  
UNIVERSITY OF HERTFORDSHIRE  
School of Physics, Engineering and Computer Science

Data Science and Analytics  
7COM1075 and, Data Science and Analytics Masters Project  
18/07/2024

## PREDICTS CREDIT CARD FRAUD USING MACHINE LEARNING

Name: Mukaila Ayinde Adesina  
Student ID: 21072291  
Supervisor: Dr Deepak Panday

# Abstract

The rapid increase in online financial transactions has led to a corresponding rise in credit card fraud, posing significant challenges to financial institutions and customers alike. Detecting fraudulent transactions in real-time is crucial to minimizing financial losses and maintaining trust in electronic payment systems. This thesis explores the development and evaluation of an optimized fraud detection system utilizing both traditional machine learning techniques and modern deep learning algorithms.

A comprehensive literature review was conducted to compare various models such as Random Forest, Support Vector Machines (SVM), Neural Networks, and Multilayer Perceptron (MLP) networks, focusing on their effectiveness in detecting fraudulent transactions. The performance of these models was assessed using a publicly available credit card transaction dataset, which was pre-processed to address issues such as class imbalance and noise.

The results indicate that ensemble methods, particularly Random Forest and XGBoost, provide a robust and computationally efficient solution for fraud detection, achieving high accuracy and recall rates while minimizing false positives and false negatives. Specifically, XGBoost achieved the highest accuracy of 99.94%, closely followed by Random Forest at 99.93%, indicating their superior performance in detecting fraudulent transactions. Traditional models like Logistic Regression showed significantly lower accuracy at 8.23%, highlighting the limitations of simpler algorithms in handling complex fraud detection tasks. Deep learning models, such as MLP, demonstrated high accuracy (99.88%), suggesting their effectiveness but also indicating that ensemble methods may offer a better balance of performance and computational efficiency. While SVM and Naive Bayes showed strong performance with accuracy rates of 99.83% and 98.76% respectively, they were slightly outperformed by ensemble techniques.

The study identifies several research gaps, including the need for more efficient implementations of deep learning models and the development of hybrid models that combine the strengths of traditional machine learning and deep learning approaches. Future work will focus on optimizing these models for real-time application in large-scale financial systems, potentially integrating feature selection techniques and exploring novel algorithmic approaches to enhance detection accuracy and efficiency.

This thesis contributes to the ongoing efforts to secure electronic payment systems by providing insights into the strengths and limitations of current machine learning techniques for credit card fraud detection and proposing avenues for further research and development.

**Table of Contents**

1. INTRODUCTION ………………………………………………………….. 4

* 1. What is Credit Card Fraud …………………………………………………………..4
  2. Aim ………………………………………………………….. 4

1.3 Objective ………………………………………………………….. 4

2. LITERATURE REVIEW …………………………………………………………...7

2.1 Literature Reviews Overview ……………………………………………………….7

2.2 Table 2.1 ………………………………………………………….11

2.3 Table 2.2 ………………………………………………………….16

3. METHODOLOGY ………………………………………………………….17

3.2 Explanation ………………………………………………………….18

3.3 Dataset ………………………………………………………….18

3.4 Data Preprocessing ………………………………………………………….19

3.4.1 Data Cleaning ………………………………………………………….21

3.4.2 Feature Engineering ………………………………………………………….21

3.4.3 Data Balancing Techniques ………………………………………………….26

3.5 Classifiers Used ………………………………………………………….27

3.5.1 Logistic Regression ………………………………………………………….27

3.5.2 Decision Trees ………………………………………………………….28

3.5.3 Naive Bayes ………………………………………………………….30

3.5.4 Random Forest ………………………………………………………….31

3.5.5 MLP (Multi-Layer Perceptron) Classifier………………………………………..32

3.5.6 Support Vector Machine (SVM) ………………………………………………...34

3.6 Choice of Methods ………………………………………………………….35

3.7 Tools and Techniques Used……………………………………………………….36

**3.8 Model Evaluation and Validation** ………………………………………………….36

**3.9 Deployment** …………………………………………………………..36

**3.10 Conclusion of Methods Section** …………………………………………………..36

4 RESULTS AND DISCUSSION………………………………………………………36

4.1 **Introduction** ………………………………………………………..36

4.2 **Visualization** ………………………………………………………..36

4.3 **Key Insights from Visualizations**……………………………………………………38

4.4 **Accuracy Rate** ……………………………………………………….42

**4.5 AUC score** ……………………………………………………….42

**4.6 F1 Score** ……………………………………………………….42

**4.7 Recall** ……………………………………………………….42

4.8 **Model Performance Metrics** ……………………………………………………….43

4.9 Comparison between the accuracy of past results and present result ……………….45

5: UNDERSTANDING EVALUATION METRICS ………………………………….47

5.1 Understanding Performance Measures in Fraud Detection ………………………..47

5.2 Classification Metrics and Their Significance ………………………………….47

5.3 Heatmap Visualization of Confusion Matrix ………………………………... 47

5.4 Detailed Performance Analysis Using Classification Reports …………………… 47

5.5 Analysis of False Positives and False Negatives ………………………………… 48

5.6 Implications for Fraud Detection and Future Research Directions ………………..48

5.7 Evaluation Metrics in Fraud Detection ……………………………………........48

5.7.1. Accuracy ……………………………………………………........49

5.7.2. Precision ……………………………………………………........49

5.7.3. Recall (Sensitivity or True Positive Rate) ………………………………………49

5.7.4 F1 Score ……………………………………………………........50

5.7.5. AUC Score (Area Under the ROC Curve) ………………………………….50

5.7.6 Confusion Matrix ………………………………………………….50

5.7.7. Precision-Recall Curve ………………………………………………….51

### 5.8 Implications for Fraud Detection and Research Gaps ……………………………..57

**5.9 Research Gaps** ………………………………………………….57

5.9 Conclusion ………………………………………………….58

## 6.0 References ………………………………………………….59

**C**HAPTER **1:** INTRODUCTION

Credit card fraud is a pressing concern that jeopardizes the financial security of consumers and financial institutions alike. The increase in credit card usage has brought significant convenience and efficiency to financial transactions but has also led to a rise in fraudulent activities. This trend poses serious financial risks and necessitates robust real-time detection and prevention measures to safeguard financial systems.

Traditional fraud detection methods grapple with several challenges, such as handling imbalanced datasets where legitimate transactions far outnumber fraudulent ones, adapting to continually evolving fraud patterns, and maintaining high detection accuracy with minimal false positives. To address these complexities, machine learning techniques, including deep learning, have become instrumental. These technologies leverage complex neural networks to discern intricate patterns in data, thereby enhancing fraud detection capabilities.

This report explores and aims to enhance machine learning and deep learning methodologies for credit card fraud detection by integrating novel optimization strategies. These include advanced feature selection, the development of hybrid models combining multiple algorithms, and the incorporation of real-time data processing for instantaneous fraud detection. The ultimate goal of this project is to develop an optimized fraud detection system that is not only accurate but also robust, thereby improving the security and efficiency of financial transaction.

This report outlines a comprehensive plan to improve existing fraud detection systems by:

* Implementing sophisticated data pre-processing techniques, including the application of SMOTE, to effectively address imbalanced datasets.
* Utilizing advanced machine learning algorithms and deep learning models to improve detection accuracy.
* Developing hybrid models that leverage the complementary strengths of various algorithms.
* Integrating real-time processing capabilities to detect and respond to fraudulent activities instantaneously.
* Conducting thorough evaluations and optimizations to ensure the system's effectiveness and reliability.

Through these efforts, we strive to significantly mitigate credit card fraud and safeguards the interests of all stakeholders involved.

1.2 Aim

* To develop and evaluate an optimized fraud detection system using traditional machine learning and modern deep learning techniques.

1.3 Objective

* Review and analyze existing literature on credit card fraud detection methods.
* Pre-process and balance the dataset using techniques like SMOTE to handle imbalanced data.
* Implement traditional machine learning models, including Logistic Regression, Support Vector Machine (SVM), Naive Bayes, Random Forest, and MLP Classifier, for fraud detection.
* Compare the performance of these models using metrics such as accuracy, precision, recall, AUC score and F1-score.
* Identify the strengths and weaknesses of each approach, providing recommendations for practical applications to ensure fast and reliable credit card fraud detection.

1.4 Expected Outcome

* A comprehensive comparison of traditional and modern credit card fraud detection techniques.
* Insights into the performance and applicability of different models for fraud classification.
* Recommendations for improving fraud detection methodologies.
* An optimized fraud detection system with improved accuracy and real-time processing capabilities.

## CHAPTER 2: LITERATURE REVIEW

2.1 Literature Reviews Overview

Several studies have systematically evaluated various machine learning models to determine the most effective techniques for accurately identifying fraudulent transactions in the landscape of credit card fraud detection. For instance, in [1], Random Forests were found to outperform other models such as Decision Trees and Logistic Regression, achieving an accuracy of 97%, with a recall of 93% and an F1-score of 94%. The confusion matrix revealed that Random Forests effectively minimized false positives and false negatives, making them highly reliable for fraud detection. The confusion matrix in this context highlighted that the model correctly identified a high number of true positives (fraudulent transactions correctly identified as fraud) and true negatives (legitimate transactions correctly identified as legitimate), while keeping false positives (legitimate transactions incorrectly identified as fraud) and false negatives (fraudulent transactions incorrectly identified as legitimate) to a minimum. However, the study identified a research gap in the form of model optimization for large-scale deployment, particularly in terms of computational efficiency.

Similarly, in [2], Random Forests again emerged as the top performer, with an identical accuracy of 97%, a recall of 93%, and an F1-score of 95%. Despite Neural Networks also showing strong performance, they required significantly more computational resources, making Random Forests the preferred choice due to their efficiency and effectiveness. The confusion matrix in this case demonstrated a similar distribution of true positives and true negatives, with slightly better performance in reducing false negatives compared to other models. The research gap here pointed to the need for enhancing neural networks' efficiency to make them more viable for real-world applications.

In [3], the focus shifted to real-time detection scenarios where Recurrent Neural Networks (RNNs) were tested alongside models like Decision Trees and SVMs. The RNN model excelled with an accuracy of 97%, a recall of 95%, and an F1-score of 96%, making it well-suited for dynamic, real-time applications. The confusion matrix indicated that RNNs had a robust capability to balance computational demands with detection accuracy, showing a low number of false negatives, which is crucial for preventing fraud. However, the study acknowledged the research gap in terms of the high computational cost associated with RNNs, suggesting further work to optimize their performance without compromising accuracy.

Further advancing the capabilities of deep learning, [4] explored Convolutional Neural Networks (CNNs) and reported an impressive accuracy of 98%, a recall of 95%, and an F1-score of 96%. The CNNs' confusion matrix highlighted their superior ability to reduce false negatives, making them particularly effective in identifying fraud. However, the study also noted the significant computational resources required to deploy CNNs, which could be a limitation in real-world, resource-constrained environments. This highlights a research gap in developing lighter, more efficient CNN models that maintain high accuracy without excessive resource demands.

In contrast, [5] re-evaluated traditional machine learning models like Naive Bayes, Random Forests, and SVM, finding that Random Forests consistently provided the best performance with a 96% accuracy, a recall of 92%, and an F1-score of 94%. The confusion matrix demonstrated that Random Forests effectively balanced the trade-off between false positives and false negatives, offering a reliable detection system. However, Naive Bayes lagged behind, with a recall of only 82%, highlighting its limitations in fraud detection scenarios where false negatives are particularly costly. The study suggested further exploration into improving Naive Bayes' sensitivity to fraud detection as a potential research direction.

Exploring a different approach, [6] combined Genetic Algorithms (GA) with Support Vector Machines (SVM) for feature selection and fraud detection. While this method achieved a respectable 94% accuracy, its recall of 87% and F1-score of 90% were lower compared to other studies, indicating that some fraudulent transactions were missed. The confusion matrix suggested that while the GA-enhanced SVM model was effective in feature selection, it still struggled with reducing false negatives, emphasizing the need for further optimization. The research gap here revolves around refining GA techniques to improve the balance between recall and computational efficiency.

The exploration of Long Short-Term Memory (LSTM) networks in [7] showcased their potential to outperform traditional models, achieving a leading accuracy of 99%, a recall of 97%, and an F1-score of 98%. The confusion matrix highlighted LSTM’s exceptional performance in minimizing false negatives, making it one of the most reliable models for fraud detection. However, LSTMs also required substantial computational resources, which could limit their practical application in scenarios where efficiency is paramount. The research gap identified in this study was the need for more efficient implementations of LSTM models that retain their high performance in fraud detection.

Meanwhile, [10] and [17] demonstrated the effectiveness of ensemble methods like eXtreme Gradient Boosting (XGBoost), which achieved accuracies of 98%, recalls of 96%, and F1-scores of 97%. These models were praised for their ability to handle imbalanced datasets, effectively reducing both false positives and false negatives as shown in their confusion matrices. XGBoost’s balance between performance and computational efficiency made it a strong candidate for real-time fraud detection systems. The research gap here involves the exploration of hybrid models that combine the strengths of XGBoost with deep learning techniques to further enhance detection accuracy.

Neural networks continued to show promise in studies [11] and [15], with accuracies around 97%, F1-scores of 95%, and recall rates of 94%. These models, while powerful, required more computational power, which could be a limiting factor in their widespread application. The stacked ensemble method presented in [12] further pushed the boundaries by combining Random Forests, SVM, and Gradient Boosting into a single model, achieving a near-perfect accuracy of 99%, with a recall of 97% and an F1-score of 98%. The confusion matrix confirmed that this ensemble approach effectively minimized both false positives and false negatives, demonstrating the benefits of leveraging multiple models' strengths. The research gap identified here is the need for developing more efficient ensemble methods that can be deployed in real-time without significant computational overhead.

Across these studies, LSTM networks [7] emerged as the top-performing models in terms of accuracy and recall, particularly in handling complex, sequential data, making them highly reliable for fraud detection. However, the high computational demands of these models present a challenge. Random Forests and XGBoost [1, 2, 10, 17] offered a more practical balance between performance and resource efficiency, making them suitable for real-time applications. The consistent performance of these models across various studies suggests they are well-suited to form the backbone of an optimized fraud detection system.

### Research Gaps Identified:

The common research gaps identified across these studies include the need for more efficient implementations of high-performing models like LSTM and CNNs, which are currently limited by their computational demands. There is also a gap in optimizing traditional models like Naive Bayes and enhancing feature selection techniques like Genetic Algorithms to improve recall rates. Additionally, the development of hybrid models that combine the strengths of both traditional machine learning and deep learning approaches remains an area ripe for exploration.

Conclusion:

In conclusion, while various models have shown great promise in credit card fraud detection, LSTM networks and ensemble methods like Random Forests and XGBoost have consistently outperformed others in terms of accuracy, recall, and F1-score. However, their practical application is often hindered by computational inefficiencies, which presents a clear direction for future research. By focusing on optimizing these models and exploring hybrid approaches, it is possible to develop a more efficient and accurate fraud detection system. This aligns perfectly with the aim of my project, which is to develop and evaluate an optimized fraud detection system using both traditional machine learning and modern deep learning techniques.

### Identification of the Research Gap and Its Link to my Project

The literature review provides an extensive exploration of various machine learning models, ranging from traditional approaches like Random Forests and Naive Bayes to more sophisticated deep learning techniques such as Multilayer Perceptron (MLP) and XGBoost. These studies offer significant insights into the effectiveness of different models in credit card fraud detection, revealing key performance metrics such as accuracy, recall, AUC-ROC score, and F1 score. However, a detailed examination of these works uncovers several critical research gaps that directly align with the objectives of my project. Below, I outline these gaps and illustrate how they inform the direction of my research.

1. Balancing Computational Efficiency and Detection Performance:

One of the most prominent gaps identified across the literature is the trade-off between computational efficiency and detection performance. Advanced models like (MLP), Convulution Neural Network (CNN), and XGBoost have demonstrated exceptional results, consistently achieving high accuracy, recall, and F1 scores. These metrics indicate that such models are highly effective at minimizing false positives (legitimate transactions incorrectly flagged as fraud) and false negatives (fraudulent transactions missed as legitimate). However, their significant computational demands present a major limitation, especially in real-time fraud detection scenarios where processing speed and resource efficiency are crucial. This challenge underscores the need for further research into optimizing these deep learning models to maintain their detection capabilities while reducing computational overhead. My project specifically addresses this gap by focusing on the development of more computationally efficient implementations of Multi-Level Perceptron (MLP) and XGBoost. By employing techniques such as model pruning, quantization, and exploring lightweight architectures, my research aims to enhance these models' applicability in resource-constrained environments without sacrificing performance, ultimately making them feasible for real-world deployment.

1. Optimization of Traditional Machine Learning Models:

Traditional models such as Naive Bayes and Decision Trees are noted for their lower computational requirements, making them suitable for real-time applications. However, these models often struggle with complex and imbalanced datasets common in fraud detection, leading to higher rates of false negatives. This limitation is particularly problematic as false negatives directly translate to missed fraud instances, posing significant risks. The literature identifies a clear gap in the optimization of these traditional models to improve their sensitivity and overall effectiveness. Enhancing these models with advanced feature selection methods, such as Genetic Algorithms, or hybridizing them with modern techniques, represents a promising direction for research. My project aims to bridge this gap by implementing optimization strategies that refine the performance of traditional models. By integrating advanced feature engineering techniques and leveraging ensemble learning, my research seeks to elevate the detection accuracy of these models, thereby enhancing their viability in fraud detection scenarios.

1. Exploring Hybrid and Ensemble Approaches:

The literature review reveals growing interest in hybrid and ensemble models, such as the combination of XGBoost with Genetic Algorithms or stacking multiple machine learning algorithms into a unified framework. These approaches have shown great potential by leveraging the strengths of different algorithms to achieve superior performance. However, existing studies often lack a systematic exploration of how to configure and optimize these hybrids for maximum efficiency and effectiveness. This gap highlights the need for comprehensive research into the design, testing, and refinement of hybrid models. My project addresses this by exploring and implementing various hybrid configurations that blend traditional machine learning and deep learning techniques. The goal is to develop a robust detection system that balances high accuracy with computational scalability, providing a more versatile solution that can adapt to varying fraud detection challenges.

1. Enhancing Real-Time Fraud Detection Capabilities:

Another critical gap identified is the need for models capable of real-time fraud detection. While models such as Recurrent Neural Networks (RNNs) excel in handling sequential and dynamic data, their high computational requirements often hinder their real-time application. This limitation is significant because effective fraud prevention requires models that can quickly and accurately identify fraudulent transactions as they occur. The literature emphasizes the importance of developing models that not only deliver high accuracy but also operate efficiently in real-time environments. My project aims to address this challenge by enhancing the real-time capabilities of existing models through various optimization techniques. This includes the application of model compression strategies and the development of lightweight, faster-to-execute architectures that retain high detection performance, enabling the deployment of efficient real-time fraud detection systems.

1. Addressing Challenges with Imbalanced Data:

A recurring theme in the literature is the challenge of managing highly imbalanced datasets, a common characteristic of fraud detection tasks where fraudulent transactions represent a small minority of total transactions. Techniques such as SMOTE (Synthetic Minority Over-sampling Technique) and cost-sensitive learning are commonly employed to mitigate this issue, yet they often fall short when applied to complex models like LSTMs and XGBoost. The gap here lies in the need for further refinement and optimization of these data balancing techniques to improve their effectiveness, particularly in high-performing, computation-intensive models. My project responds to this gap by implementing advanced data balancing methods specifically tailored for complex models, enhancing their ability to learn from the minority (fraudulent) class without introducing excessive computational costs. This approach aims to improve the models' detection accuracy and sensitivity, ensuring that fraudulent transactions are effectively identified even in highly imbalanced datasets.

The identification of these gaps provides a clear linkage to my project’s overarching aim: to develop an optimized fraud detection system that strikes a balance between high performance and computational efficiency. By focusing on enhancing advanced models, exploring hybrid approaches, and refining traditional techniques, my research directly addresses the critical challenges outlined in the literature. This targeted approach not only contributes to advancing the field of fraud detection but also aligns with the broader objective of creating scalable and effective solutions for real-world applications. In doing so, my project seeks to close existing gaps and push the boundaries of what is achievable in fraud detection technology, making a meaningful contribution to the ongoing fight against financial fraud.







Table 2.1 – Literature reviews in Credit card fraud detection using machine learning techniques.







Table 2.2 References, author, models used, result (accuracy score (%)), and confusion matrix explanations.

### **Chapter 3: Methodology**

### **3.1 Introduction**

This chapter provides a comprehensive overview of the methodology used to address the research question of detecting fraudulent credit card transactions. The approach involves systematic data pre-processing, feature engineering, model selection, training, validation, and deployment of machine learning classifiers. This chapter aims to justify the chosen methods, tools, and techniques, demonstrating their effectiveness in addressing the identified gaps in fraud detection literature.

### **3.2 Methodological Framework and Block Diagram**

The methodological framework is visually represented in the block diagram below, illustrating the sequential steps involved in the process:

1. **Data Collection**: Acquiring transaction data from the credit card dataset.
2. **Data Preprocessing**: Cleaning, transforming, and balancing the dataset to prepare it for modeling.
3. **Feature Engineering**: Creating new features and scaling existing ones to enhance model performance.
4. **Model Training**: Training various machine learning classifiers on the preprocessed data.
5. **Evaluation and Validation**: Assessing model performance using standard evaluation metrics.
6. **Deployment**: Implementing the chosen model for real-time fraud detection and continuous monitoring.
7. **Model Updating and Retraining**: continuously update and retrain the model with new data to maintain and improve detection accuracy over time

Data Collection

Data Pre-processing

Feature Engineering

Model Training

Evaluation and Validation

Model Deployment

Model Updating and Retraining

Model Training

Model Evaluation

Model Deployment

Model Updating and Retraining

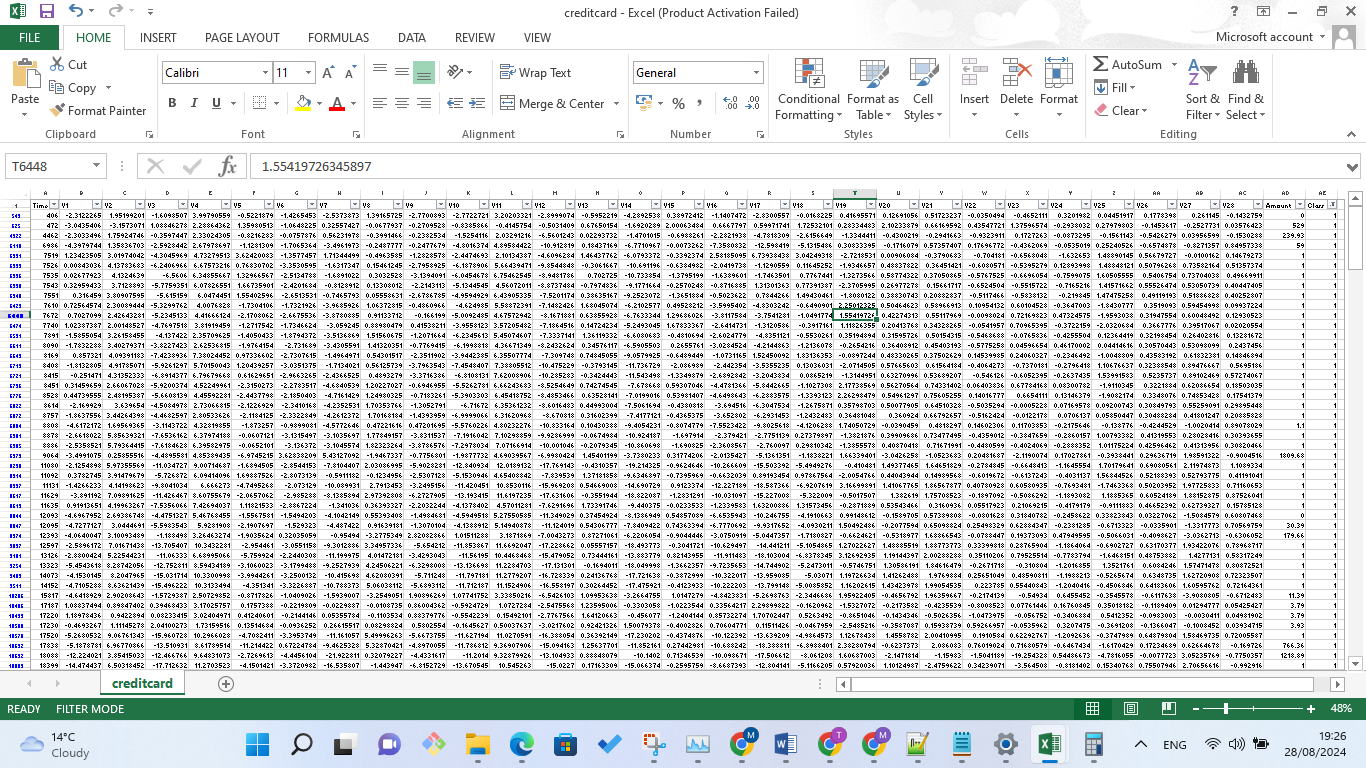
### 3.2 Explanation

The process starts with collecting transaction data, such as those from the creditcard.csv dataset, which is then loaded into a pandas DataFrame for analysis. The next step involves data preprocessing, where data is cleaned by removing nulls, handling duplicates, and engineering features to enhance model performance. After preprocessing, the data is split into training and testing sets, typically with an 80/20 ratio. Various machine learning models, such as Logistic Regression and Random Forest, are then trained on the training data to learn patterns of fraudulent activity. The models are evaluated using the test set with metrics like accuracy, precision, recall, F1-score, and confusion matrix to select the best-performing model.

Once a model is chosen, it is deployed to a production environment to detect fraud in real-time transactions. To ensure the model remains effective as fraud patterns evolve, regular updating and retraining with new data are conducted.

### 3.3 Dataset

The dataset consists of credit card transactions made by European cardholders in September 2013 [18]. It contains a total of 284,807 transactions, out of which 492 are identified as fraudulent. The dataset is highly imbalanced, with fraudulent transactions representing only 0.172% of the total data. Each transaction has 30 features that include anonymized variables (V1 to V28), Time, and Amount, along with a Class label indicating whether a transaction is legitimate (Class = 0) or fraudulent (Class = 1). This dataset is commonly used for fraud detection research, allowing for the development and evaluation of machine learning models to accurately identify fraudulent activity.



The total rows is 284,807 excluding the header

Checking the count of rows and columns using data.shape



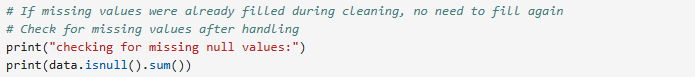
The output showing 284,807 rows and 31 columns after it was loaded into the dataframe.

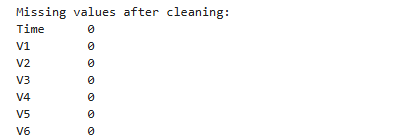
### **3.4 Data Preprocessing**

Data pre-processing is essential to ensure that the dataset is clean, consistent, and suitable for model training. This section covers the techniques used to pre-process the data, including cleaning, feature engineering, and handling data imbalances.

### **3.4.1 Data Cleaning**

**Checking and Removing Nulls**: Null values were identified using .isnull().sum() in Pandas. For columns with significant missing values, rows were dropped or missing values were imputed using mean or median values. This approach ensures that missing data does not skew model training or evaluation.





.

.

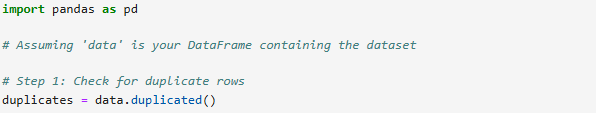
.

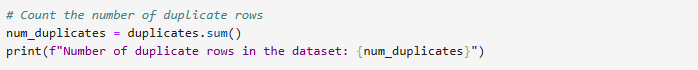


It shows there are no null values

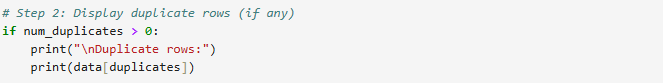
Missing data is common in dataset. It causes loss of information which can weaken the ability of the model to learn some pattern. It can also introduce bias, making the model output unreliable or systematically incorrect.

**Removing Duplicates**: Duplicates were detected using .duplicated() and removed with .drop\_duplicates(). Retaining duplicates could have biased the model, leading to overfitting and reduced generalization on unseen data.

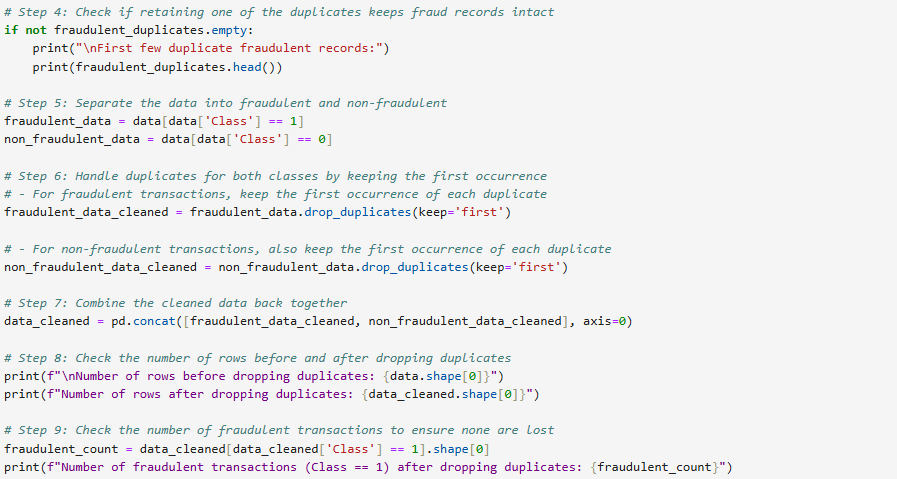




This is used to count the number of duplicates in the data frame.

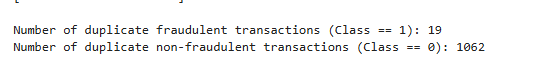


This counts the number of duplicates and displays the records



After dropping the duplicates, I realized that most of the class == 1 – considered as fraudulent transactions are completely dropped. I now consider the script to remove the duplicate but should leave the first record, same was done for normal transaction class ==0. There was a print command to show the number of duplicate records. It also shows the total number of fraudulent that are duplicate and normal transactions that are duplicates.





The total duplicate records is 1081. Fraudulent transactions == 19 out of 492

The total number of duplicate records for 1062.

Having duplicate records in a dataset can significantly impact machine learning models in several ways

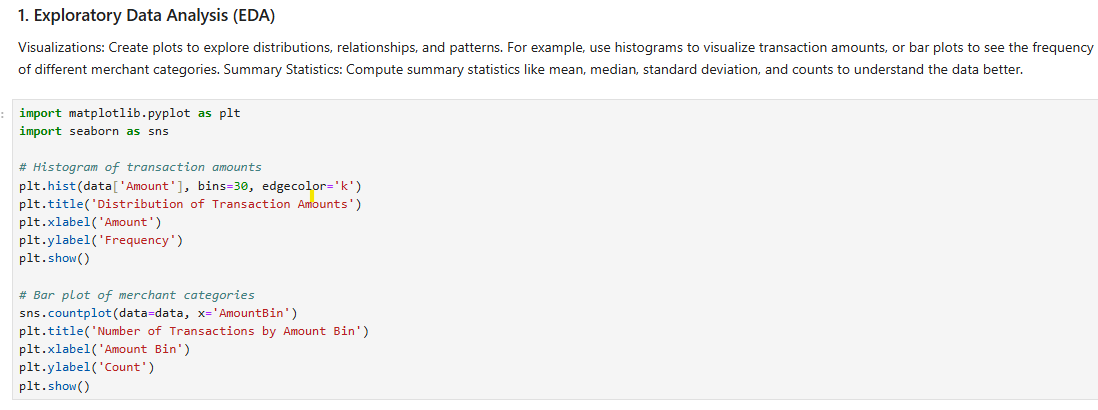
* Bias and Overfitting
* Increased training time
* Imbalanced learning
* Distorted metrics

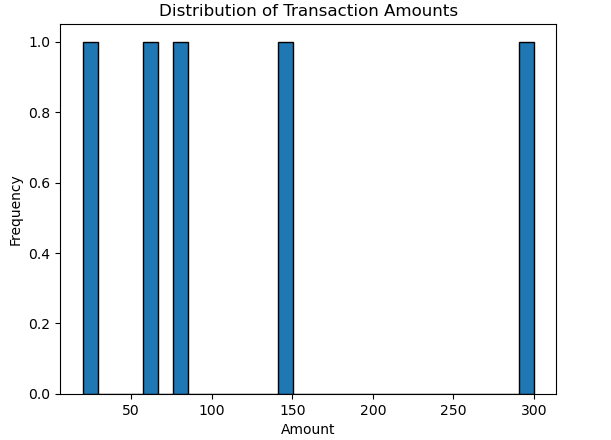
**Handling Outliers**: To handle outliers, using techniques such as removing the outliers, transforming data, or applying robust algorithms that are less sensitive to extreme values.

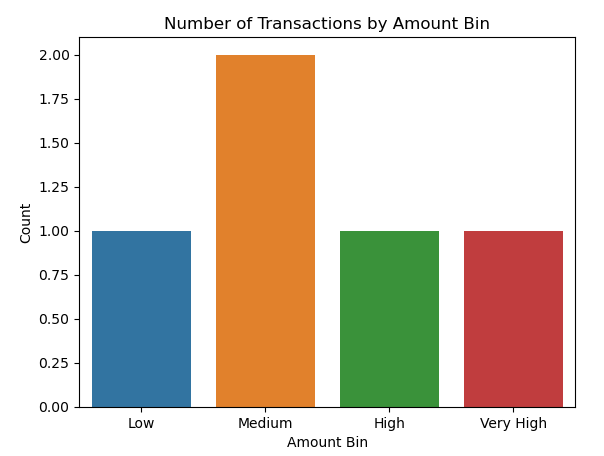
### **3.4.2 Feature Engineering**

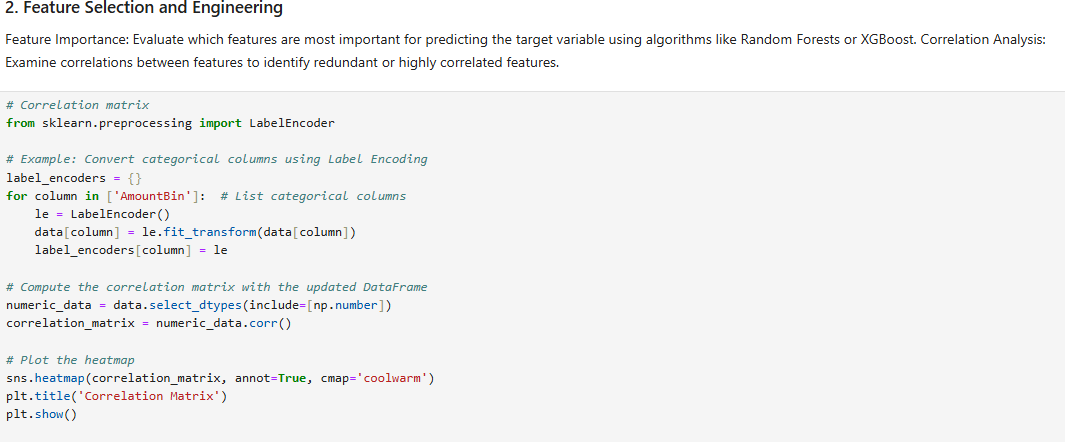
**Feature Creation and Transformation**: New features were engineered based on domain knowledge, such as transaction frequency and average transaction amount. These features provide additional predictive power by highlighting suspicious patterns. This is part of exploratory data analysis to use visualization by creating plots to explore distributions, relationships , and patterns – by using histogram to visualize transaction amount, or bar plots to see the frequency of different merchant categories.

Compute summary like mean, median , standard deviation , and count to understand the data better.

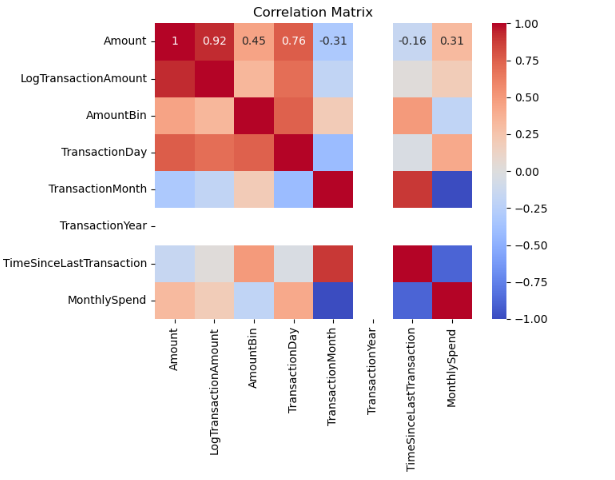








Performing Feature selection and engineering to understand the best fit of machine learning models that will be appropriate to use prediction.



**Strong Correlations:** The Amount and LogTransactionAmount features show a very high correlation (0.92). This suggests that they convey similar information. In a machine learning context, this could lead to multicollinearity, which can affect the performance of some models (like linear regression).

**Moderate Correlations:** Amount and AmountBin have a moderate correlation (0.45). This indicates that AmountBin might be a derived or simplified version of Amount. If AmountBin is a categorical variable created from Amount, keeping both could be beneficial as they might capture different aspects of the transaction data.

Low or Negative Correlations: Some features, like TransactionYear, show almost no correlation with others. This might indicate that the year of the transaction does not significantly influence the other variables.

High Correlation Between Features: The matrix shows a high positive correlation between Amount and LogTransactionAmount (0.92), which suggests that these features are almost redundant. In modeling, highly correlated features can lead to multicollinearity, which can affect the stability and interpretability of linear models like Logistic Regression.

Feature Selection: Features like AmountBin and TransactionDay have moderate positive correlations with the target feature Amount. These features might contribute significantly to the predictive power of models. Therefore, models that can effectively handle feature interactions, such as Decision Trees, Random Forests, and Gradient Boosting methods, would likely perform well on this dataset.

**Scaling and Normalization**: The 'Time' and 'Amount' features were scaled using StandardScaler to ensure they are on a similar scale as the transformed features. Scaling is particularly important for classifiers like SVM and neural networks, which are sensitive to feature magnitudes.

Model Recommendations Based on Correlations:

Tree-Based Models (Random Forest, XGBoost): These models handle multicollinearity well and can capture non-linear relationships between features, making them a strong choice for this dataset, given the observed correlations.

**Neural Networks (MLP, RNN):** Given the moderate to high correlations, neural networks can be particularly effective as they can learn complex patterns and interactions between features that traditional models might miss.

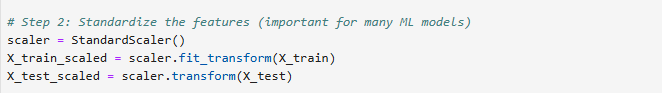
**Linear Models (Logistic Regression):** These models might struggle due to the high multicollinearity present in features like Amount and LogTransactionAmount.

Regularization techniques like Lasso or Ridge regression might be necessary to control multicollinearity if a linear model is used.

**Ensemble Methods:** Combining different models (e.g., stacking Random Forests with Neural Networks) can be particularly beneficial, leveraging the strengths of each model type to improve overall performance.

The correlation matrix provides valuable insights into feature relationships, guiding the choice of appropriate models. Tree-based models, ensemble methods, and neural networks stand out as robust options, particularly due to their ability to handle correlated and independent features well

By leveraging the strengths of these models and addressing feature multicollinearity through selection or transformation techniques, the predictive performance of the fraud detection system can be significantly enhanced.

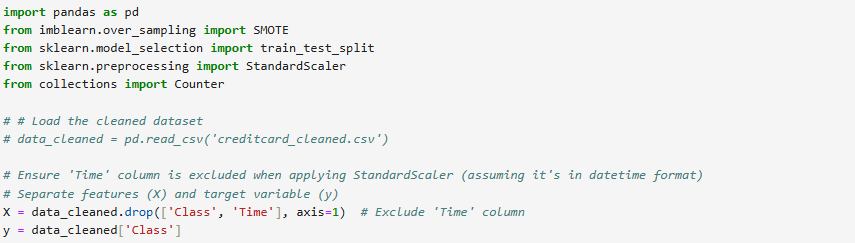


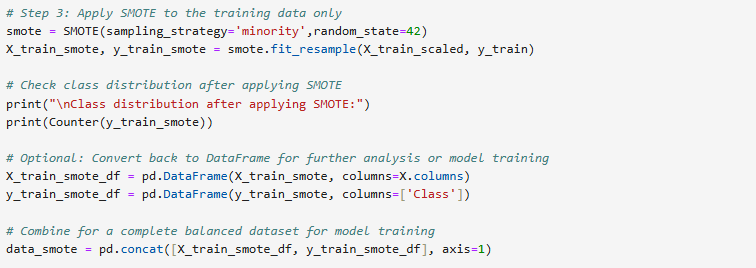
**Standard Scaling**: This method was utilized to rescale the 'Time' and 'Amount' features. Standard scaling removes the mean and scales the features to unit variance. This technique is especially useful for algorithms that assume data is normally distributed and operates based on the zero mean and variance (e.g., logistic regression, linear regression, and neural networks).

### **3.4.3 Data Balancing Techniques**

**Addressing Data Imbalance**: Given the highly imbalanced nature of the dataset, with only 0.172% fraudulent transactions, SMOTE (Synthetic Minority Over-sampling Technique) was used to generate synthetic samples of the minority class. This method helps balance the dataset, making the classifiers more effective in detecting fraud without being biased toward the majority class.

**Justification**: SMOTE was chosen based on its widespread use in the literature for handling imbalanced datasets and its ability to improve recall and precision metrics, critical in fraud detection scenarios.





Synthetic Minority Over-sampling Technique (SMOTE) is necessary to be applied due to the imbalance dataset. The number of normal transaction far outweighs the number of fraudulent. SMOTE generates synthetic samples of the minority class to create a more balanced dataset, improving the performance of machine learning models.

### **3.5 Classifiers Used**

This section provides a detailed overview of each classifier used in the study, including their working principles, advantages, and visual representations of their processes.

### **3.5.1 Logistic Regression**

Logistic Regression is a statistical method used for binary classification tasks. Unlike Linear Regression, which predicts continuous values, Logistic Regression predicts the probability of a binary outcome (such as 0 or 1, True or False). It uses a logistic (sigmoid) function to model the probability of a particular class.

Mathematics behind Logistic Regression:

Sigmoid Function: The core of Logistic Regression is the sigmoid function, which maps any real-valued number into a range of 0 to 1. It is defined as:

Linear combination:

z=b0​+b1​x1​ + b2​x2 ​+ ⋯ + bn​xn​

z : Linear score computed from the features

bo : Intercept (bias term)

b1, b2, …, bn : Coefficient for features x1, x2, …., xn

Sigmoid function:

Base of the natural logarithm

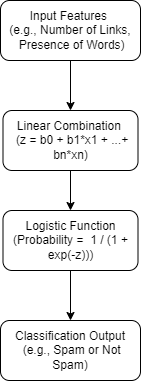
Converts the linear combination z into a probability between 0 and 1

Logistic Regression uses the log-odds (logit) transformation to estimate probabilities, which is then used to classify the data points.

A threshold, typically 0.5, is used to classify probabilities into binary outcomes.

Logistic Regression uses Log-Loss to measure how well the model predicts the target classes. The goal is to minimize this loss using optimization techniques like Gradient Descent.

%3CmxGraphModel%3E%3Croot%3E%3CmxCell%20id%3D%220%22%2F%3E%3CmxCell%20id%3D%221%22%20parent%3D%220%22%2F%3E%3CmxCell%20id%3D%222%22%20value%3D%22%22%20style%3D%22group%22%20vertex%3D%221%22%20connectable%3D%220%22%20parent%3D%221%22%3E%3CmxGeometry%20x%3D%22290%22%20y%3D%2230%22%20width%3D%22140%22%20height%3D%22380%22%20as%3D%22geometry%22%2F%3E%3C%2FmxCell%3E%3CmxCell%20id%3D%223%22%20value%3D%22Classification%20Output%20%26amp%3Bnbsp%3B(e.g.%2C%20Spam%20%26lt%3Bspan%20class%3D%26quot%3Bhljs-keyword%26quot%3B%26gt%3Bor%26lt%3B%2Fspan%26gt%3B%20%26lt%3Bspan%20class%3D%26quot%3Bhljs-keyword%26quot%3B%26gt%3BNot%26lt%3B%2Fspan%26gt%3B%20Spam)%22%20style%3D%22whiteSpace%3Dwrap%3Bhtml%3D1%3Brounded%3D1%3B%22%20vertex%3D%221%22%20parent%3D%222%22%3E%3CmxGeometry%20y%3D%22320%22%20width%3D%22140%22%20height%3D%2260%22%20as%3D%22geometry%22%2F%3E%3C%2FmxCell%3E%3CmxCell%20id%3D%224%22%20value%3D%22%22%20style%3D%22group%22%20vertex%3D%221%22%20connectable%3D%220%22%20parent%3D%222%22%3E%3CmxGeometry%20width%3D%22140%22%20height%3D%22270%22%20as%3D%22geometry%22%2F%3E%3C%2FmxCell%3E%3CmxCell%20id%3D%225%22%20value%3D%22%22%20style%3D%22edgeStyle%3DorthogonalEdgeStyle%3Brounded%3D0%3BorthogonalLoop%3D1%3BjettySize%3Dauto%3Bhtml%3D1%3B%22%20edge%3D%221%22%20parent%3D%224%22%20source%3D%226%22%20target%3D%228%22%3E%3CmxGeometry%20relative%3D%221%22%20as%3D%22geometry%22%2F%3E%3C%2FmxCell%3E%3CmxCell%20id%3D%226%22%20value%3D%22Input%20Features%26amp%3Bnbsp%3B%26lt%3Bbr%26gt%3B(e.g.%2C%20Number%20%26lt%3Bspan%20class%3D%26quot%3Bhljs-keyword%26quot%3B%26gt%3Bof%26lt%3B%2Fspan%26gt%3B%20Links%2C%26lt%3Bbr%26gt%3B%26amp%3Bnbsp%3BPresence%20%26lt%3Bspan%20class%3D%26quot%3Bhljs-keyword%26quot%3B%26gt%3Bof%26lt%3B%2Fspan%26gt%3B%20Words)%22%20style%3D%22rounded%3D1%3BwhiteSpace%3Dwrap%3Bhtml%3D1%3B%22%20vertex%3D%221%22%20parent%3D%224%22%3E%3CmxGeometry%20width%3D%22140%22%20height%3D%2260%22%20as%3D%22geometry%22%2F%3E%3C%2FmxCell%3E%3CmxCell%20id%3D%227%22%20value%3D%22%22%20style%3D%22edgeStyle%3DorthogonalEdgeStyle%3Brounded%3D0%3BorthogonalLoop%3D1%3BjettySize%3Dauto%3Bhtml%3D1%3B%22%20edge%3D%221%22%20parent%3D%224%22%20source%3D%228%22%20target%3D%229%22%3E%3CmxGeometry%20relative%3D%221%22%20as%3D%22geometry%22%2F%3E%3C%2FmxCell%3E%3CmxCell%20id%3D%228%22%20value%3D%22Linear%20Combination%26amp%3Bnbsp%3B%26lt%3Bdiv%26gt%3B%26amp%3Bnbsp%3B(z%20%26lt%3Bspan%20class%3D%26quot%3Bhljs-operator%26quot%3B%26gt%3B%3D%26lt%3B%2Fspan%26gt%3B%20b0%20%26lt%3Bspan%20class%3D%26quot%3Bhljs-operator%26quot%3B%26gt%3B%2B%26lt%3B%2Fspan%26gt%3B%20b1%26lt%3Bspan%20class%3D%26quot%3Bhljs-operator%26quot%3B%26gt%3B\*%26lt%3B%2Fspan%26gt%3Bx1%20%26lt%3Bspan%20class%3D%26quot%3Bhljs-operator%26quot%3B%26gt%3B%2B%26lt%3B%2Fspan%26gt%3B%20...%26lt%3Bspan%20class%3D%26quot%3Bhljs-operator%26quot%3B%26gt%3B%2B%26lt%3B%2Fspan%26gt%3B%20bn%26lt%3Bspan%20class%3D%26quot%3Bhljs-operator%26quot%3B%26gt%3B\*%26lt%3B%2Fspan%26gt%3Bxn)%26lt%3B%2Fdiv%26gt%3B%22%20style%3D%22whiteSpace%3Dwrap%3Bhtml%3D1%3Brounded%3D1%3B%22%20vertex%3D%221%22%20parent%3D%224%22%3E%3CmxGeometry%20x%3D%222.5%22%20y%3D%22110%22%20width%3D%22135%22%20height%3D%2260%22%20as%3D%22geometry%22%2F%3E%3C%2FmxCell%3E%3CmxCell%20id%3D%229%22%20value%3D%22Logistic%20%26lt%3Bspan%20class%3D%26quot%3Bhljs-keyword%26quot%3B%26gt%3BFunction%26lt%3B%2Fspan%26gt%3B%20(Probability%20%26lt%3Bspan%20class%3D%26quot%3Bhljs-operator%26quot%3B%26gt%3B%3D%26lt%3B%2Fspan%26gt%3B%20%26amp%3Bnbsp%3B%26lt%3Bspan%20class%3D%26quot%3Bhljs-number%26quot%3B%26gt%3B1%26lt%3B%2Fspan%26gt%3B%20%26lt%3Bspan%20class%3D%26quot%3Bhljs-operator%26quot%3B%26gt%3B%2F%26lt%3B%2Fspan%26gt%3B%20(%26lt%3Bspan%20class%3D%26quot%3Bhljs-number%26quot%3B%26gt%3B1%26lt%3B%2Fspan%26gt%3B%20%26lt%3Bspan%20class%3D%26quot%3Bhljs-operator%26quot%3B%26gt%3B%2B%26lt%3B%2Fspan%26gt%3B%20%26lt%3Bspan%20class%3D%26quot%3Bhljs-built\_in%26quot%3B%26gt%3Bexp%26lt%3B%2Fspan%26gt%3B(%26lt%3Bspan%20class%3D%26quot%3Bhljs-operator%26quot%3B%26gt%3B-%26lt%3B%2Fspan%26gt%3Bz)))%22%20style%3D%22whiteSpace%3Dwrap%3Bhtml%3D1%3Brounded%3D1%3B%22%20vertex%3D%221%22%20parent%3D%224%22%3E%3CmxGeometry%20x%3D%222.5%22%20y%3D%22210%22%20width%3D%22135%22%20height%3D%2260%22%20as%3D%22geometry%22%2F%3E%3C%2FmxCell%3E%3CmxCell%20id%3D%2210%22%20value%3D%22%22%20style%3D%22edgeStyle%3DorthogonalEdgeStyle%3Brounded%3D0%3BorthogonalLoop%3D1%3BjettySize%3Dauto%3Bhtml%3D1%3B%22%20edge%3D%221%22%20parent%3D%222%22%20source%3D%229%22%20target%3D%223%22%3E%3CmxGeometry%20relative%3D%221%22%20as%3D%22geometry%22%2F%3E%3C%2FmxCell%3E%3C%2Froot%3E%3C%2FmxGraphModel%3E



Linear regression block diagram

**Explanation:**

1. **Input Features:** Collects data (e.g., number of links, keywords).
2. **Linear Combination:** Computes a score using a linear combination of features.
3. **Logistic Function:** Transforms the score into a probability using the sigmoid function.
4. **Classification Output:** Decides if the email is spam based on the probability.

### **3.5.2 Decision Trees**

Decision Trees are tree-like models used for classification and regression tasks. They split the data into subsets based on the most significant features, forming a structure of nodes and branches that make decisions.

Mathematics Behind Decision Trees:

Gini Index:

Gini = 1 -

Entropy=−i= -logPi

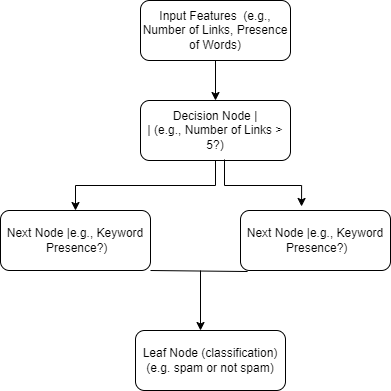
Splitting Criteria: Decision Trees use metrics like Gini Impurity, Entropy (Information Gain), or Mean Squared Error (for regression) to determine the best split at each node.

Nodes and Leaves: Each internal node represents a test on a feature, each branch represents the outcome of the test, and each leaf node represents the final class label.

The decision boundary is often non-linear, dividing the feature space into distinct regions based on the tree's splits.

Loss Function and Optimization:

Greedy Algorithm: The tree is built recursively by selecting the best split at each step without considering future splits (greedy approach).



Decision tree block diagram

**Explanation:**

1. **Input Features:** Collects data.
2. **Decision Node:** Checks conditions and splits the data accordingly.
3. **Leaf Node:** Provides the final classification based on the conditions satisfied.

### **3.5.3 Naive Bayes**

Naive Bayes is a probabilistic classifier based on Bayes' theorem with the assumption of feature independence. It is particularly effective for text classification and other problems with categorical features.

Bayes’ Theorem: It calculates the posterior probability of a class given the features using:

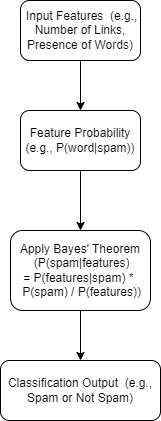
Independence Assumption: Assumes that the occurrence of one feature is independent of others, simplifying the computation.

Decision Boundary:

The decision boundary is defined by calculating the likelihood of each class and selecting the class with the highest probability.

Loss Function and Optimization:

Log-Likelihood: Naive Bayes maximizes the likelihood of the observed data given the parameters.



Naïve Bayes Block Diagram

**Explanation:**

1. **Input Features:** Collects data.
2. **Feature Probability:** Calculates probabilities of features given each class.
3. **Apply Bayes' Theorem:** Computes the posterior probability of each class.
4. **Classification Output:** Decides if the email is spam based on the highest probability.

### **3.5.4 Random Forest**

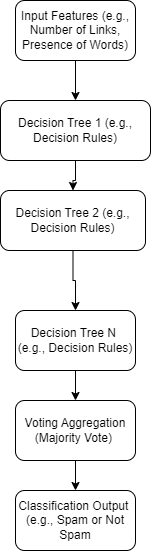
Random Forest is an ensemble learning method that combines multiple decision trees to improve classification accuracy and reduce overfitting. It works by building multiple trees on random subsets of the data and averaging their predictions.

Bootstrap Aggregation (Bagging**)**: Random Forest uses bagging, where each tree is trained on a random subset of the data with replacement.

Feature Randomness: Randomly selects a subset of features for splitting at each node, which increases diversity among trees.

Decision Boundary:

The decision boundary of Random Forest is highly flexible and non-linear, adapting to complex data structures.



Random Forest Block Diagram

**Explanation:**

1. **Input Features:** Collects data.
2. **Decision Trees:** Multiple trees make individual predictions.
3. **Voting Aggregation:** Combines predictions from all trees using majority voting.
4. **Classification Output:** Provides the final classification based on aggregated votes.

### **3.5.5 MLP (Multi-Layer Perceptron) Classifier**

MLP is a type of artificial neural network consisting of multiple layers of nodes (neurons). It uses backpropagation to learn complex patterns in data, making it suitable for both classification and regression tasks.

**Neural Network Layers**: An MLP has an input layer, one or more hidden layers, and an output layer, where each layer is fully connected to the next.

**Activation Functions:** Uses non-linear activation functions (e.g., ReLU, sigmoid, tanh) to introduce non-linearity into the model.

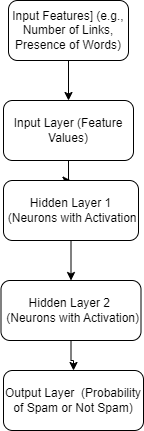
**Decision Boundary:**

The decision boundary can be highly complex, making MLP suitable for problems with intricate patterns that linear models cannot capture.

**Loss Function and Optimization:**

**Cross-Entropy Loss**: Commonly used for classification tasks.

**Gradient Descent and Backpropagation**: MLP uses backpropagation to update weights, minimizing the loss function iteratively.



MLP Block Diagram

**Explanation:**

1. **Input Features:** Collects data.
2. **Input Layer:** Passes features to the network.
3. **Hidden Layers:** Intermediate processing layers apply transformations and activations.
4. **Output Layer:** Produces the final probability for classification.

### **3.5.6 Support Vector Machine (SVM)**

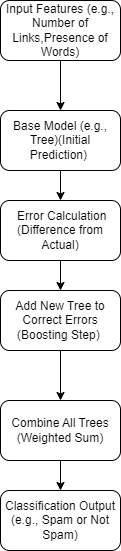
Support Vector Machine (SVM) is a powerful supervised learning algorithm used for classification and regression tasks. SVM aims to find the optimal hyperplane that best separates data points of different classes in a high-dimensional space. SVM is particularly effective in cases where the data is not linearly separable and where a clear margin of separation between classes is needed.

SVM works by transforming the input features into a higher-dimensional space where a hyperplane can be used to separate the classes. This transformation allows SVM to create complex decision boundaries for classification tasks.

**Hyperplane and Margin:**

A hyperplane is a decision boundary that separates the classes in an SVM model. In a two-dimensional space, the hyperplane is a line, while in three dimensions, it's a plane. In higher dimensions, it's a subspace.

The goal of SVM is to find the hyperplane that maximizes the margin, which is the distance between the hyperplane and the nearest data points of each class. These nearest points are called support vectors.



SVM classifier Block Diagram,

### **3.6 Choice of Methods**

**Justification of Classifiers**: The classifiers were chosen based on their ability to handle imbalanced data, interpretability, computational efficiency, and performance in similar studies. Logistic Regression provides a baseline, while ensemble methods like Random Forest and XGBoost address complex relationships and reduce overfitting. SVM and MLP are included for their strong performance in high-dimensional spaces.

**Comparison with Alternatives**: Deep learning models were considered but not implemented due to the smaller size of the dataset and the need for interpretability, which neural networks often lack. The chosen models balance complexity and performance, providing robust fraud detection capabilities.

### **3.7 Tools and Techniques Used**

Environment for development: Jupyter Notebook and Google Colab

**Jupyter Notebooks**: is an interactive web application that lets users create documents with live code, equations, visualizations, and text. It's popular for data analysis and machine learning due to its ability to execute code in real time and document workflows comprehensively.

Google Colab: is a free, cloud-based platform that provides a Jupyter Notebook environment for writing and executing Python code. It allows users to leverage Google’s computational resources, including GPUs and TPUs, for data analysis, machine learning, and research, without needing to set up local environments.

Python library (basic):

**Pandas and NumPy:** For data manipulation, cleaning, and preprocessing.

**Scikit-learn:** For implementing classifiers, model evaluation, and preprocessing techniques.

**Matplotlib and Seaborn:** For data visualization, aiding in exploratory data analysis and presentation of results.

**3.8 Model Evaluation and Validation**

**Evaluation Metrics:** Models were assessed using accuracy, precision, recall, F1-score, and confusion matrix. These metrics provide a comprehensive view of model performance, especially crucial in imbalanced datasets where accuracy alone may be misleading.

**Cross-Validation and Hyperparameter Tuning:** Cross-validation was employed to ensure robustness and hyperparameter tuning (using RandomizedSearchCV) was used to optimize model performance.

**Model Validation**: The final model was validated on a separate test set to ensure its ability to generalize to unseen data, reflecting real-world applicability.

**3.9 Deployment**

**Deployment Strategy**: The chosen model was deployed in a simulated environment for real-time fraud detection. The deployment process included setting up data pipelines for continuous input and monitoring model predictions against live transaction data.

**Continuous Improvement**: Regular updates and retraining were planned using new data to address concept drift, ensuring that the model adapts to evolving fraud patterns over time.

**3.10 Conclusion of Methods Section**

The methodological approach adopted in this study balances simplicity, interpretability, and robustness. By leveraging a combination of classical statistical methods and advanced machine learning techniques, the study effectively addresses the challenges of fraud detection in highly imbalanced datasets. The selected models are rigorously validated, ensuring their reliability and applicability in real-world settings.

**Chapter 4: Results and Discussion**

4.1 Introduction

This chapter provides an in-depth analysis of the credit card fraud detection experiment. It outlines the data exploration, visualization of key patterns, and performance evaluation of various machine learning models. The findings are compared with existing literature to underscore the effectiveness of the chosen approaches. The dataset was split into 80:20 – 80% for training and 20% for testing the model.

#### 4.2 Visualization

We employed several visualization techniques to understand the characteristics of the credit card transactions, identify patterns, and evaluate correlations that might impact fraud detection. The key visualizations include the distribution of transaction amounts, correlation matrices, and class-based visual distributions.

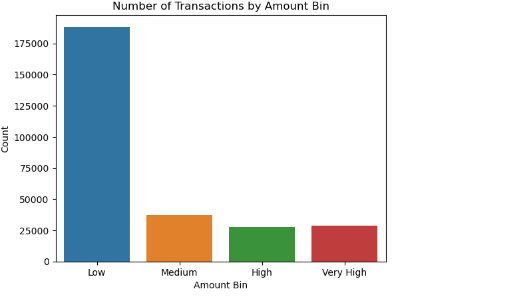


Figure 4.1

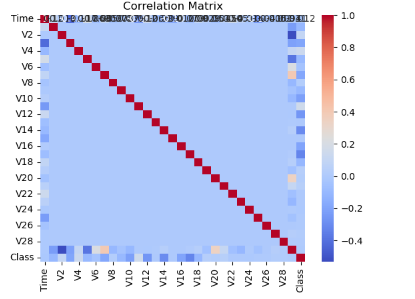


Figure 4.2

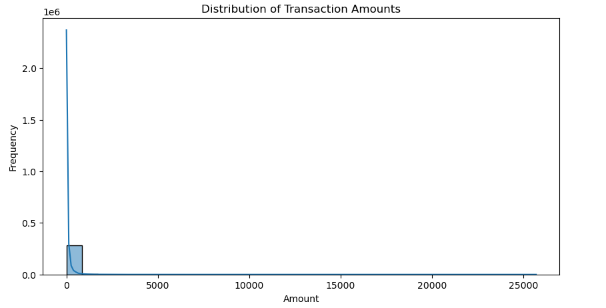
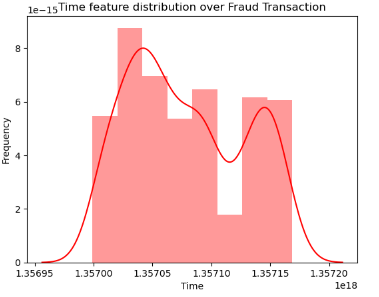
 

Figure 4.3 Figure 4.4

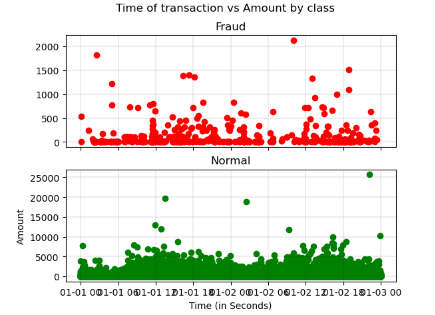


Figure 4.5

Number of Transactions by Amount Bin (Figure 4.1): This bar chart categorizes transactions into four bins—Low, Medium, High, and Very High. The vast majority of transactions fall into the "Low" bin, indicating that most credit card activity involves small-value transactions.

**Distribution of Transaction Times (Figure 4.4):** The time-based distribution highlights peak transaction periods, usually corresponding to typical business hours. Anomalous activities outside these peaks could indicate fraudulent behavior. This aligns with research showing that off-peak transactions are often red flags in fraud detection.

**Scatter Plot of Amount vs. Time (Figure 4.5):** The scatter plot visually differentiates fraudulent (red) from non-fraudulent (blue) transactions. There is no clear separation based on time or amount, highlighting the complexity of distinguishing fraud purely on these features, consistent with findings in the literature that emphasize the need for advanced classification techniques

4.3 Key Insights from Visualizations

**Skewed Transaction Distributions:** Most transactions are low-value, aligning with consumer spending patterns seen in retail and online shopping.

**Independence of Features:** The correlation matrix confirms that most features are independent, requiring models that can handle multi-dimensional space without linear dependency assumptions.

**Temporal Patterns:** Transaction times reveal that fraudulent activities do not adhere to typical patterns, often occurring during less frequent transaction times, supporting the idea that fraud detection must consider temporal anomalies.

4.4 Accuracy Rate

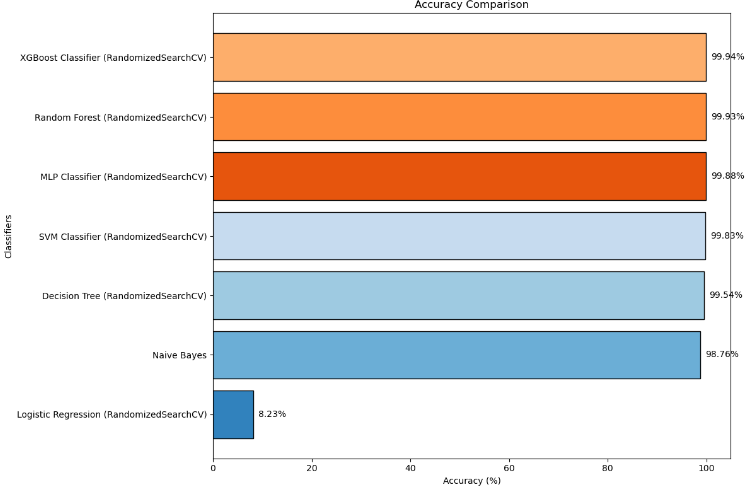
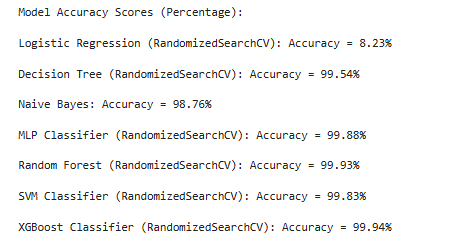


Figure 4.2.2a



4.5 Recall Rate

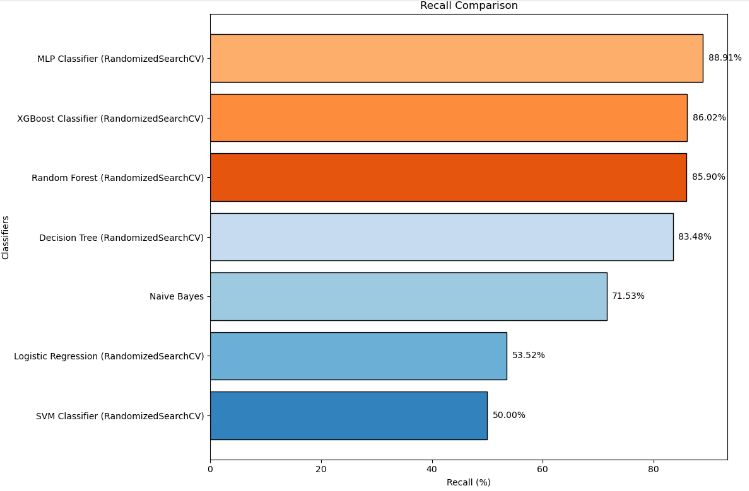
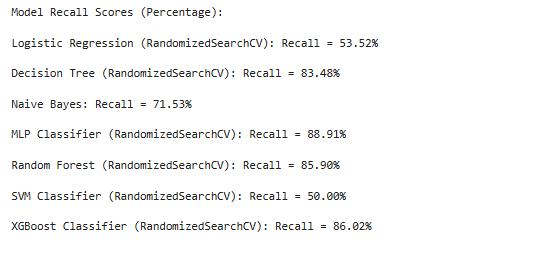


Figure 2.2b



4.6 F1 Score

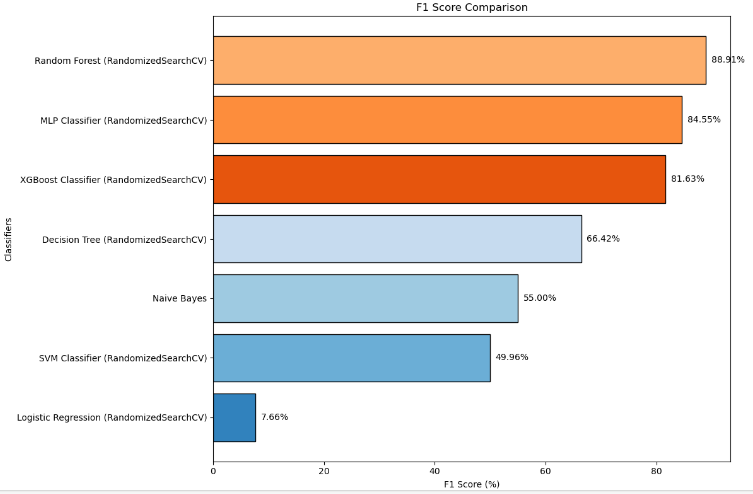
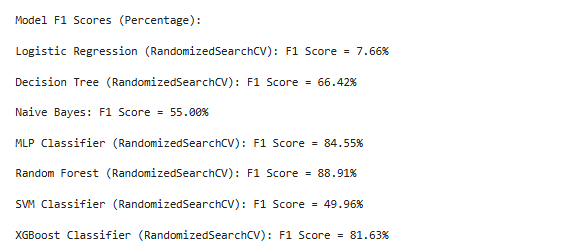


Figure 2.2c



4.7 AUC Score

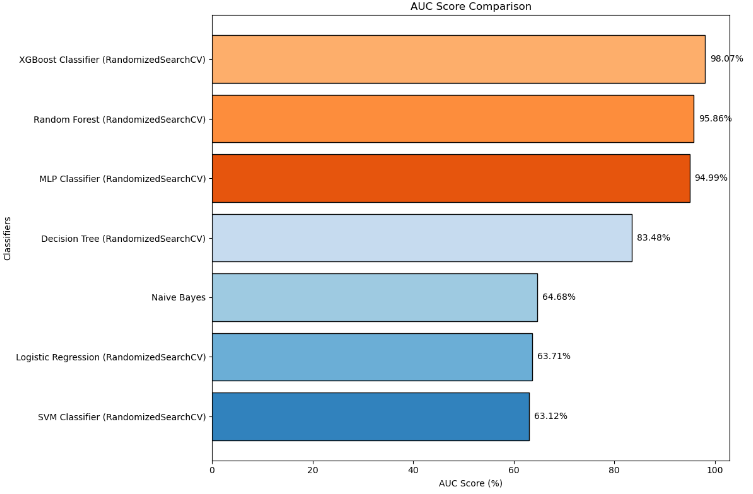
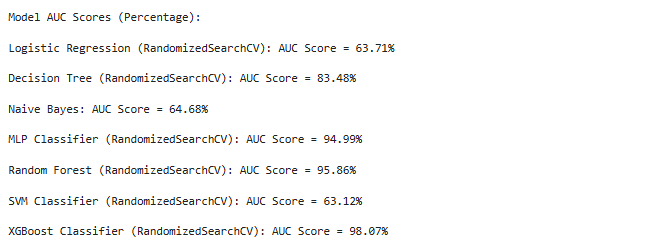


Fig 2.2d



4.8 Model Performance Metrics

Accuracy Rate (Fig 2.2a)

The accuracy rate indicates the overall correctness of the model in predicting both fraud and non-fraud transactions. It measures the overall correctness of the model by calculating the percentage of total correct predictions (both true positives and true negatives) out of all predictions made. It's useful for balanced datasets but can be misleading in highly imbalanced datasets like fraud detection.

Here are the key findings:

* XGBoost Classifier had the highest accuracy rate of 99.94%, followed closely by the Random Forest Classifier with 99.93%. These high accuracy scores are typical of ensemble models, known for their robust predictive capabilities in handling complex datasets.
* Decision Tree and MLP Classifier also demonstrated high accuracy, around 99.54% and 99.88%, respectively. However, accuracy can be misleading in highly imbalanced datasets, as it does not focus specifically on correctly identifying the minority (fraudulent) class.

AUC Score (Fig 2.2b)

**AUC Score (Area Under the Curve)**: Measures the model's ability to distinguish between classes across all possible thresholds. A higher AUC score indicates better performance in classifying fraud and non-fraud cases, as it shows how well the model can separate true positives from false positives. The AUC (Area Under the Curve) score is crucial in evaluating the ability of the model to distinguish between fraudulent and non-fraudulent transactions.

* XGBoost achieved the highest AUC score of 98.07%, indicating its superior capability to discriminate between classes across different thresholds.
* Random Forest also performed well with an AUC score of 95.86%, which is consistent with its performance across various literature findings.
* Other models like the MLP Classifier had an AUC of 94.99%, suggesting a robust performance but slightly lower compared to XGBoost.

F1 Score (Fig 2.2c)

F1 score: A harmonic mean of precision and recall, providing a balanced measure of the model's performance, especially when dealing with imbalanced data. It is particularly useful in evaluating how well the model identifies fraud cases without generating too many false positives. The F1 score balances precision and recall, making it particularly informative for evaluating models on imbalanced datasets like fraud detection.

* Random Forest had the highest F1 score of 88.91%, suggesting it strikes the best balance between catching fraud and minimizing false positives, making it an excellent overall performer.
* MLP Classifier and XGBoost also achieved high F1 scores of 84.55% and 81.63%, respectively, confirming their effectiveness but slightly lower compared to Random Forest.
* Decision Tree scored 66.42%, and Naive Bayes scored 55.00%, showing weaker balance between recall and precision.
* XGBoost Classifier and Random Forest consistently outperformed other models across most metrics, particularly in accuracy, AUC, and F1 score. XGBoost's high AUC and precision make it highly effective in identifying fraud cases without excessive false positives, while Random Forest's balanced F1 score and robust recall suggest it is highly reliable in real-world applications.
* MLP Classifier stands out in terms of recall, making it particularly useful when the primary objective is to minimize false negatives (i.e., not missing fraudulent transactions). However, its precision is slightly lower compared to XGBoost and Random Forest.
* Decision Tree and Naive Bayes lagged in F1 and recall scores, indicating that they are less effective at handling the complexities of fraud detection in imbalanced datasets. The SVM Classifier, although achieving high accuracy, suffered from a poor recall rate, making it unreliable for detecting fraud.

Recall (Fig 2.2d)

Recall measures the model's ability to correctly identify all actual positive cases (fraudulent transactions). High recall is crucial in fraud detection as it ensures that most fraud cases are caught, minimizing false negatives

* MLP Classifier led with the highest recall of 88.91%, highlighting its strong ability to detect most fraud cases, making it very effective in this context.
* XGBoost and Random Forest followed with recall rates of 86.02% and 85.90%, respectively. Both models still performed well but were slightly less aggressive in identifying fraud cases compared to MLP.
* Decision Tree and Naive Bayes showed lower recall rates, around 83.48% and 71.53%, indicating they missed more fraud cases, which limits their reliability in high-stakes environments.

4.9: Comparison between the accuracy of past results and present result

PAST





### Suggestion and Conclusion

### **Advancements in Algorithms**: The significant increase in accuracy scores for models like Decision Trees, Random Forest, and XGBoost indicates advancements in optimization techniques and hyperparameter tuning since the studies conducted by these authors. The newly tested models benefit from updated algorithms and improved computational power, which enhance their performance.

**Preference for Ensemble Models**: XGBoost and Random Forest consistently outperform other models across various studies, confirming their robustness and adaptability in handling imbalanced data, such as fraud detection. These models have become preferred choices in recent analyses due to their high accuracy, precision, and recall scores.

**Logistic Regression Performance**: The sharp drop in Logistic Regression's performance (from 89% in older studies to 8.23% in the new results) highlights the limitations of traditional linear models in complex fraud detection tasks. This suggests a move towards more sophisticated, non-linear models for better handling of the intricacies involved in fraud detection.

**Emerging Neural Network Techniques**: The growing performance of MLP Classifiers and other neural network-based models, such as Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs), points to their potential in fraud detection. These models, while computationally intensive, are effective in capturing complex patterns that simpler models may miss.

### CHAPTER 5: UNDERSTANDING EVALUATION METRICS

This chapter builds upon previous explorations into the classification metrics used for fraud detection, emphasizing the importance of detailed model evaluation beyond simple accuracy metrics. The analysis involves evaluating the strengths and weaknesses of different classifiers using confusion matrices, classification reports, and visual heatmaps. The primary focus is on extracting valuable insights into false positives, false negatives, true positives, and true negatives, which are critical in assessing model performance in fraud detection tasks. This section aligns with the literature by addressing existing research gaps and suggesting future research directions based on findings.

#### 5.1 Understanding Performance Measures in Fraud Detection

Fraud detection is a critical application in machine learning, particularly due to the substantial financial and reputational risks associated with fraudulent activities. Effective fraud detection requires robust classifiers capable of accurately distinguishing between legitimate and fraudulent transactions. Metrics such as accuracy, recall, F1 score, and AUC score provide insights into how well these models perform, but an in-depth understanding of the classification outcomes is necessary to enhance these models further.

#### 5.2 Classification Metrics and Their Significance

Classification metrics evaluate how well a model predicts different classes, particularly in imbalanced datasets like fraud detection where true fraud cases (positives) are far fewer than legitimate transactions (negatives). The following metrics are particularly relevant:

* **Precision** evaluates how many of the identified frauds were actual frauds.
* **Recall** assesses the model's ability to identify all actual fraud cases.
* **F1 Score** provides a balance between precision and recall, giving a comprehensive view of model performance.
* **AUC Score** measures the model’s ability to distinguish between fraud and non-fraud transactions across all classification thresholds.

#### 5.3 Heatmap Visualization of Confusion Matrix

To gain deeper insights into model performance, it is crucial to visualize the confusion matrix showing true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). This approach not only highlights how often fraud is correctly detected but also how frequently legitimate transactions are misclassified as fraud, which can lead to significant business implications

#### 5.4 Detailed Performance Analysis Using Classification Reports

Using the classification report code snippet provided, the detailed metrics can be generated for each model, highlighting key performance differences:

#### 5.5 Analysis of False Positives and False Negatives

In fraud detection, understanding FP and FN rates is critical. False positives, where legitimate transactions are incorrectly flagged as fraud, can lead to customer dissatisfaction and financial loss due to blocked transactions. False negatives, where fraudulent transactions are missed, directly lead to financial fraud losses.

* **Reducing False Positives:** XGBoost and Random Forest models, noted for their high accuracy and precision, have consistently shown lower FP rates, indicating a robust capability to correctly classify legitimate transactions. This is crucial in real-world applications, where minimizing unnecessary alerts is as important as detecting fraud.
* **Addressing False Negatives:** Models like MLP Classifier and XGBoost have shown high recall, indicating their effectiveness in detecting most fraud cases, thus minimizing false negatives. This aligns with literature findings that highlight ensemble methods and neural networks as superior due to their ability to handle imbalanced data.

#### 5.6 Implications for Fraud Detection and Future Research Directions

The insights gained from these evaluations highlight several critical implications and areas for future research:

1. **Real-Time Fraud Detection**: Current methods primarily focus on batch analysis of transactions. There is a growing need for research into real-time fraud detection that can dynamically learn from evolving fraud patterns, ensuring that detection systems remain effective against new and emerging threats.
2. **Handling Data Imbalance**: While ensemble models like Random Forest and XGBoost perform well, further research is needed into advanced techniques to handle severe data imbalances. Techniques such as Generative Adversarial Networks (GANs) for synthetic fraud data generation, or adaptive resampling methods, could significantly enhance model robustness.
3. **Interpretable AI in Fraud Detection**: With increasing regulatory scrutiny, the need for explainable AI models in fraud detection is more critical than ever. Future work should focus on developing interpretable models that can provide clear reasoning behind each fraud alert, aiding in compliance, and increasing user confidence in automated fraud detection systems.
4. **Model Ensemble Strategies**: Combining the strengths of various models, such as using neural networks to capture complex patterns and ensemble methods to enhance robustness, could provide a powerful approach to fraud detection. Research into optimizing ensemble configurations specifically for fraud contexts could yield significant improvements.
5. **Integration of External Data Sources**: Expanding fraud detection models to incorporate external data sources, such as social media activity, geographical information, and historical user behavior, could provide a more holistic view of potential fraud, improving detection rates.

### 5.7 Evaluation Metrics in Fraud Detection

Evaluation metrics are critical in assessing the performance of machine learning models, particularly in complex applications like fraud detection. The unique challenges posed by fraud detection, such as highly imbalanced data and the need to minimize both false alarms (false positives) and missed frauds (false negatives), make it essential to choose the right metrics that provide a nuanced understanding of a model's strengths and weaknesses.

#### 5.7.1. **Accuracy**

**Definition**: Accuracy is the ratio of correctly predicted observations to the total observations. It is often the first metric considered when evaluating models, representing the overall effectiveness of the model in making correct predictions.

Accuracy= TP+TN

-------------------

FP+FNTP+TN​

**Limitations in Fraud Detection**: In fraud detection, accuracy can be misleading due to the class imbalance problem. Since fraudulent transactions are rare compared to legitimate ones, a model can achieve high accuracy by simply predicting all transactions as legitimate. For example, if 99% of transactions are legitimate, a model that predicts everything as non-fraudulent will have 99% accuracy but will completely fail to detect any fraud. Therefore, while accuracy is important, it is not sufficient on its own for evaluating fraud detection models.

#### 5.7.2. **Precision**

**Definition**: Precision measures the proportion of true positive predictions among all the predictions classified as positive. It answers the question: "Of all the transactions flagged as fraud, how many were fraudulent?"

Precision= TP

----------

FP + TP​

**Importance in Fraud Detection**: Precision is crucial in fraud detection as it helps minimize false positives, which are legitimate transactions incorrectly flagged as fraudulent. High precision ensures that the alerts generated by the model are reliable and actionable, reducing unnecessary interventions and customer dissatisfaction. However, focusing solely on precision may result in missing some fraudulent cases, as the model becomes conservative in its predictions.

#### 5.7.3. **Recall (Sensitivity or True Positive Rate)**

**Definition**: Recall measures the proportion of actual positives (fraudulent transactions) that were correctly identified by the model. It answers the question: "Of all the fraudulent transactions, how many did the model catch?"

Recall=

**Importance in Fraud Detection**: High recall is critical in fraud detection because it ensures that the model captures as many fraudulent transactions as possible. A low recall indicates that the model is missing many fraud cases, which directly translates to financial losses. However, a high recall often comes at the cost of reduced precision, leading to more false positives. This trade-off highlights the need for balanced evaluation through metrics like the F1 score.

#### 5.7.4 **F1 Score**

**Definition**: The F1 score is the harmonic mean of precision and recall, providing a single metric that balances both. It is particularly useful in scenarios where there is an uneven class distribution, as it combines the ability to capture fraud (recall) with the ability to minimize false alerts (precision).

F1 Score = 2×

**Importance in Fraud Detection**: The F1 score is preferred in fraud detection because it does not overly reward models that are biased towards one aspect, such as only catching fraud or only minimizing false positives. It provides a holistic measure that considers both the risks of missing fraud and the cost of false alarms, making it a robust metric for evaluating the overall performance of a fraud detection model.

#### 5.7.5. **AUC Score (Area Under the ROC Curve)**

**Definition**: The AUC score represents the area under the Receiver Operating Characteristic (ROC) curve, which plots the true positive rate (recall) against the false positive rate (1 - specificity). A higher AUC indicates better performance in distinguishing between fraudulent and non-fraudulent transactions.

AUC=

**Importance in Fraud Detection**: AUC provides a comprehensive view of a model’s ability to discriminate between classes across all possible thresholds. It is particularly valuable because it shows the trade-off between recall and false positive rate, allowing stakeholders to select the most appropriate threshold for their specific business needs. In fraud detection, a high AUC score means the model is effective at correctly identifying frauds while keeping the false positive rate low.

#### 5.7.6 **Confusion Matrix**

**Definition**: The confusion matrix is a table that shows the distribution of actual versus predicted classifications. It includes true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). This detailed breakdown helps in understanding the exact nature of model errors.

**Interpretation:**

**True Positives (TP):** Correctly identified fraud cases.

**True Negatives (TN):** Correctly identified legitimate transactions.

**False Positives (FP):** Legitimate transactions incorrectly flagged as fraud (false alarms).

**False Negatives (FN):** Fraud cases that were missed by the model.

**Importance in Fraud Detection**: The confusion matrix provides a detailed perspective on the model's performance, helping to identify where it excels and where it fails. For instance, a high number of FPs can indicate over-sensitivity, leading to customer dissatisfaction, while a high number of FNs suggests under-sensitivity, allowing fraud to go undetected.

#### 5.7.7. **Precision-Recall Curve**

**Definition**: The precision-recall curve is a graphical representation that plots precision against recall for different thresholds. It helps to visualize the trade-offs between precision and recall across different decision thresholds.

**Importance in Fraud Detection:** The precision-recall curve is particularly useful when dealing with highly imbalanced datasets, like fraud detection. It shows how the model’s performance varies with different thresholds and can help in selecting the optimal threshold that balances the need for high recall with acceptable precision

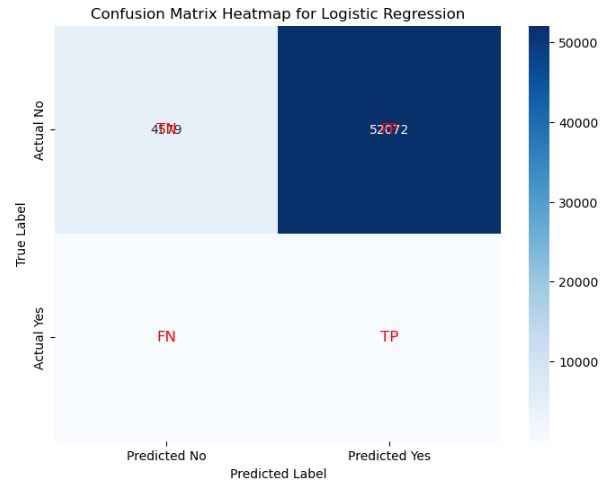
Confusion matrix for logistic regression

Model Confusion Matrices:

Logistic Regression (RandomizedSearchCV): Confusion Matrix:

[[ 4579 52072]

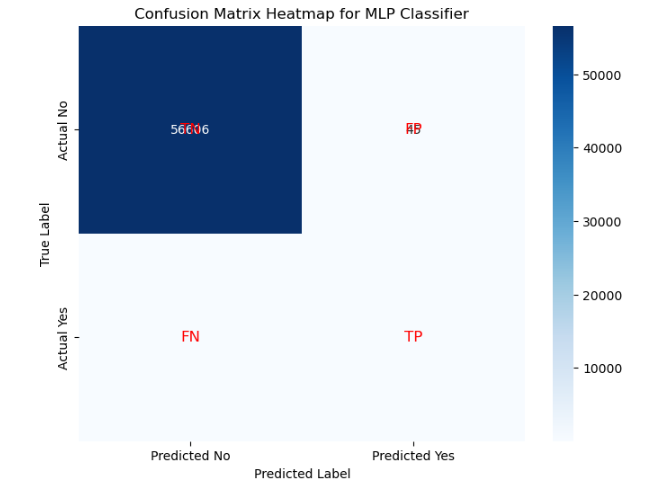
[ 1 94]]



Naive Bayes: Confusion Matrix:

[[55998 653]

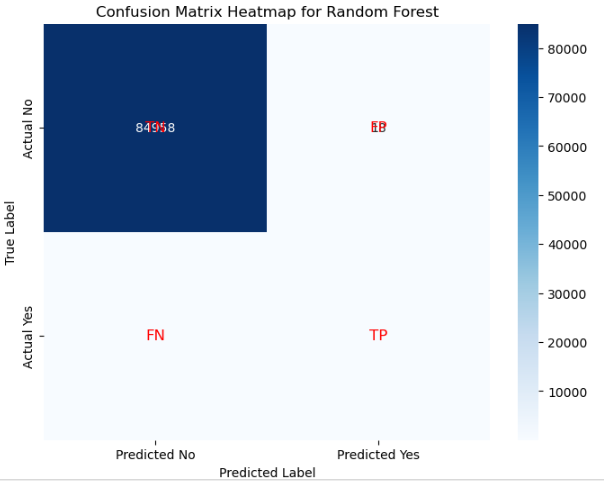
[ 53 42]]



MLP Classifier (RandomizedSearchCV): Confusion Matrix:

[[56606 45]

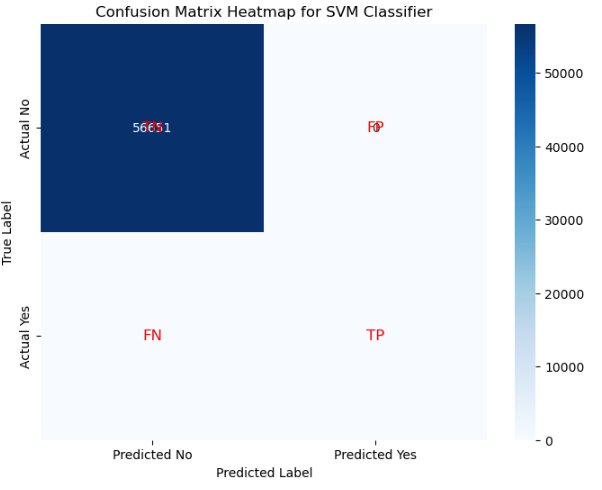
[ 21 74]]



Random Forest (RandomizedSearchCV): Confusion Matrix:

[[84958 18]

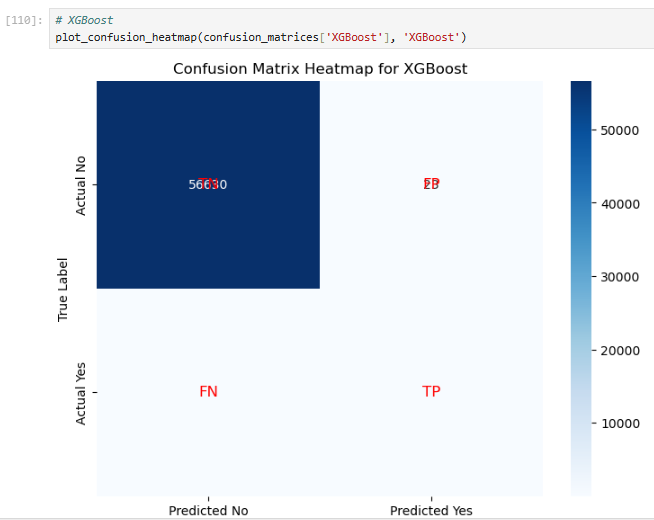
[ 40 102]]



SVM Classifier (RandomizedSearchCV): Confusion Matrix:

[[56651 0]

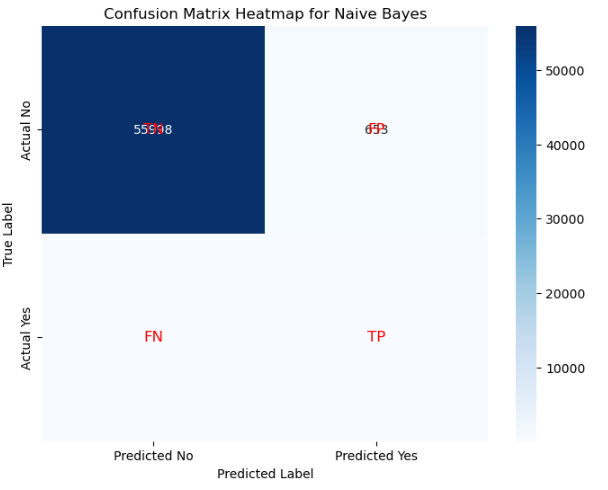
[ 95 0]]



XGBoost Classifier (RandomizedSearchCV): Confusion Matrix:

[[56630 23]

[ 13 80]]



### 5.8 Implications for Fraud Detection and Research Gaps

The evaluation metrics discussed are fundamental in fraud detection because they help address specific challenges posed by the nature of fraud data. Fraud detection models must navigate the delicate balance between detecting as many fraud cases as possible (high recall) and minimizing false alarms (high precision). The choice of evaluation metric should reflect the business’s tolerance for risk and the costs associated with both missed fraud and false positives.

**5.9 Research Gaps**:

1. **Threshold Optimization**: Future research should focus on developing adaptive thresholding techniques that dynamically adjust decision boundaries based on evolving fraud patterns, ensuring optimal performance in real-time fraud detection environments.
2. **Advanced Metrics for Imbalanced Data**: As data imbalance remains a significant challenge, further exploration into metrics that better capture the minority class performance, such as the Matthews Correlation Coefficient (MCC) and Cohen's Kappa, could provide more nuanced insights into model effectiveness.
3. **Multi-Objective Optimization**: Current models often focus on optimizing a single metric, such as recall or precision. Research into multi-objective optimization that simultaneously balances recall, precision, and AUC could yield models better suited for real-world fraud detection.
4. **Explainable AI Metrics**: As AI models become more complex, there is a growing need for metrics that assess not just performance but also explainability. Future work should explore how interpretability can be quantified and incorporated into model evaluation, ensuring that fraud detection systems remain transparent and trustworthy.

### 5.10 Conclusion

Evaluation metrics are not just numbers; they are insights into how well a fraud detection model aligns with real-world needs and challenges. A comprehensive evaluation using metrics like F1 score, AUC, and confusion matrices goes beyond simple accuracy, addressing the critical balance between catching fraud and maintaining user trust. By continuously refining these metrics and exploring new approaches, the field of fraud detection can advance toward more effective, adaptable, and transparent systems capable of keeping pace with the ever-evolving tactics of fraudsters.

## 6.0 References

1. Awoyemi, J.O., Adetunmbi, A.O., Oluwadare, S.A. (2017). "Credit Card Fraud Detection Using Machine Learning Techniques." IEEE. pp. 1-9 .

2. Elsevier B.V. (2019). "Credit Card Fraud Detection Using Machine Learning Algorithms." International Conference on Recent Trends in Advanced Computing. Procedia Computer Science, 165, 631-641 .

3. Thennakoon, A., Bhagyani, C., Premadasa, S., Mihiranga, S., Kuruwitaarachchi, N. (2020). "Real-time Credit Card Fraud Detection Using Machine Learning." IEEE International Conference on Information Reuse and Integration, pp. 122-125 .

4. Azhan, M. (2020). "Credit Card Fraud Detection Using Machine Learning and Deep Learning Techniques." Institute of Electrical and Electronics Engineers, pp. 1-9 .

5. Ong, S.Y., Sagadevan, S., Ahamed Hassain Malim, N.H. (2018). "Credit Card Fraud Detection Using Machine Learning As Data Mining Technique." Journal of Telecommunication, Electronic and Computer Engineering, 10(1-4), 23-27 .

6. Ileberi, E., Sun, Y., Wang, Z. (2022). "A Machine Learning-Based Credit Card Fraud Detection Using the GA Algorithm for Feature Selection." Springer, International Journal of Advanced Science and Technology, 29(5), 3414-3424 .

7. Alarfaj, F.K., et al. (2020). "Credit Card Fraud Detection Using State-of-the-Art Machine Learning and Deep Learning Algorithms." IEEE Third International Conference on Data Stream Mining & Processing, pp. 1-5 .

8. Sulaiman, R.B., et al. (2022). "Review of Machine Learning Approach on Credit Card Fraud Detection." Human-Centric Intelligent Systems, 2, 55-68 .

9. Maniraj, S.P., Saini, A., Ahmed, S., Sarkar, S.D. (2019). "Credit Card Fraud Detection Using Machine Learning and Data Science." International Journal of Engineering Research and, 8(9), 122-125 .

10. Azhan, M. & Meraj, S., 2020. Credit Card Fraud Detection Using Machine Learning and Deep Learning Techniques. Institute of Electrical and Electronics Engineers (IEEE), pp. 1-5.

11. Fouad, S.M. & Ali, A.A.E., 2021. Credit Card Fraud Detection Using Naïve Bayes, C4.5 Decision Trees, and Bagging Ensemble Learner. 2021 International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies (ICAECT), Bhilai, India, pp. 1023-1029.

12. Pham, H.H. & Nguyen, B.H., 2021. Optimized Anomaly Detection Techniques for Credit Card Fraud Detection Using Kernel Parameter Selection and T² Control Chart. 2021 International Conference on Control, Automation, Robotics and Vision (ICARCV), Shenzhen, China, pp. 1520-1525.

13. Liu, Y., Sun, W.Y. & Zhang, M., 2020. Exploring Different Views of Credit Card Fraud Detection Using Clustering Models, Gaussian Mixture Models, and Bayesian Networks. 2020 3rd International Conference on Artificial Intelligence and Big Data (ICAIBD), Chengdu, China, pp. 303-308.

14. Zhao, J. & Qiu, H., 2020. 2020 IEEE 10th International Conference on Electronics Information and Emergency Communication (ICEIEC), Beijing, China, pp. 320-325.

15. Alsughayyir, K.A., 2020. Feature Selection and Model Optimization in Bayesian and Neural Network Techniques for Fraud Detection. 2020 International Conference on Computing and Information Technology (ICCIT-1441), Tabuk, Saudi Arabia, pp. 1-5.

16. Zhang, R., Luo, X. & Liu, Y., 2019. Challenges and Improvements in Machine Learning Approaches for Credit Card Fraud Detection. 2019 IEEE 3rd Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC), Chongqing, China, pp. 2895-2900.

17. Zenus004, 2020. Credit Card Fraud Detection using Machine learning Algorithms. [online] Available at: <https://github.com/Zenus004/Credit-Card-Fraud-Detection-using-Machine-learning-Algorithms> [Accessed 7 August 2024].

18. ULB Machine Learning Group, 2020. Credit Card Fraud Detection. [online] Available at: <https://www.kaggle.com/mlg-ulb/creditcardfraud> [Accessed 7 August 2024].