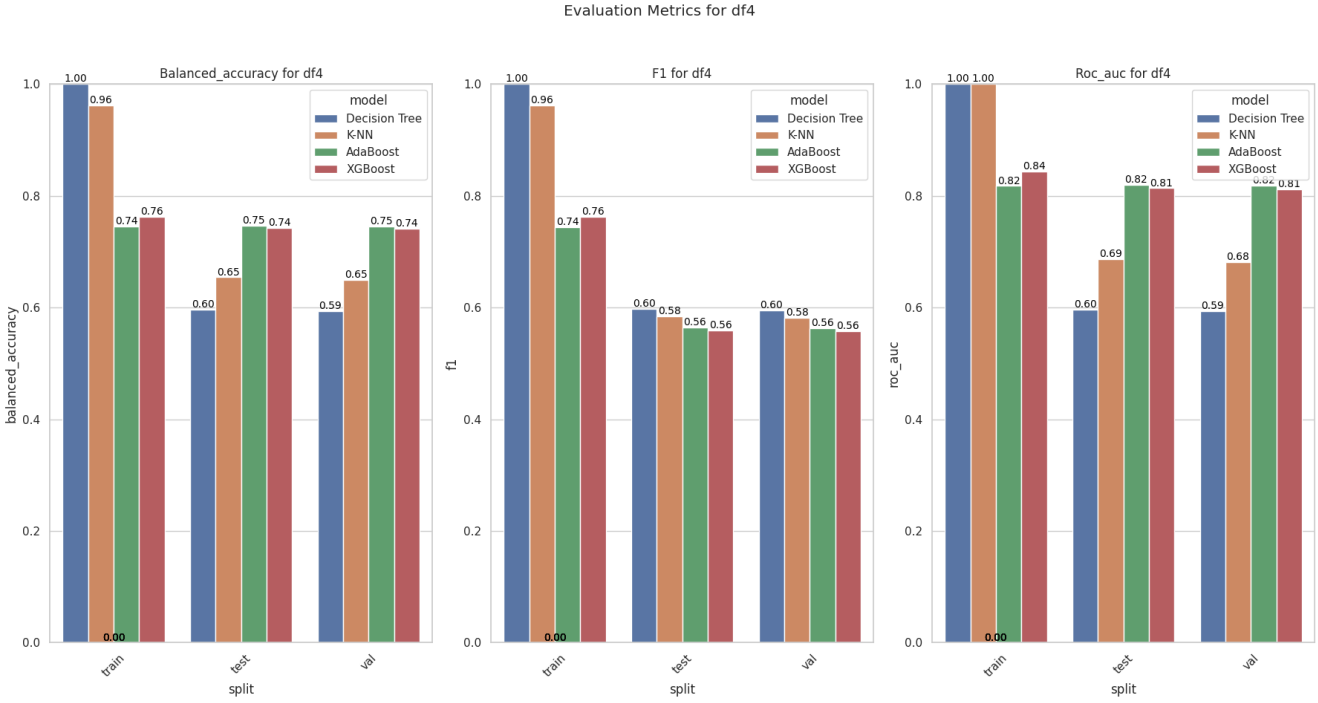

Figure 16: Model performances on Oversampling handled dataset4