THE AMERICAN UNIVERSITY IN CAIRO | SCHOOL OF SCIENCES AND ENGINEERING

DEPARTMENT OF MATHEMATICS AND ACTUARIAL SCIENCE

Data-Level Imbalance Handling Techniques: A Comparative Study on Credit Card Fraud Detection

Data Science Senior Project

Supervised by Dr Noha Youssef

Malak Gaballa - 900201683

Masa Tantawy - 900201312

Moustafa El Mahdy - 900201154

Table of Contents

Introduction 2
Methodology 3
Dataset 5
Machine Learning Algorithms 6
    Random Forest 6
    XGBoost 7
    LightGBM 7
Data Imbalance Handling Techniques 8
    Undersampling 8
        1. Random Undersampling (RUS) 9
        2. TomekLinks 9
    Oversampling 10
        1. Random Oversampling (ROS) 10
        2. SMOTE 10
    Over Sampling followed by Under Sampling 11
        1. SMOTE + Tomek (SmoteTomek) 11
        2. SMOTE + ENN (Smoteen) 12
Performance Measures 12
Data Preprocessing 16
Exploratory Data Analysis 18
Results 22
Discussion 25
Conclusion 29
References 31

Introduction

Living in the 21st century, technology is an essential part of our daily lives and plays a vital role in almost all financial procedures. With the ongoing concerns regarding sustainability along with the rise in fintech institutions and e-commerce, several communities are starting to adopt a cashless and contactless lifestyle allowing individuals to undergo financial transactions without carrying money, but instead using digital options such as cards or mobile wallets. However, with the rising number of transactions that occur due to the very large mass of users and their spread, misuse appears. Similar to traditional financial transactions, frauds are a critical concern when using these technologies; fraudulent transactions occur due to a stolen card, a card issued based on false information, or by the cardholder themselves who are bankrupt and thus unable to afford the payment (Singh et al., 2021). If a fraudulent transaction is not identified at an appropriate time, multiple casualties can occur to both the customer and the bank, which may be financial, reputational, or otherwise.

One of the most common and prevalent forms of financial technologies is credit cards as almost every individual above the age of 18 nowadays holds at least 1 credit or debit card. In fact, many countries are creating channels to allow those below this age to use one since banks are the main place to store money. With this in mind, this technology must be made very safe to use, meaning that the chance of fraudulent transactions must be minimised. There are two possible ways for that: first, trying to block a transaction that appears to be fraudulent before the money is transferred, which is known as fraud prevention; second, fraud detection is identifying successful fraud transactions that have occurred (Pozzolo & Bontempi, 2015). Conspicuously, technological advances have the capacity to operate on these massive numbers of transactions, unlike humans, and can be of higher accuracy. For this reason, the topic of credit card fraud detection has received a lot of attention in the field of machine learning; multiple researchers have used machine learning models to develop effective credit card fraud detection systems that can flag fraudulent transactions (Singh et al., 2021).

"Data-Level Imbalance Handling Techniques: A Comparative Study on Credit Card Fraud Detection," is a title that encapsulates the primary focus and scope of the research endeavor. In essence, the paper is going to delve into the realm of credit card fraud detection, an increasingly critical area in the financial sector given the rising sophistication of fraudulent activities. The term "Data-Level Imbalance Handling Techniques" refers to a suite of

methodologies employed to address the inherent class imbalance prevalent in fraud detection datasets, where instances of fraudulent transactions are often significantly outnumbered by legitimate transactions. This study seeks to investigate and compare various techniques utilized at the data level to tackle this imbalance effectively. By analyzing the effectiveness of different methods, the research aims to provide insights into the most suitable approaches for enhancing the accuracy and robustness of credit card fraud detection systems, thereby contributing to the ongoing efforts to strengthen financial security and integrity.

Methodology

Machine learning is an approach used to identify patterns and rules in a given data thus allowing predictions for unknown data points (Singh et al., 2021). This involves training the machine learning model on a set of data points known as the training set, then testing the model for evaluation on another set of points known as the test set. There are four main categories of machine learning: supervised, unsupervised, semi-supervised, and reinforcement learning. In supervised learning, the label of each data point is known during the training process, but absent in the testing process as it is trying to be predicted whereas in unsupervised learning, both datasets have no labels. When the dataset combines both labeled and unlabeled points, this is known as semi-supervised learning. As for reinforcement learning, the model is trained to maximise a set reward or goal in a certain situation. In the case of credit card fraud detection, supervised learning can be used to label each transaction as either fraudulent or legitimate, which depends on classification models, or unsupervised learning can be utilised since it can be considered an anomaly detection prediction where fraudulent transactions are unusual from other data points. In this paper, three classification models were tested, which are explained in **Chapter 1**, to develop a credit card fraud detection system.

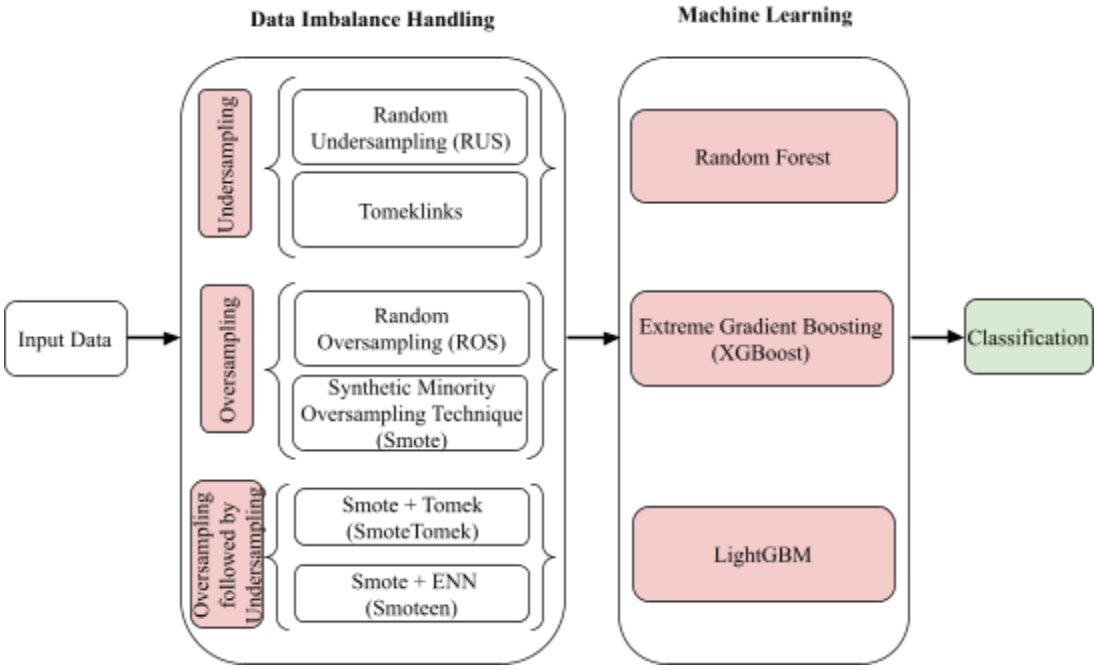
A major issue with the use of classification models in the case of credit card fraud detection is data imbalance (Singh et al., 2021). This happens when there are more instances, or data points, in the dataset with one label compared to the other; in other words, there are much more legitimate transactions compared to fraudulent ones since generally the latter have much fewer chances of happening compared to the former. As a result, the performance of the model is aggravated as it becomes biased towards the majority class thus detecting legitimate transactions more accurately than fraudulent ones. This contradicts the initial goal of the credit card fraud detection system where fraudulent transactions are of interest since it is less threatening for a

legitimate transaction to be flagged by the system than for a fraudulent transaction to go unnoticed. To overcome this issue, the extreme imbalance in the data structure needs to be reduced ideally until a model can be trained and tested on balanced data. There are multiple approaches to handle data imbalance; the data-level approaches used in this paper are discussed in **Chapter 2**.

This paper aims to compare multiple data imbalance handling techniques and machine learning models that yield an efficient credit card fraud detection system that can identify whether each transaction is legitimate or fraudulent. To handle data imbalance before model training, oversampling, undersampling, and oversampling followed by undersampling are used. Particularly, these are the techniques used:

1. Undersampling: Random Undersampling (RUS) and Tomeklinks
2. Oversampling: Random Oversampling (ROS) and Smote
3. Oversampling followed by Undersampling: Smote + Tomek (SmoteTomek) and Smote + ENN (Smoteen)

After the data is pre-processed, three supervised machine learning models are used: Random Forest, Extreme Gradient Boosting (XGBoost), and LightGBM. These models are evaluated based on a set of performance measures that are explained in **Chapter 3**. The diagram below describes the process of the credit card fraud detection system.



Dataset

The dataset used in this research paper consists of simulated credit card transactions completed over a two-year span, specifically from Jan 1st, 2019 to 31st December 2020. It consists of 555,719 transactional records and 22 features, excluding the target column where 1 represents a fraudulent transaction and 0 represents legitimate transactions. The dataset covers 1000 unique customers completing transactions with 800 different merchants.

Feature	Data Type	Description
id	Numeric - Discrete	Unique identifier for each transaction
Trans_date_trans_time	DateTime	Timestamp of the transaction (date and time)
Cc_num	Numeric - Discrete	Unique customer identification number
Merchant	String	Name of merchant involved in the transaction
Category	Categorical	Transaction type (e.g., personal, childcare)
Amt	Numeric - Continuous	Transaction amount
First	String	Cardholder's first name
Last	String	Cardholder's last name
Gender	Binary	Cardholder's gender
Street	String	Cardholder's street address
City	Categorical	Cardholder's city of residence
State	Categorical	Cardholder's state of residence
Zip	Categorical	Cardholder's zip code
Lat	Numeric - Continuous	Latitude of cardholder's location
Long	Numeric - Continuous	Longitude of cardholder's location
City_pop	Numeric - Discrete	Population of the cardholder's city
Job	Categorical	Cardholder's job title
Dob	Date	Cardholder's date of birth

Trans_num	String	Unique transaction identifier.
Unix_time	Numeric - Discrete	Transaction timestamp (Unix format)
Merch_lat	Numeric - Continuous	Latitude of merchant
Merch_long	Numeric - Continuous	Longitude of merchant
Is_fraud	Binary	Fraudulent transaction indicator (1 = fraud, 0 = legitimate)

Machine Learning Algorithms

In recent years, there has been a shift in the field of credit card fraud detection. Individual classification models, which were previously considered highly accurate and reliable, are now being recognized as less effective in addressing the complexities of modern data. Challenges such as complex data structures, high-dimensional features, and class imbalance have limited the accuracy of these models. As a result, there has been a growing interest in using ensemble classifiers, which combine the predictions of multiple models to improve overall performance. Ensemble classifiers leverage the strengths of different models and mitigate their weaknesses, leading to more robust and accurate fraud detection. This shift towards ensemble classifiers reflects the understanding that collaborative decision-making can yield better results in the context of credit card fraud detection, where accuracy and reliability are crucial. Since the problem of credit card fraud detection is a binary classification one, logistic regression was selected as a baseline mode for comparison against more advanced models. Based on previous research, it has been observed that the commonly used ensemble classifiers are Random Forest, Extreme Gradient Boost (XGBoost), and LightGBM. Accordingly, the 3 aforementioned models in addition to logistic regression will be used in the comparative study.

Random Forest

Random Forest is an ensemble, tree-based machine-learning method that is primarily used for classification and regression (Singh & Jain, 2022). It is a collection of decision trees, with each tree generated using a random subset of observations and features. With each tree having different observations and features, the model is diversified. For the final prediction, the outputs of all trees are combined, and the final output is decided using a majority vote. Because of its nature, the random forest classifier prevents over-fitting, handles an existing class

imbalance or high dimensional feature spaces, and is robust against outliers (Singh et al., 2021). Given the classifier's ability to handle imbalanced datasets, random forest is a popular choice for credit card fraud detection (Chung & Lee, 2023). Among three examined papers, the highest performance measures were yielded when the random forest classifier was employed (Singh et al., 2021; Singh & Jain, 2022; Salekshahrezaee et al., 2023).

XGBoost

XGBoost, short for Extreme Gradient Boosting, is a powerful machine learning algorithm that has gained significant popularity in recent years. It is particularly well-suited for classification tasks, including credit card fraud detection. XGBoost is an ensemble method that combines the predictions of multiple weak learners, typically decision trees, to create a strong predictive model (Salekshahrezaee et al., 2023). It leverages gradient boosting, which is a technique that trains subsequent models to correct the errors made by the previous models in the ensemble. This iterative process allows XGBoost to gradually improve its performance and make more accurate predictions. In credit card fraud detection, XGBoost has demonstrated impressive performance, often outperforming other algorithms (Singh et al., 2021). It can effectively deal with imbalanced datasets, where fraudulent transactions are rare compared to legitimate ones. XGBoost's ability to handle high-dimensional data, capture complex relationships, and provide feature importance analysis makes it a popular choice for credit card fraud detection tasks, delivering robust and accurate results.

LightGBM

LightGBM is a high-performance gradient-boosting framework that has gained popularity in machine learning applications, including credit card fraud detection. It is designed to be efficient and scalable, making it well-suited for handling large datasets with high-dimensional features. LightGBM employs a unique approach called the Gradient-based One-Side Sampling (GOSS) technique, which focuses on reducing the number of attributes through the categorization of mutually exclusive features and selecting the most informative instances during the training process, significantly reducing computational costs (Salekshahrezaee et al., 2023). This technique, along with other optimizations, enables LightGBM to train models faster while maintaining high accuracy. In the context of credit card fraud detection, LightGBM has demonstrated strong performance, often achieving competitive results such as an F1 Score of 0.909 compared to other algorithms in the study conducted by

Salekshahrezaee et al. (2023). It can effectively handle imbalanced datasets, capture complex patterns, train data faster, consume less memory, process large-scale data, and provide insights into feature importance (Bakhtiari et al., 2023). LightGBM's combination of speed, accuracy, and scalability makes it a popular choice for credit card fraud detection, enabling efficient and effective identification of fraudulent transactions.

Data Imbalance Handling Techniques

There are multiple approaches to deal with data imbalance that have been used in several domains. As explained earlier, in an imbalanced dataset, the number of instances or examples of a single class significantly outweighs the number of instances of another class, which can act as a challenge when applying ML algorithms since they may become biased toward the majority class, leading to poor performance on the minority class. Generally, the imbalanced class problem can be tackled through one of four approaches: data-level, algorithm-level, cost-sensitive and classifiers ensembling approach. The techniques used in this paper fall under the data-level approach, which relies on handling imbalance during the data pre-processing data, before the model is trained. It is also known as the external approach. The algorithm-level, on the other hand, is known as the internal approach where the classification algorithm itself is modified to handle imbalance such as by assigning higher penalties for misclassifying minority class instances or adjusting the decision threshold. As for the cost-sensitive approach, this is a hybrid of the internal and external approaches, and the classifier's ensemble approach combines multiple classifiers into one model using techniques such as bagging and boosting to accommodate for the class imbalance (Singh et al., 2021).

Undersampling

One approach to address data imbalance is undersampling. This approach relies on eliminating members of the majority class to afford a balanced distribution of classes in the newly created dataset (Singh et al., 2021). There are two forms of undersampling: Random undersampling and Informative undersampling. The former randomly eliminates points from the majority class, as implied by the name, until the dataset is balanced, and the latter uses a predefined rule to select points that will be eliminated from the majority class (Singh et al., 2021). One of each of these two forms is used in this paper, which is described as follows:

1. Random Undersampling (RUS)

Random Undersampling (RUS) is one of the most commonly used data sampling techniques for imbalanced data (Singh et al., 2021). Random points from the majority class are selected and eliminated from the dataset. This happens until the ratio between the majority and minority classes reaches a set level, such as 1:1 or 1:20 (Salekshahrezaee et al., 2023). While this makes RUS relatively fast and useful for large datasets, a major problem with this technique is that it may lead to the loss of valuable information due to the deletion of important points (Salekshahrezaee et al., 2023).

2. TomekLinks

Another technique for undersampling is TomekLinks, which is a form of "label noise filter" (Salekshahrezaee et al., 2023, p.3). A Tomeklink is defined by two points from the different classes, such that these two points are the closest to each other using a distance metric *d*. In other words, for a pair of points *x* and *y* from opposing classes, there exists no point *z* such that *d(x,z)* or *d(y,z)* is lower than *d(x,y)*. This means that one of these two points is noise or both are borderline; as a result, "these cross-class pairs are valuable in defining the class boundary." (Salekshahrezaee et al., 2023, p.3). Once these links have been identified, as shown in Figure 1 below, elements of the majority class where Tomeklinks exist are eliminated. It is important to note that tomeklinks does not completely balance the dataset in the case of extreme data imbalance, namely credit card fraud detection, since its goal is to enhance the separability among the classes by widening the gap between them rather than completely balancing the data.

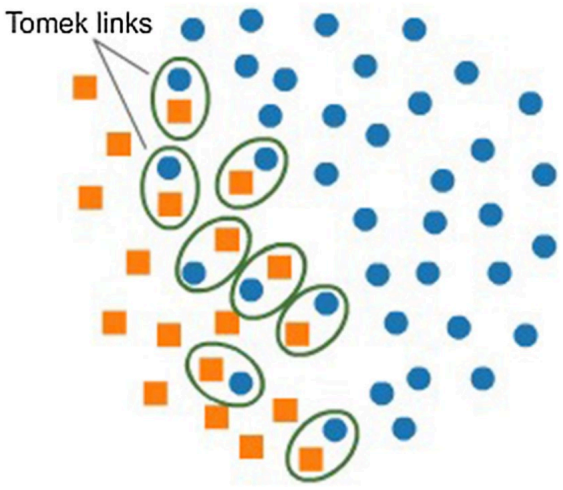


Figure 1: Tomek Links

Oversampling

Oversampling is a technique used in machine learning and data analysis to address the issue of imbalanced datasets. Also known as upsampling, this is a sampling technique, which balances the dataset by replicating the minority class samples (increasing the number of instances of the minority class) by generating synthetic examples or replicating existing examples. The goal is to balance the class distribution in the dataset, allowing the machine learning algorithm to better learn the patterns of the minority class. The oversampling method creates the superset without losing the original data, but it may lead to an overfitting problem and more complex computational computing when the size of the data is large enough (Chawla et al., 2002). There are myriads of oversampling techniques and in our research, we will be tackling two of these techniques which are:

1. Random Oversampling (ROS)

Random oversampling is an effective and widely used oversampling approach to handle the class imbalance problem. It is the process of randomly selecting minority class samples and replicating them up to the point where we achieve a more balanced dataset. The random oversampling method creates the same new data points from the original dataset which can increase the possibility of overfitting of the model. Let us assume a dataset S has a class imbalance problem consisting of the minority class samples, Smin, and majority class samples, Smax (Chawla et al., 2002). Let A be the randomly generated replicated data points of minority class samples then the number of samples in the new dataset is given as:

$$|S| = S_{max} + (S_{min} + |A|)$$

2. SMOTE

SMOTE, Synthetic Minority Oversampling Technique, is a popular oversampling method for an imbalanced dataset in which new synthetic samples are generated by interpolating between existing, adjacent minority class instances to balance different classes (Singh et al., 2021). It generates synthetic instances along the line segments connecting existing minority class instances in the feature space without causing overfitting, yet it may cause overgeneralization because it generates synthetic minority class data points randomly without considering majority class samples (Chawla et al., 2002). This interpolation of the synthetic data points is generated using minority class data instances and the K-Nearest Neighbours (KNN) method. Based on the requirement of synthetic data points, the KNN algorithm selects minority class instances and

interpolates between all K data points, and plots new data points randomly. According to the requirement of the number of synthetic data points, these points are added to the original data, as shown in Figure 2 below.

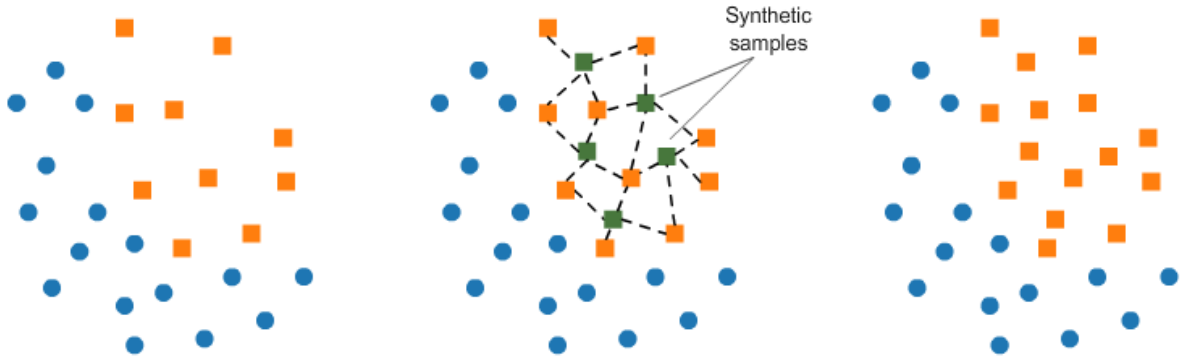


Figure 2: SMOTE

The new synthetic data point, A, created using the Smote method is defined as:

$$A = X + c(X_R + X)$$

Where X is a data point in the dataset, X_R is the randomly selected data-point among the K nearest neighbours of X and c ($0 \leq c \leq 1$) is a constant. (Chawla et al., 2002)

Over Sampling followed by Under Sampling

Both the techniques, oversampling and undersampling, discussed above have some limitations and drawbacks. For instance, oversampling works well for the small dataset, and the model build is trained on balanced data that tends to overfit. In contrast, undersampling works well if we have a large dataset, and there is a chance of losing valuable information (Chawla et al., 2002). Over Sampling followed by Under Sampling is a combination strategy that tackles class imbalance in a dataset. This approach involves applying oversampling to the minority class to increase its representation in the dataset and then undersampling the majority class to reduce its dominance to reach a more balanced dataset. There are various oversampling followed by undersampling techniques and in our research we will experiment with two of these techniques which are:

1. SMOTE + Tomek (SmoteTomek)

SMOTE Tomek is a combination of two techniques: SMOTE and Tomek links where Tomek links are pairs of instances from different classes that are close to each other, and removing these pairs can help to clarify the decision boundary between both classes. SMOTE

Tomek combines the oversampling capability of SMOTE with the undersampling technique of Tomek links to enhance the quality of the dataset for training a machine learning model. First, the imbalance dataset is oversampled using the Smote technique, and then Tomeklinks are identified and removed from the oversampled dataset. As we know, the Smote can lead to overfitting the model and this overfitting can be removed by the Tomeklinks approach. Using the Smote followed by the Tomeklink approach, the dataset can be perfectly balanced, and also the performance of the classifiers can be improved (Chawla et al., 2002).

2. SMOTE + ENN (Smoteen)

SMOTE-ENN is a combination of two techniques: SMOTE for oversampling the minority class and Edited Nearest Neighbour (ENN) for cleaning the dataset by removing certain instances. SMOTEEN combines the oversampling capability of SMOTE with the undersampling technique of ENN which enhances the quality of the data by creating a more balanced and informative dataset. Smoteen is a very effective technique in which Edited Nearest Neighbour (ENN) will remove all misclassified dataset instances that differ from their neighbourhood. The nearest neighbour is evaluated on predefined criteria. If the sample does not fulfill the criteria, these samples will be removed else not. Now, on the resampled dataset, the Smote technique will make it a perfectly balanced dataset by creating synthetic data points (Chawla et al., 2002).

Performance Measures

As explained earlier, machine learning models are developed in two stages: training and testing. During the model testing, each model's performance is evaluated on the same dataset to ensure fair evaluation. In the case of credit card fraud detection, the metrics used must account for the imbalanced nature of the data. Although accuracy is a common metric, it is not suitable for use in the case of class imbalance. Particularly, since fraudulent transactions are of interest, it is more important for the models to correctly predict fraudulent transactions compared to legitimate ones. In other words, it is critical that the model is unable to miss any fraud among the transactions and correctly identify it rather than labelling a legitimate transaction as fraudulent. Hence, the selected metrics that focus on these objectives are precision, recall, and F1 score. Each of these is described below in addition to the elements used, which are shown in the confusion matrix in Table 1.

Table 1: Confusion Matrix

	Actual Value		
Predicted Values		**Positive (1)**	**Negative (0)**
	Positive (1)	True Positive (TP)	False Positive (FP)
	Negative (0)	False Negative (FN)	True Negative (TN)

- **TP:** The number of data points with actual labels as positive and predicted as positive.
- **TN:** The number of data points with actual labels as negative and predicted as negative.
- **FP:** The number of data points with actual labels as negative and predicted as positive; this is known as type 1 error.
- **FN:** The number of data points with actual labels as positive and predicted as negative; this is known as type 2 error.

The recall, precision, and F1-score all range from 0 to 1, where higher values indicate better model performance in all the metrics.

- **Recall:** This is the ratio, known as true positive rate (TPR), of correctly predicted positive points out of all actual positive points in the dataset.

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

- **Precision:** This is the ratio, known as false positive rate (FPR), of correctly predicted positive points out of all predicted positive points in the dataset.

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

- **F1-Score:** This is a weighted average of both the recall and precision and is useful when one of these two metrics is high and the other is low. In such scenarios, the F1-score takes both into account while punishing extreme values since it uses the harmonic mean and not the arithmetic mean.

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

To select the best performing model, evaluation took place over 2 stages. First, the data was split into a train-test split in an 80%-20% ratio respectively, preserving the same imbalance ratio in both datasets. This ensures that all the different models are evaluated on the same data for fair comparison. The testing dataset has 111,144 transactions and 41 columns, and it exhibits

extreme imbalance, as only 0.386% of transactions are labelled as fraudulent; this is because imbalance handling techniques have not been applied to it. Accordingly, using the accuracy metric is unsuitable thus the precision, recall, and F1 score will be used as they are appropriate in this case as explained earlier. As for the training data, there are 6 different versions on which the models are trained for each data imbalance handling approach: rus, tomeklinks, ros, smote, smotetomek, and smoteen. The original imbalanced training dataset has 444,575 transactions and 41 columns. After imbalance handling, the 6 yielded training datasets have a size and an imbalance ratio as shown in Table 2. The datasets handled using rus, ros, smote, and smotetomek are perfectly balanced while smoteen is almost balanced. On the other hand, tomeklinks preserved the same imbalance ratio as the original data but reduced the number of transactions in order to enhance the class separability as explained earlier. The details of the class ratios and dataset sizes are shown in Table 2 and Figures 3, 4,5,6,7,8.

Table 2: Train-Test Data Description

Dataset	***Size*** *(transactions)*	***Imbalance Ratio***
Original Dataset	555,719	99.614% : 0.386%
Random Undersampling	3,432	50% : 50%
TomekLinks	444,338	99.614% : 0.386%
Random Oversampling	885,718	50% : 50%
SMOTE	885,718	50% : 50%
SmoteTomek	885,718	50% : 50%
Smoteen	882,740	49.831% : 50.169%
Test Set	111,144	99.614% : 0.386%

Figure 3: Original Data

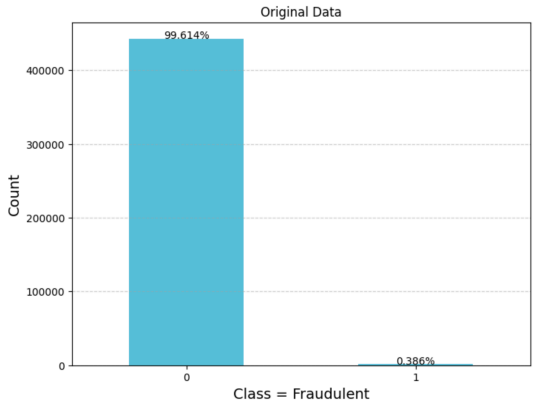


Figure 4: Random Under Sampling Training Data

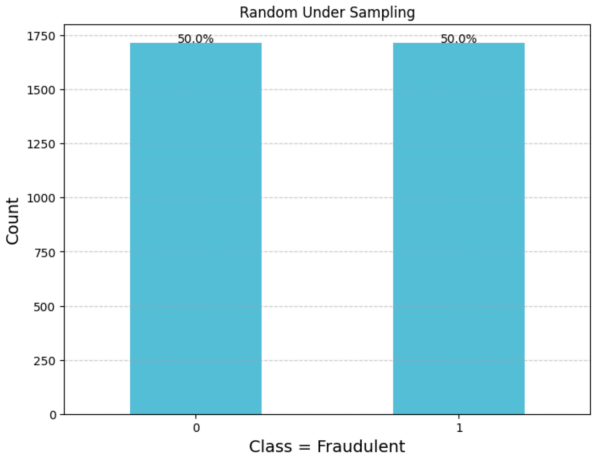


Figure 5: TomekLinks Training Data

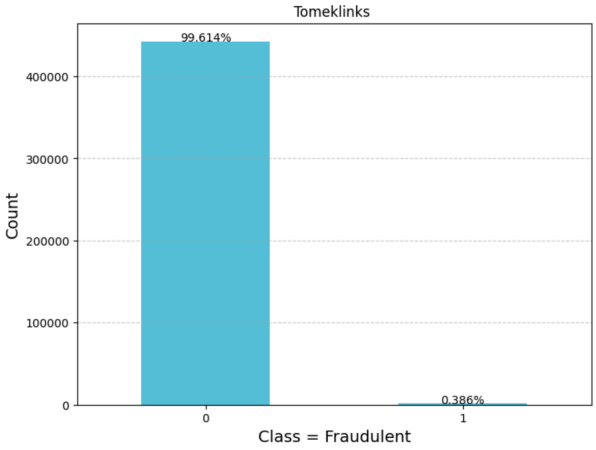


Figure 6: Random Over Sampling Training Data

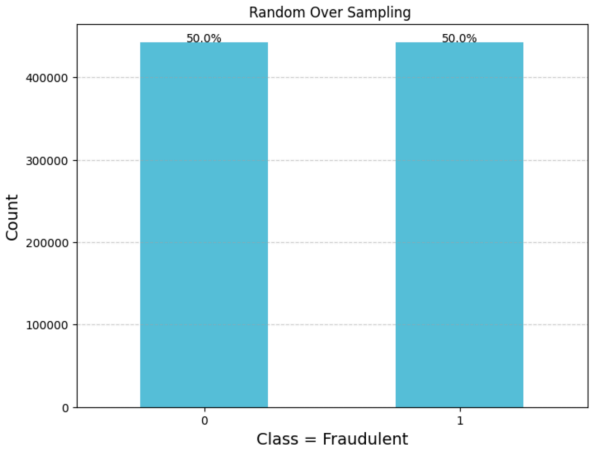


Figure 7: SMOTE Training Data

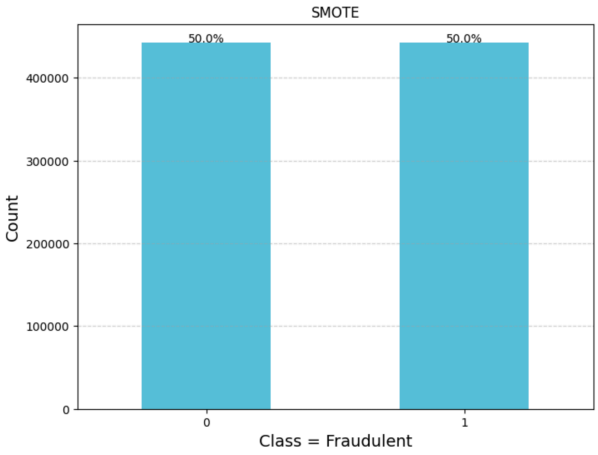


Figure 9: SmoteTomek Training Data

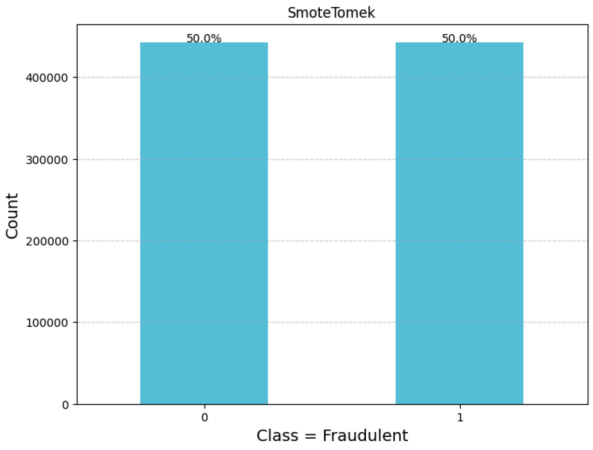
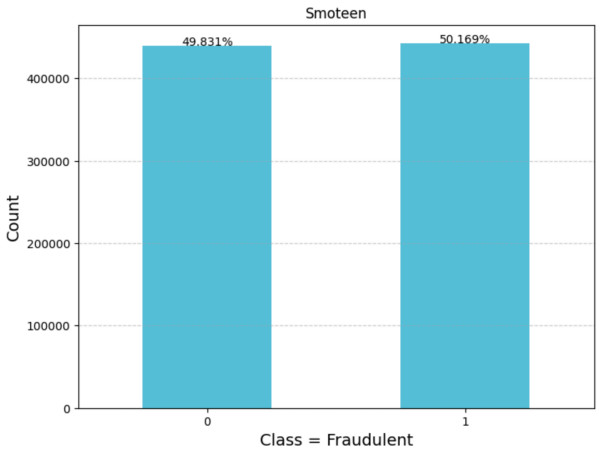


Figure 10: Smoteen Training Data



Second, the models were evaluated based on 5-fold cross-validation; this is after acknowledging that evaluation using train-test split does not take into account whether each model overfits or not. Thus, k-fold cross-validation using k=5 was used to determine the performance of the models. Using each data imbalance handling technique, the original imbalance dataset was adjusted, yielding the same ratios of imbalance as in the train-test split approach shown in Table X and the figures above given a larger dataset size compared to the 80% split.

Data Preprocessing

Preparing the dataset for analysis and modelling involved several preprocessing steps. These include dealing with categorical data, splitting the data, and handling data imbalance.

Primarily, the columns *id, cc_num, merchant, first, last, street, city, zip, trans_num,* and *unix_time* were dropped since they're insignificant and will not affect the classification decision. Next, the feature *trans_date_trans_time* column was converted to a datetime format, allowing for the extraction of the temporal features *year, day, month,* and *hour*. The column *year* was dropped since all the transactions took place in 2020, and the column *day* was used to extract the *weekday* before being dropped. Similarly, the cardholder's *age* was created using the *dob* (date of birth) column then the latter was dropped. As for the columns *lat, long, merch_lat,* and *merch_long*, they were used to extract the *distance* in km between the location of the merchant and the transaction before being dropped.

For easy maneuvering of categorical columns, the column *category* was grouped into 14 categories such as entertainment, food and dining, and gas transport. The categories of *job* were

also grouped into the 9 industries: business, creative, education, engineering and technical, healthcare, legal, media, social services, and entertainment. Likewise, the *state* column categories were grouped into the 4 categories north, midwest, south, and west. Lastly, the numeric columns which are *month, hour, age, distance, amt,* and *city pop* were scaled and the final categorical columns which are *weekday, category, gender, state,* and *job* were one-hot encoded. For the column *gender,* the value 0 indicated the class male and the value 1 indicated female.

After these preprocessing steps, the final dataset consisted of 555,719 records and 42 columns, including the target variable. These steps were essential to enhance the dataset's usability and readiness for subsequent analysis and machine learning tasks, ensuring all features were appropriately formatted and encoded.

Exploratory Data Analysis

Exploratory data analysis (EDA) was conducted in two phases, before and after preprocessing, to extract insights specifically related to fraudulent transactions and to analyze the distributions and correlations of the dataset's features. The initial EDA phase involved exploring raw data to identify potential patterns and anomalies indicative of fraudulent activities. Subsequently, preprocessing steps were applied to address data quality issues. Following preprocessing, a second round of EDA was performed to evaluate the impact of preprocessing on feature distributions and correlations, as well as to refine insights into fraudulent transactions. This sequential approach ensured a thorough understanding of the dataset, enabling the extraction of precise insights and the formulation of targeted strategies to combat fraudulent behavior.

It was crucial to begin by examining the distributions of key numerical features, such as transaction amount and city population. Both distributions exhibited positive skewness, characterized by a concentration of data towards lower values with a longer tail extending towards higher values. This skewness suggests that the majority of transactions or city populations tend to be clustered at lower values, with fewer occurrences of exceptionally high values. In the case of transaction amount, this skewness may indicate a prevalence of smaller transactions, with only a minority of transactions accounting for larger amounts. Similarly, the positive skewness observed in city population distribution may imply that most cities in the dataset have relatively smaller populations, while a few cities exhibit significantly larger populations. This skewness highlights the necessity of standardizing numerical features, as it ensures that the data is on a comparable scale and mitigates the impact of outliers, facilitating more accurate analyses and model performance assessments.

Figure 11 : TRX Amount Distribution

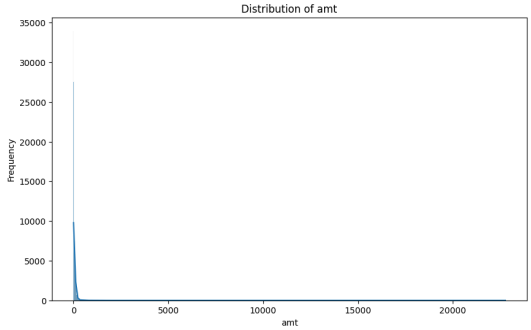
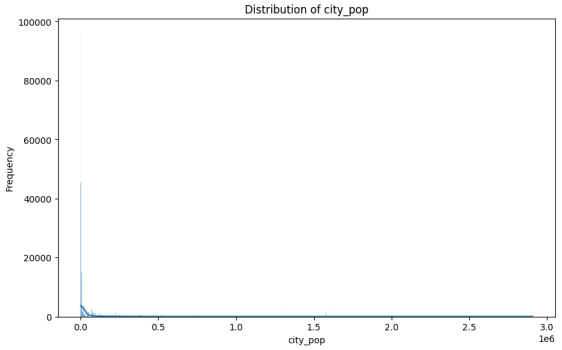
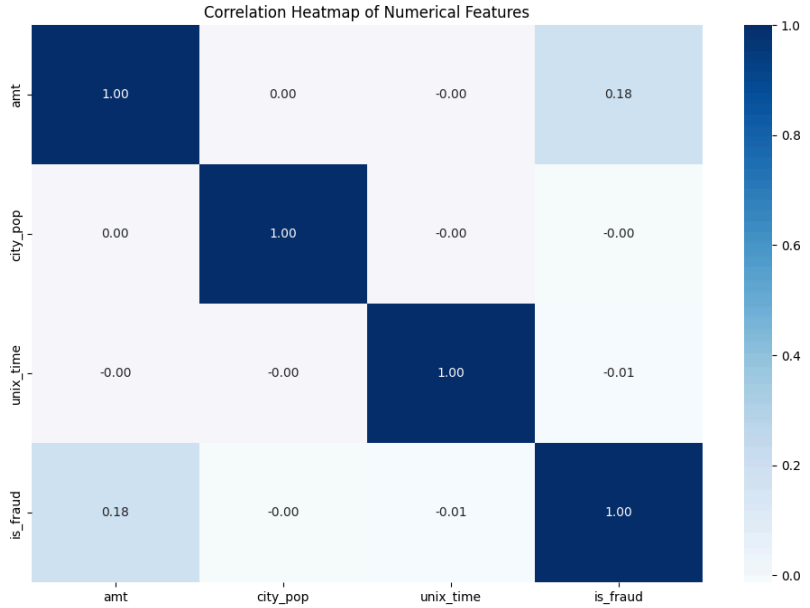


Figure 12 : City Population Distribution



Next, it was crucial to look at the correlations between numerical values. Since most features are categorical, the correlation matrix only included four variables, including the target feature. Interestingly, there was little correlation among the predictors. However, a notable discovery was the slight positive correlation between the target variable and transaction amount. This suggests that higher transaction amounts may have a higher chance of being classified as fraudulent. Understanding these relationships is vital for further analysis and model building.

Figure 13 : Correlation Heatmap



Analyzing the nominal features was equally important. The transaction category revealed that the majority of transactions were for gas or transportation, grocery shopping, home appliances, and physical as well as online shopping. Regarding cardholder occupations, a large portion of customers worked as video editors, exhibition designers, architects, or engineers. Additionally, most cardholders resided in Texas, New York, or Pennsylvania. These insights into transaction categories, cardholder occupations, and geographic distributions provide valuable context for understanding consumer behavior and preferences within the dataset.

Figure 14: TRX Type Frequency Plot

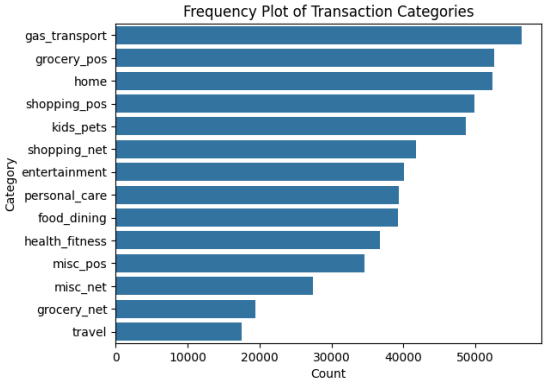


Figure 15: Jobs Frequency Plot

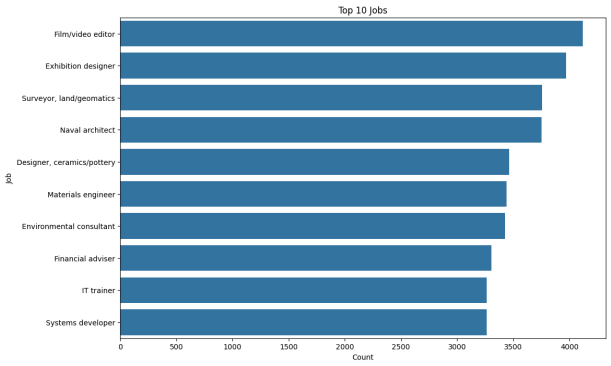
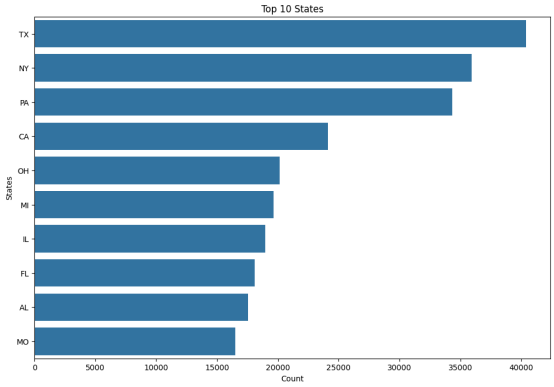


Figure 16: States Frequency Plot



Analyzing counts of fraudulent transactions by various features allows for a precise understanding of the distribution and characteristics of fraudulent activities within the dataset. Gender-wise, females accounted for 1164 fraudulent transactions, while males accounted for 981, indicating potential gender-specific trends in fraud. Transaction category analysis reveals online shopping and physical grocery shopping as the most common categories associated with fraud, with 506 and 485 fraudulent transactions, respectively. Occupations such as science writers and licensed conveyancers are among the most frequent job titles linked to fraudulent transactions. Geographically, New York, Pennsylvania, and Texas stand out as states with the highest counts of fraudulent transactions, with 175, 114, and 113 fraudulent transactions, respectively. These insights provide targeted information for effective fraud prevention strategies.

After preprocessing the data, additional EDA was conducted. The analysis of transaction counts by month revealed that August (8) had the highest number of fraudulent transactions, with 415 occurrences, followed closely by October (10) with 384 transactions, and September (9) with 340 transactions. Conversely, June (6) had the lowest count of fraudulent transactions, with

only 133 occurrences. Regarding the hour of the day, fraudulent activities peaked between 22:00 and 23:00, with 550 and 538 fraudulent transactions, respectively. A notable drop in fraudulent transactions was observed during the early morning hours, with only 194 occurrences at 03:00. These findings offer insights into temporal trends in fraudulent activities, highlighting potential periods of heightened risk for fraudulent transactions.

When examining fraudulent transactions by region, it became apparent that the Southern region experienced a higher frequency of fraud compared to other regions, followed by the Midwest, North, and West. This suggests a regional disparity in fraudulent activity, with the Southern region being particularly susceptible. In terms of days of the week, fraudulent transactions were most commonly observed on Mondays, Tuesdays, and Sundays. This temporal pattern indicates that the beginning and middle of the week are associated with increased fraudulent activity, potentially due to factors such as reduced oversight or increased transaction volume during these periods. Understanding these regional and temporal trends provides valuable insights for implementing targeted fraud prevention measures and allocating resources effectively.

Figure 17: Fraudulent TRX by Region

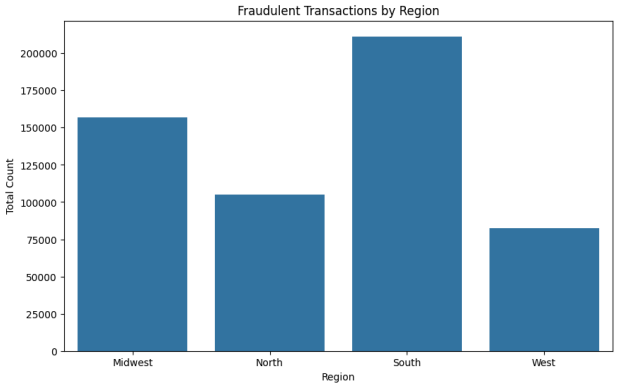
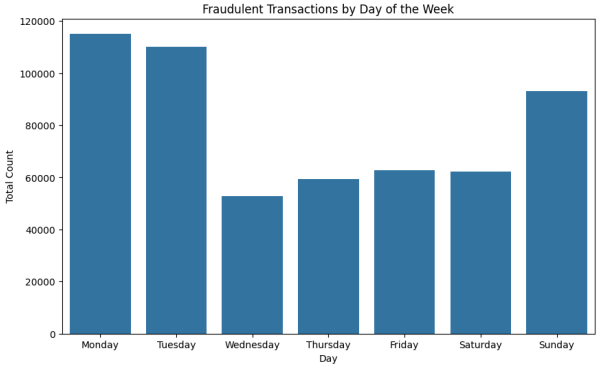


Figure 18: Fraud TRX by Weekday



Results

In this chapter, the results of the different models will be illustrated. As explained earlier, the models are evaluated based on the recall, precision, and F1 score through 2 different approaches which are 80%-20% train-test split, and 5-fold Cross-validation. The performance of the different models trained on different data imbalance handled datasets are shown in table 3, table 4, table 5, table 6, table 7, and table 8.

Table 3: Train-Test Split Recall

	Logistic Regression	Random Forest	XGBoost	LightGBM
RUS	0.75524	0.96270	**0.98135**	**0.97669**
Tomkelinks	0.0	0.96270	**0.98135**	**0.97669**
ROS	0.74825	0.69697	0.92308	0.95804
Smote	0.67599	0.70862	0.86247	0.87646
SmoteTomek	0.67599	0.70629	0.86247	0.87646
Smoteen	0.67599	0.71562	0.86713	0.88345

Table 4: Train-Test Split Precision

	Logistic Regression	Random Forest	XGBoost	LightGBM
RUS	0.02428	0.10677	0.10549	0.11269
Tomkelinks	0.0	0.11241	0.10549	0.11269
ROS	0.02553	**0.96764**	0.62857	0.31042
Smote	0.31419	0.91843	0.75820	0.54101
SmoteTomek	0.31419	**0.93519**	0.75820	0.54101
Smoteen	0.31419	0.86813	0.69533	0.49413

Table 5: Train-Test Split F1-Score

	Logistic Regression	Random Forest	XGBoost	LightGBM

RUS	0.04705	0.19223	0.19050	0.20207
Tomkelinks	0.0	0.20132	0.1905	0.20207
ROS	0.04937	**0.81030**	0.74788	0.46891
Smote	0.42899	**0.81102**	0.80698	0.66904
SmoteTomek	0.42899	0.80478	0.80698	0.66904
Smoteen	0.42899	0.78517	0.77178	0.63378

Table 6: Cross Validation Recall

	Logistic Regression	Random Forest	XGBoost	LightGBM
RUS	0.77056	0.94599	0.96328	0.96548
Tomkelinks	0.0	0.94183	0.96328	0.96548
ROS	0.76859	**1.0**	**1.0**	**1.0**
Smote	0.96938	0.99964	0.99972	0.99927
SmoteTomek	0.96938	0.99962	0.99972	0.99927
Smoteen	0.96938	0.99963	**0.99974**	0.99931

Table 7: Cross Validation Precision

	Logistic Regression	Random Forest	XGBoost	LightGBM
RUS	0.86197	0.96492	0.96541	0.96945
Tomkelinks	0.0	0.96384	0.96541	0.96945
ROS	0.87461	**0.99988**	0.99778	0.99189
Smote	0.99434	0.99974	0.99898	0.99713
SmoteTomek	0.99434	0.99972	0.99898	0.99713
Smoteen	0.99434	**0.99986**	0.9992	0.9977

Table 8: Cross Validation F1-Score

	Logistic Regression	Random Forest	XGBoost	LightGBM
RUS	0.81341	0.96492	0.96431	0.96745
Tomkelinks	0.0	0.96384	0.96431	0.96745
ROS	0.81818	**0.99988**	0.99889	0.99593
Smote	0.9817	0.99974	0.99935	0.9982
SmoteTomek	0.9817	0.99972	0.99935	0.9982
Smoteen	0.9817	**0.99986**	0.99947	0.99851

As evident by the tables above, logistic regression consistently demonstrates inferior performance relative to Random Forest, XGBoost, and LightGBM across all datasets, in both the train-test split and cross-validation analyses. This supports why the model was included in the study primarily: to serve as a baseline for comparative purposes. In the analysis of the train-test split results, it was observed that the highest recall scores were achieved by XGBoost with RUS and with TomekLinks, both equal to 0.98135. The second best-performing model judging by recall was LightGBM with RUS and with Tomeklink scoring 0.97669. In terms of precision, Random Forest combined with ROS exhibited the best performance with a score of 0.96764, followed closely by Random Forest with SmoteTomek which yielded a precision of 0.93519. Regarding the F1 score, Random Forest with SMOTE outperformed all other models with a result of 0.81102, and Random Forest using ROS ranked second with a F1 score of 0.81030. The cross-validation results revealed that Random Forest with ROS, XGBoost with ROS, and LightGBM with ROS all attained a perfect recall, achieving the highest possible score of 1.0, followed closely by XGBoost with Smoteen, scoring 0.99974. In precision, Random Forest with ROS was the leading model, followed by Random Forest with SMOTEEN with precisions of 0.99988 and 0.99986 respectively. For the F1 score, Random Forest with ROS again secured the highest ranking with a score of 0.99988, with Random Forest using SMOTEEN as the second best, achieving an F1 score of 0.99986.

In summary, logistic regression remains the least effective model across all evaluated metrics and data imbalance handling techniques, both in cross-validation and train-test split evaluation. Random Forest and XGBoost consistently exhibit superior performance across

various metrics, particularly when employing sampling techniques such as ROS and TomekLinks to address data imbalance. These findings suggest that these models, in conjunction with appropriate data imbalance handling methods, are more suitable for the datasets utilised in this study. The criteria for selecting the recommended best-performing model are discussed in the following chapter.

Discussion

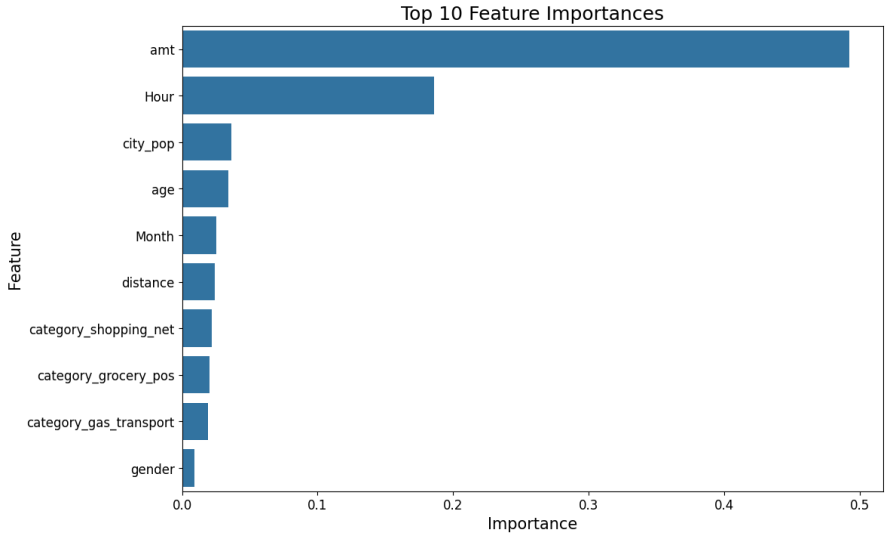
After evaluating the modelling results for all models using train/test split and cross-validation, the model was selected based primarily on the cross-validation results, as they help mitigate overfitting. With this decision, the recall metric was prioritised when selecting the best model, with precision considered a secondary priority. This approach was taken because the primary goal of this paper is to construct a model that correctly classifies fraudulent transactions. Misclassifying a fraudulent transaction as legitimate would be highly problematic, whereas identifying a legitimate transaction as fraudulent poses no significant threat, as the transaction would merely be flagged for further review and can be quickly resolved without any damaging consequences. Therefore, recall is the prioritised metric in this context. Since recall and precision are not given equal importance, the F1-Score is not beneficial for these purposes, as it equally weights both metrics by computing the harmonic mean. Consequently, the F1-Score was not considered when selecting the best model as well as the results of the train-test split. According to the results, the Random Oversampling (ROS) imbalance technique generally yielded high-performance metrics across all models, specifically with the Random Forest model. Therefore, the Random Forest model, integrated with the Random Oversampling imbalance technique, was selected as the best model because it achieved the highest recall and precision of 1.0 and 0.99988, respectively.

From the modelling results, logistic regression's weak performance can be explained by the fact that it is a single classifier in comparison to the other 3 models used which are ensemble classifiers and therefore have better performance. This model does not adequately flag fraudulent transactions due to the extreme data imbalance, specifically in the case of tomeklinks. In order to combat the weak performance of logistic regression, other levels of data imbalance handling can be used, particularly the algorithm-level approach where the decision threshold can be adjusted to better identify fraudulent transactions. It is also worth noting that although class imbalance is still persistent in tomeklinks training dataset, models trained on this dataset do not necessarily

poorly perform. In fact, models trained on this imbalanced dataset achieve equal performance to other models trained on balanced datasets. This indicates that enhancing class separability is indeed effective in handling class imbalance.

After selecting the Random Forest model integrated with the Random Oversampling technique, it became imperative to inverse-transform the data and interpret the model to understand how it identifies fraudulent transactions and the decision process involved in their identification. De-standardization of the data was necessary to bring the features back to their original scale, interpreting the model's results more intuitive and meaningful. First and foremost, measuring the feature importance of each attribute in the dataset was crucial to discerning which features the model heavily relies on for identifying fraudulent activities. According to the feature importance plot, the transaction amount emerged as the most critical feature, as it often exhibits unusual patterns in fraudulent transactions, such as unusually large or small amounts. Similarly, the hour of the transaction can be indicative of fraudulent behavior, as certain times of the day may be more susceptible to fraud. The cardholder's city population and age can provide insights into the demographics of fraudulent transactions, with densely populated areas potentially experiencing higher rates of fraud. The month of the transaction may also influence fraud patterns, with seasonal variations affecting transaction behavior. Additionally, the distance between the merchant and the cardholder can signal potential fraud, especially if transactions occur at unusually distant locations. Finally, the type of transaction, such as online shopping or gas purchases, can serve as important indicators of fraudulent activity, as certain transaction categories may be more prone to fraud than others. By understanding the significance of these features, the credit card fraud detection model can effectively prioritize and analyze transactions to enhance fraud detection accuracy and mitigate financial losses. Moreover, these findings directly align with the insights gained from the exploratory data analysis phase, confirming the relevance of the EDA insights in guiding the decision process of identifying fraudulent transactions.

Figure 19 : Model Feature Importance



As a tree-based ensemble machine learning method, Random Forest constructs multiple decision trees, each with a root node serving as the starting point for decision-making within the tree. In simple terms, a root node represents the feature and corresponding threshold used to initiate the splitting process when determining predictions. Extracting the number of trees in the model and identifying the most frequently used feature as the root node provides valuable insights into the decision-making process of the Random Forest model. Upon analysis, it has been determined that the model comprises a total of 100 trees, with the feature indicating whether or not the transaction was completed in online shopping being the most common root node across 17 of these trees. This choice of root node holds significant implications for identifying fraudulent transactions since online shopping transactions often demonstrate unique characteristics that may serve as red flags for fraudulent behaviour. Given that online platforms require the insertion of credit card details, they become prime targets for fraudulent activities such as theft and unauthorised transactions. The prominence of this feature as the root node underscores its importance in the model, supported by its high ranking in the feature importance plot. By prioritising this feature as the starting point for decision-making, the model can effectively discern and analyse patterns indicative of fraudulent activities during online transactions. Consequently, leveraging this feature enables the model to enhance its accuracy in detecting fraudulent behaviour, thereby bolstering fraud prevention measures and safeguarding against financial losses.

Understanding the maximum and minimum tree depths in the context of credit card fraud detection is crucial for assessing the decision-making complexity of the model. Tree depth refers to the length of the longest path from the root node to a leaf node within a decision tree. In simpler terms, it represents the number of decision rules involved in classifying transactions as legitimate or fraudulent. In our analysis, the minimum tree depth was 33, indicating relatively shallow decision trees with fewer decision rules. Conversely, the maximum tree depth was 37, suggesting deeper decision trees with more intricate decision rules. These depths signify the range of complexities inherent in the model's decision-making process. Shallow trees with fewer rules may capture broad patterns in transaction data, while deeper trees with more rules can discern finer nuances and anomalies. This diversity in tree depths enables the model to effectively identify fraudulent transactions across various levels of complexity, ultimately enhancing fraud detection accuracy and mitigating financial losses.

To visualise the decision-making process of the random forest model and gain insights into how it identifies fraudulent transactions, the tree with the minimum depth was selected for visualisation. However, with a depth of 33, the tree remained highly complex and difficult to interpret. Hence, only the first 3 levels of the tree were plotted after de-standardizing the features to extract meaningful insights and information. In the visualisation, each node represents a decision point where the model evaluates a specific feature and its threshold value to determine the next step in the classification process.

According to the tree plot, the root node is whether or not the transaction was completed for online shopping purposes which indicates that this is the feature on which the model initializes evaluation. If the transaction wasn’t completed during online shopping, then the transaction is assessed against a certain threshold value regarding the transaction amount which is 2.48 after de-standardization. If the transaction was completed for online shopping purposes, the transaction is evaluated as to whether or not the amount is less than 39.099. If it's less than this amount, the transaction is automatically classified as a legitimate transaction, hence labeled 0. This is just a glimpse of the decision trees and how the decision process is being held.

Figure 20 : Model Decision Tree Path

Conclusion

In conclusion, the primary aim of this study is to compare different machine learning algorithms against various data-level imbalance handling techniques to enhance credit card fraud detection systems. The findings from extensive analysis underscore the critical importance of addressing data imbalance in fraud detection datasets, where fraudulent transactions are significantly outnumbered by legitimate ones. Selecting the best performing model was made on 5-fold cross-validation evaluation and prioritising the recall metric over precision while maintaining a high figure for the latter metric. Accordingly, results demonstrate that Random Forest with Random Oversampling (ROS) emerged as the most effective model for detecting fraudulent transactions since it has a superior ability to identify fraudulent transactions while maintaining high precision. This is particularly significant as the primary goal of fraud detection is to ensure that no fraudulent transactions go undetected, even at the cost of increasing false positives. Furthermore, the study highlights the inadequacies of logistic regression compared to the other ensemble methods, and they also indicate that the undersampling technique tomklinks is indeed inefficient in handling imbalance despite maintaining the class imbalance in the data.

To enhance the robustness and efficiency of credit card fraud detection systems, future research and practical implementations should consider several recommendations. One important recommendation is the implementation of logistic regression models in both Python and R. Utilising R for statistical techniques can offer diverse methodologies and insights, leveraging R's

extensive statistical libraries. By comparing the performance of logistic regression models developed in both Python and R, a comprehensive evaluation of model effectiveness can be achieved. R provides unique statistical tools that complement Python's capabilities, such as Generalised Linear Models (GLM), Ridge Regression, and Lasso Regression. Applying these techniques to credit card fraud detection can highlight the strengths and limitations of different implementations, aiding in the selection of the most effective approach for handling fraud detection. Additionally, incorporating execution time as an evaluation metric is crucial for a more efficient analysis. The performance of a fraud detection system is not solely dependent on accuracy metrics like precision, recall, and F1 score, but also on the speed of execution, particularly in real-time transaction processing environments. Measuring the execution time for training and testing models is essential to understanding their scalability and efficiency. Fast execution times are vital for large-scale deployment in financial institutions where rapid decision-making is necessary. By comparing execution times across different models and data handling techniques, the most time-efficient and scalable solutions can be identified.

It is also suggested to develop a new metric that assigns varying importance to recall over precision, reflecting our study's emphasis on recall. This new metric would provide a more tailored evaluation of model performance, emphasizing the detection of fraudulent transactions. Given the critical nature of fraud detection, it is essential to prioritize recall, as explained before, to ensure that no fraudulent transactions are missed, even if it means accepting a higher false positive rate. This approach underscores the importance of capturing every potential fraudulent transaction to mitigate financial losses and enhance security. Furthermore, creating an application for banks that runs these advanced fraud detection models in real-time could significantly enhance their ability to identify fraudulent transactions almost instantaneously. Such an application would integrate seamlessly with existing banking systems, providing real-time alerts and decision-making support. This would not only improve the security and trustworthiness of financial transactions but also increase operational efficiency by reducing the time and resources spent on manual fraud detection processes.

In summary, these recommendations aim to improve the effectiveness, efficiency, and practical applicability of credit card fraud detection systems, ensuring that they are better equipped to handle the complexities of real-world financial environments while prioritizing the detection of fraudulent activities.

References

Bakhtiari, S., Nasiri, Z., & Vahidi, J. (2023). *Multimedia Tools and Applications*, *82*(19), 29057–29075. https://doi.org/10.1007/s11042-023-14698-2

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic Minority Over-sampling technique. *Journal of Artificial Intelligence Research*, *16*, 321–357. https://doi.org/10.1613/jair.953

Chung, J., & Lee, K. (2023). Credit Card Fraud Detection: An Improved Strategy for high Recall using KNN, LDA, and linear regression. *Sensors*, *23*(18), 7788. https://doi.org/10.3390/s23187788

Credit card fraud prediction. (2024, March 11). Kaggle. https://www.kaggle.com/datasets/kelvinkelue/credit-card-fraud-prediction/data

Pozzolo, A. D., & Bontempi, G. (2015). Adaptive machine learning for credit card fraud detection. *Université Libre De Bruxelles*. https://difusion.ulb.ac.be/vufind/Record/ULB-DIPOT:oai:dipot.ulb.ac.be:2013/221654/Details

Salekshahrezaee, Z., Leevy, J. L., & Khoshgoftaar, T. M. (2023). The effect of feature extraction and data sampling on credit card fraud detection. *Journal of Big Data*, *10*(1). https://doi.org/10.1186/s40537-023-00684-w

Singh, A., & Jain, A. (2022). An efficient credit card fraud detection approach using cost‐sensitive weak learner with imbalanced dataset. *Computational Intelligence*, *38*(6), 2035–2055. https://doi.org/10.1111/coin.12555

Singh, A., Ranjan, R. K., & Tiwari, A. (2021). Credit Card Fraud Detection under Extreme Imbalanced Data: A Comparative Study of Data-level Algorithms. *Journal of*

Experimental and Theoretical Artificial Intelligence, *34*(4), 571–598.

https://doi.org/10.1080/0952813x.2021.1907795