A PROJECT REPORT ON

CREDIT CARD FRAUD DETECTION USING MACHINE LEARNING

SUBMITTED IN PARTIAL FULFILLMENT FOR THE DEGREE

OF

MASTER OF COMPUTER SCIENCE

BY

TAMYA BHARDWAJ

(ROLL NO.- 230719)

UNDER THE GUIDANCE OF

Dr. ANAMIKA SHUKLA SHARMA

HEAD

(DEPARTMENT OF COMPUTER SCIENCE)

GOVT. E. RAGHAVENDRA RAO P.G. SCIENCE COLLEGE BILASPUR

(CHHATTISGARH INDIA)

SESSION 2024-25

DEPARTMENT OF COMPUTER SCIENCE

GOVT. E. RAGHVENDRA RAO P.G. SCIENCE COLLEGE BILASPUR(C.G.)

SESSION 2024-25

A PROJECT REPORT ON

CREDIT CARD FRAUD DETECTION USING MACHINE LEARNING

BY

TAMYA BHARDWAJ

SUBMITTED IN PARTIAL FULFILLMENT FOR THE DEGREE

OF

MASTER OF COMPUTER SCIENCE

UNDER THE GUIDANCE OF

Dr. ANAMIKA SHUKLA SHARMA

HEAD

(DEPT. OF COMPUTER SCIENCE)

SUBMITTED TO

Dr. ANAMIKA SHUKLA SHARMA

HEAD

(DEPT. OF COMPUTER SCIENCE)

CERTIFICATE

This is to certify that the project entitled "**CREDIT CARD FRAUD DETECTION USING MACHINE LEARNING ”** is carried out by **Tammya (230719)** under my guidance and supervision for award of the degree Master of Computer Science, Department of Computer Science, Government E. Raghavendra Rao PG Science College, Bilaspur (CG).

To the best of my knowledge and belief the project: -

1. Embodies the work of the candidate herself.
2. Has duly been completed in the specified time.
3. Fulfils the requirements of the ordinance relating to M.Sc. degree of the college, and
4. Is up to the standard both in respect of content and language for being referred to the examiners.

Project Guide: Head of the department

Dr. ANAMIKA SHUKLA SHARMA **Dr. ANAMIKA SHUKLA SHARMA**

Head (Department of Computer Science)

DECLARATION

I declare that the project work entitled "**CREDIT CARD FRAUD DETECTION USING MACHINE LEARNING "** has been carried by me under the supervision of internal guide **Dr. Anamika Shukla Sharma**, at the Department of Computer Science, Government E. Raghavendra Rao PG Science College Bilaspur (CG).

I further declare that this is an original work carried by me and to the best of my knowledge this project report does not contain any part of any work, that has been submitted for the award of any degree either in this University or in any other University / Deemed university without proper citation.

.

Tamya Bhardwaj

M.sc IV Sem (Computer Science)

(220722)

(Govt. E. Raghavendra Rao P.G. Science College

Bilaspur (C.G.))

ACKNOWLEDGEMENT

This acknowledgement transcends the reality of formality when I would like to express deep gratitude and respect to all those people behind the scenes who guided, inspired and helped me for the completion of my project work. I consider myself fortunate enough to get such an outstanding project. This project would add as an asset to my academic profile, and I take this opportunity to express my sincere thanks to **Dr. Anamika Shukla Sharma,** Head (Department of Computer Science), Government E. Raghvendra Rao Post Graduate Science College, Bilaspur (C.G), for providing me a golden opportunity to work on this project and for her able guidance, valuable suggestion and discussion whenever required. I also express special thanks to **Dr. Kajal Kiran Gulhare**, Head (Department of Information Science), Government E. Raghvendra Rao Post Graduate Science College, Bilaspur (C.G), for her constant motivation and support. Finally, I express my indebtedness to **Mr. Harish Dewangan**, Assistant Professor (Department of Computer Science), Government E. Raghvendra Rao Post Graduate Science College, Bilaspur (C.G), and **Mr. Manishankar,** Assistant Professor (Department of Computer Science), Government E. Raghvendra Rao Post Graduate Science College, Bilaspur (C.G) for their valuable suggestions and co-operation without which it could not have been completed in time.

Tamya Bhardwaj

M.sc IV Sem

(Computer Science)

ABSTRACT

Credit card fraud has become a major challenge in the digital financial ecosystem, leading to significant financial losses and undermining trust in electronic payment systems. This thesis presents a comprehensive study on the detection of fraudulent credit card transactions using machine learning techniques. The study investigates multiple classification algorithms—including Decision Trees, Random Forests, Logistic Regression, and ensemble methods like AdaBoost—to identify patterns indicative of fraud in highly imbalanced datasets. Emphasis is placed on addressing the imbalance problem using techniques such as oversampling (SMOTE) and undersampling. The performance of models is evaluated using metrics beyond accuracy, such as precision, recall, F1-score, and area under the ROC curve (AUC), to ensure robustness in detecting rare fraudulent activities. Real-world datasets are used to simulate authentic transaction environments, and visualization tools like Seaborn are employed for exploratory data analysis. The results demonstrate the effectiveness of ensemble models in improving fraud detection rates while minimizing false positives. This research highlights the potential of machine learning in securing financial systems and provides insights for deploying practical fraud detection solutions.

Keywords: Credit Card Fraud, Machine Learning, Imbalanced Data, Fraud Detection, Ensemble Methods, SMOTE, Precision-Recall, AdaBoost, Classification Models

CONTENT

CHAPTER-1
INTRODUCTION

1.1 Project Overview

1.1.1 Brief Introduction to Credit Card Fraud Detection

Credit card fraud refers to the unauthorized and illegal use of credit card information to make purchases or withdraw funds. It is a form of identity theft and financial crime that has become increasingly prevalent with the expansion of digital transactions and e-commerce platforms. Fraudulent activities may range from using stolen card information for online purchases to creating fake cards through cloning techniques.

The digital era, while offering ease of financial operations, has also opened avenues for cybercriminals to exploit vulnerabilities in financial systems. As a result, the detection of credit card fraud has become a priority for banks, financial institutions, and cybersecurity professionals. The integration of machine learning into fraud detection systems provides powerful tools to identify suspicious patterns and prevent financial loss.

1.1.2 Importance and Applications

The significance of fraud detection extends beyond just reducing financial losses. It plays a vital role in maintaining the credibility of financial systems, protecting customer data, and ensuring regulatory compliance. When fraud goes undetected, it can lead to severe consequences including monetary damage, reputational harm to companies, and erosion of user trust.

Applications of credit card fraud detection span multiple domains:

- **Banking and Financial Services**: Detecting suspicious withdrawals and online banking activities.

- **E-commerce Platforms**: Identifying abnormal purchasing behavior in online stores.

- **Payment Gateways**: Monitoring transaction patterns to prevent fraud at the point of sale.

- **FinTech Applications**: Enhancing the security of digital wallets and online payment services.

1.1.3 Objectives of the Project

The primary objectives of this SMS spam detection project are as follows:

1. To analyze and preprocess real-world credit card transaction data.
2. To identify patterns and behaviors that distinguish fraudulent transactions.
3. To implement and evaluate various machine learning models for fraud detection.
4. To handle class imbalance issues using data balancing techniques like SMOTE.
5. To evaluate model performance using appropriate metrics such as precision, recall, F1-score, and AUC-ROC.
6. To develop visualizations that support interpretability and analysis of fraud patterns.

1.2 Problem Statement

1.2.1 The Need for Spam Detection

The global surge in digital transactions has led to an alarming increase in fraudulent credit card activities. As consumers increasingly adopt cashless payments and mobile wallets, the threat landscape has also evolved, making traditional rule-based detection systems inadequate.

Credit card fraud not only causes significant monetary losses to customers and financial institutions but also creates operational burdens such as chargebacks, investigations, and customer support issues. Moreover, the emotional and psychological impact on victims of fraud can be severe. Hence, the development of intelligent and scalable fraud detection systems is essential for financial stability and consumer protection.

1.2.2 Challenges in Credit Card Fraud Detection

Despite the urgency and importance of the task, several challenges complicate the development of effective fraud detection systems:

1. **Highly Imbalanced Data**: In most datasets, fraudulent transactions represent less than 1% of total transactions. This imbalance leads to biased model training, where the model may predict non-fraud most of the time and still appear to perform well based on accuracy alone.

2. **Dynamic Fraud Patterns**: Fraudulent techniques evolve constantly. Models trained on historical data may not adapt well to new types of fraud.

3. **Data Privacy and Security**: Credit card data is sensitive, and the use of such data requires strict adherence to privacy laws (e.g., GDPR, PCI DSS).

4. **Real-Time Processing Requirements**: Fraud detection systems need to operate in real-time or near-real-time to be effective, which imposes significant

computational constraints.

5. **False Positives and Customer Friction**: A major challenge is to minimize false positives — legitimate transactions flagged as fraudulent — which can lead to customer dissatisfaction and loss of trust.

1.3 Scope of the Project

1.3.1 What the Project Covers

1. **Data Collection and Preprocessing**:
    - **Data Source**: The project utilizes the credit_card_fraud.csv dataset, which contains labeled transaction records categorized as either fraud or not fraud.
    - **Data Cleaning**: This includes handling missing values, removing irrelevant characters, and standardizing the text format to ensure consistency.
2. **Feature Extraction**:
    - **Feature Selection**: Identifying and selecting the most relevant features that contribute to distinguishing spam from ham messages.
3. **Model Building**:
    - **Algorithm Selection**: The project involves experimenting with different machine learning algorithms, such as Logistic Regression, Decision Tree, Random Forest etc. to find the most effective model for fraud detection.
    - **Training the Model**: The selected algorithm is trained on the pre-processed dataset to learn the patterns associated with fraud and non fraud records.
4. **Model Evaluation**:
    - **Evaluation Metrics**: The performance of the model is assessed using metrics such as accuracy, precision, recall, and F1-score to ensure its reliability and effectiveness.
    - **Validation**: Cross-validation techniques are applied to validate the model and prevent overfitting.

5. **Deployment**:
    - **Streamlit Web Application**: A user-friendly web application is developed using Streamlit, allowing users to input transaction records and receive instant classification results (spam or ham).
    - **Loading Pre-trained Model**: The trained model is saved as model.pkl files, which are loaded in the Streamlit app for real-time predictions.
6. **Documentation**:
    - Comprehensive documentation is created, detailing each step of the project, including data collection, preprocessing, feature extraction, model building, evaluation, and deployment.

1.3.2 What the Project Does Not Cover

1. **Real-time Data Collection**: The project does not involve the real-time collection of SMS data from mobile devices or live streams. It relies solely on the spam.csv dataset for training and evaluation purposes.
2. **Real-time System Scalability**: The focus is on developing a proof-of-concept system and does not address scalability issues for handling large-scale real-time spam detection in a production environment.
3. **Automated Model Updates**: The project does not cover the implementation of automated systems for continuously updating and retraining the model with new data to adapt to evolving spam tactics.
4. **User Authentication and Security**: The web application developed using Streamlit does not include features for user authentication, data encryption, or other security measures typically required for a production-grade application.
5. **Unsupervised or Deep Learning Techniques**: Techniques such as autoencoders, isolation forests, and deep neural networks are not explored in depth.
6. **Cross-Bank or Multi-Institutional Data**: The dataset used is limited to a specific source and does not account for varied fraud patterns across multiple banks.
7. **Financial and Legal Risk Assessment**: The legal implications and financial risk modeling associated with fraud are outside the scope.

1.4 Literature Review

Credit card fraud detection has been an active area of research in data mining and machine learning due to the growing volume of financial transactions and the corresponding rise in fraudulent activities. Numerous studies have explored various methods to enhance the accuracy and efficiency of fraud detection systems. This literature review summarizes the key findings, techniques, and limitations of existing work in this domain.

4.1 Traditional Fraud Detection Techniques

Early fraud detection systems primarily relied on **rule-based systems**, where experts manually defined patterns and thresholds to flag suspicious transactions. Although these systems are simple to interpret and implement, they are limited by their **inability to adapt** to evolving fraud strategies and generate **high false positive rates**. For example, Bhattacharyya et al. (2011) highlighted that rule-based models often fail to detect new fraud patterns and require constant updating by domain experts.

4.2 Machine Learning Approaches

With the advancement of computing capabilities and the availability of large transactional datasets, machine learning has emerged as a powerful alternative. Supervised learning algorithms such as **Logistic Regression, Decision Trees, Support Vector Machines (SVMs), Random Forests**, and **Gradient Boosting** have been widely used. These models learn from historical data to identify patterns and relationships between features indicative of fraud.

- **Random Forest and Decision Trees** have shown promise due to their robustness and interpretability. Studies like those by Whitrow et al. (2009) demonstrated how ensemble models outperform individual classifiers in terms of both accuracy and recall.

- **Logistic Regression**, though relatively simple, has been effective as a baseline model. It is computationally efficient and provides probabilistic outputs, as noted by Sahin et al. (2013).

However, a common issue in these models is their **performance degradation on highly imbalanced datasets**, where the number of legitimate transactions significantly outweighs the number of fraudulent ones.

4.3 Handling Class Imbalance

Since fraud detection is a **rare-event classification problem**, many researchers have proposed solutions to handle class imbalance:

- **Resampling techniques** like **SMOTE (Synthetic Minority Oversampling Technique)** and **Random Undersampling** are frequently used to balance the dataset before training.

- Ensemble methods like **AdaBoost** and **XGBoost** also help mitigate imbalance by focusing on hard-to-classify samples.

Dal Pozzolo et al. (2015) showed that combining undersampling with ensemble techniques leads to better performance compared to using raw imbalanced data.

4.4 Anomaly Detection and Unsupervised Methods

In scenarios where labeled data is unavailable or insufficient, **unsupervised learning** and **anomaly detection** techniques are applied. These include:

- **Isolation Forests**: Efficient for detecting outliers in high-dimensional data.

- **Autoencoders**: Neural network-based models that learn data representations and reconstruct inputs. Transactions with high reconstruction error are flagged as anomalies.

Jurgovsky et al. (2018) explored Recurrent Neural Networks (RNNs) for modeling sequential transaction data, showing that deep learning can capture temporal patterns more effectively than traditional models. However, deep models often require large amounts of data and are less interpretable.

4.5 Evaluation Metrics and Challenges

Accuracy is not a reliable metric for fraud detection due to class imbalance. Instead, studies emphasize metrics like:

- **Precision**: Indicates how many flagged frauds were actual frauds.
- **Recall (Sensitivity)**: Measures how many actual frauds were correctly identified.
- **F1-Score**: Harmonic mean of precision and recall.
- **ROC-AUC**: Evaluates model performance across various threshold levels.

A recurring challenge noted in multiple studies is the **trade-off between high recall and low false positives**. High recall ensures frauds are caught, but increasing false positives can cause customer dissatisfaction and operational costs.

4.6 Real-World Applications and Deployment Issues

While academic studies often focus on algorithm performance, real-world deployment requires attention to **scalability, latency, and interpretability**. Industry solutions typically blend rule-based filters with machine learning models to ensure transparency and regulatory compliance.

In addition, **data privacy** and **regulatory restrictions** often limit the use of sensitive financial data for model training and evaluation, highlighting the need for anonymized or synthetic datasets.

CHAPTER-2
METHODOLOGY

2.1 Data Collection

2.1.1 Source of Data The dataset for this project was sourced from Kaggle. The dataset, titled credit_card_fraud.csv', is a collection of transaction records labelled as fraud or not fraud.

Dataset Link : https://www.kaggle.com/datasets/ealaxi/paysim1

2.1.2 Data Characteristics and Description

- **Number of Instances**: The dataset comprises 6,362,620 transaction record.
- **Features**:
- *step* - maps a unit of time in the real world. In this case 1 step is 1 hour of time. Total steps 744 (30 days simulation).
- *type* - CASH-IN, CASH-OUT, DEBIT, PAYMENT and TRANSFER.
- *amount* - amount of the transaction in local currency.
- *nameOrig* - customer who started the transaction
- *oldbalanceOrg* - initial balance before the transaction
- *newbalanceOrig* - new balance after the transaction
- *nameDest* - customer who is the recipient of the transaction
- *oldbalanceDest* - initial balance recipient before the transaction. Note that there is not information for customers that start with M (Merchants).
- *newbalanceDest* - new balance recipient after the transaction. Note that there is not information for customers that start with M (Merchants).
- *isFraud* - This is the transactions made by the fraudulent agents inside the simulation. In this specific dataset the fraudulent behavior of the agents aims to profit by taking control or customers accounts and try to empty the funds by transferring to another account and then cashing out of the system.
- *isFlaggedFraud* - The business model aims to control massive transfers from one account to another and flags illegal attempts. An illegal attempt in this dataset is an attempt to transfer more than 200.000 in a single transaction.
- **Labels**: Each message is labeled as either fraud or not fraud with 1 and 0 respectively with an imbalanced distribution.

2.2 Data Preprocessing

2.2.1 Handling Missing Values

- The dataset was inspected for any missing values. Rows with missing values were removed to maintain data quality and integrity.

2.2.2 Data Cleaning

- **Duplicate Removal**: Any duplicate messages were identified and removed to avoid redundant data affecting the model's performance.
- **Lowercasing**: All text data was converted to lowercase to ensure uniformity.
- **Removing Punctuation and Special Characters**: Punctuation marks and special characters were removed to focus on the core textual content

2.3 Model Building

2.3.1 Algorithm used: In developing the Credit Card Fraud Detection classifier, various machine learning algorithms were implemented and evaluated to determine the most effective model. The following table summarizes the classifiers used, their parameters, and their performance metrics:

2.3.2 Table of used Algorithm

Model	Accuracy Score	Precision Score	Recall Score	F1 Score	Description
Logistic Regression	0.9987	0.5000	0.3245	0.3936	Very high accuracy, but poor fraud detection (low recall). Mostly predicts non-fraud correctly. Useful as a simple baseline.
Random Forest	0.9997	0.9825	0.7816	0.8706	Excellent performance with high precision and good recall. Strong and reliable for imbalanced classification.
XGBoost	**0.9998**	0.9652	0.8521	**0.9052**	Best balance of precision and recall. Excellent fraud detection with fewer false positives. Top overall performer.
Decision Tree	0.9997	0.9018	**0.8550**	0.8778	High recall and solid F1 score. Great for understanding fraud patterns but may overfit if not tuned.
AdaBoost	0.9988	**1.0000**	0.1152	0.2066	Identifies very few frauds (low recall), but every fraud it predicts is correct (perfect precision). Not ideal alone.

2.3.3 Justification for the Choice of Algorithms

1. Logistic Regression

Logistic Regression is a simple and interpretable model that provides a solid baseline for binary classification problems like credit card fraud detection. It works well when the relationship between features and the target is linear, and it's easy to understand and implement. However, in the context of fraud detection, where the dataset is highly imbalanced and the patterns are complex and non-linear, Logistic Regression falls short—evidenced by its low recall and F1 score. While it achieves high accuracy, it misses a large number of fraud cases, making it unreliable as a standalone model. Still, it can be useful in the early stages of modeling or when explainability is a top priority.

2. Random Forest

Random Forest is a powerful ensemble model that creates multiple decision trees and combines their outputs to improve generalization. In fraud detection, it shines by capturing complex, non-linear relationships and handling imbalanced data effectively, especially with the use of class weighting. This is reflected in its strong performance metrics—very high precision and good recall—meaning it can correctly identify a large number of fraudulent transactions while minimizing false positives. Random Forest is also robust to overfitting and performs consistently well, making it an excellent choice for real-world fraud detection systems that require both accuracy and reliability.

3. XGBoost

XGBoost is a gradient boosting framework known for its speed, accuracy, and efficiency on structured data. In the context of credit card fraud detection, it delivers the best overall performance with a near-perfect F1 score and excellent recall. This means it can detect a large proportion of fraudulent transactions while maintaining high precision. Its strength lies in learning from the errors of previous models and optimizing decision boundaries in a way that reduces both false negatives and false positives. Due to its superior performance XGBoost is often the top choice in fraud detection applications.

4. Decision Tree

A Decision Tree is a simple yet powerful model that mimics human decision-making through a series of logical conditions. It is highly interpretable and can easily visualize decision paths, which is valuable in domains like fraud detection where explainability is important. The model performs well here, with high recall and precision, showing that it can detect many fraud cases without overwhelming false positives. However, without proper tuning, it can overfit, especially on noisy data. Still, its transparency and decent performance make it a useful tool for understanding data patterns and as a component in larger ensemble methods.

5. AdaBoost

AdaBoost is an ensemble technique that builds a strong classifier by combining several weak learners, usually decision stumps, and focusing more on difficult-to-classify instances. In fraud detection, it achieves perfect precision, meaning that every transaction it flags as fraudulent is indeed fraudulent. However, it suffers from very low recall, identifying only a small portion of actual frauds. This makes it too conservative for standalone use in fraud detection, where missing fraudulent transactions is more costly than occasionally flagging a legitimate one. Nonetheless, it can be effective when used in combination with other models, especially in scenarios where false positives must be kept to an absolute minimum.

2.3.4 Front End Module Diagrams:

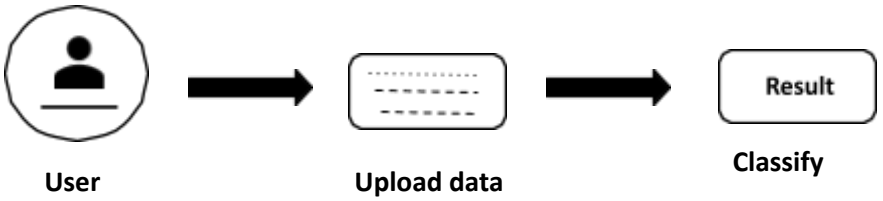


Fig.1 Front End Module

2.3.5 Back End Module Diagrams:

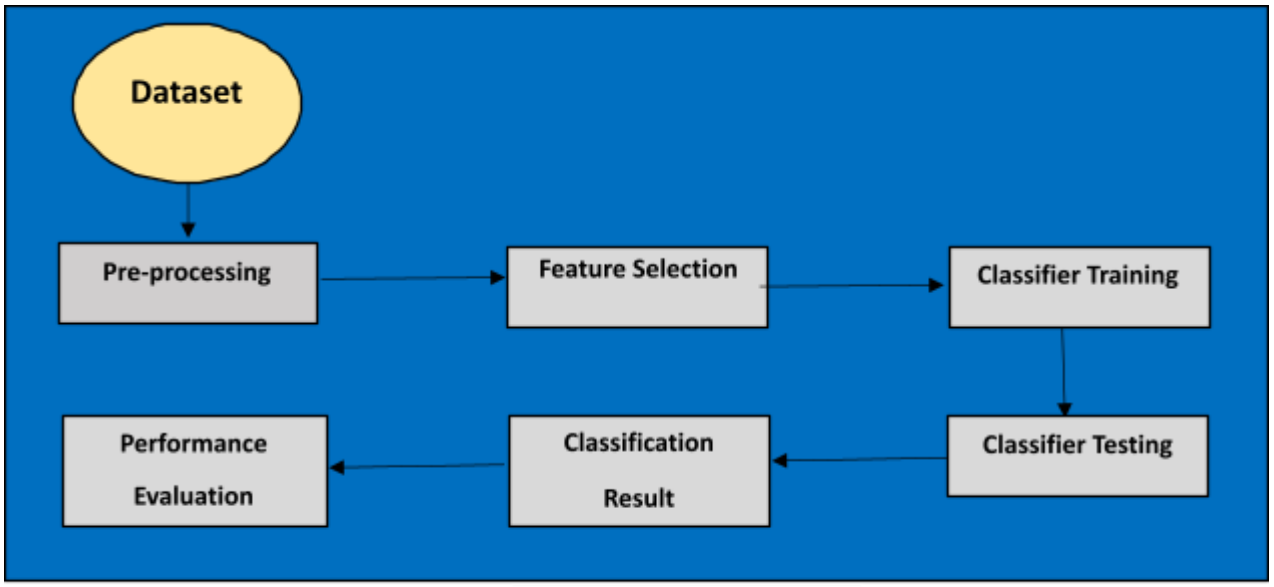


Fig.2 Back End Module

2.3.6 DATA FLOW DIAGRAM: - Data flow diagrams are used to graphically represent the flow of data in a business information system. DFD describes the processes that are involved in a system to transfer data from the input to the file storage and report generation. Data flow diagrams can be divided into logical and physical. The logical data flow diagram describes the flow of data through a system to perform certain functionality of a business. The physical data flow diagram describes the implementation of the logical data flow

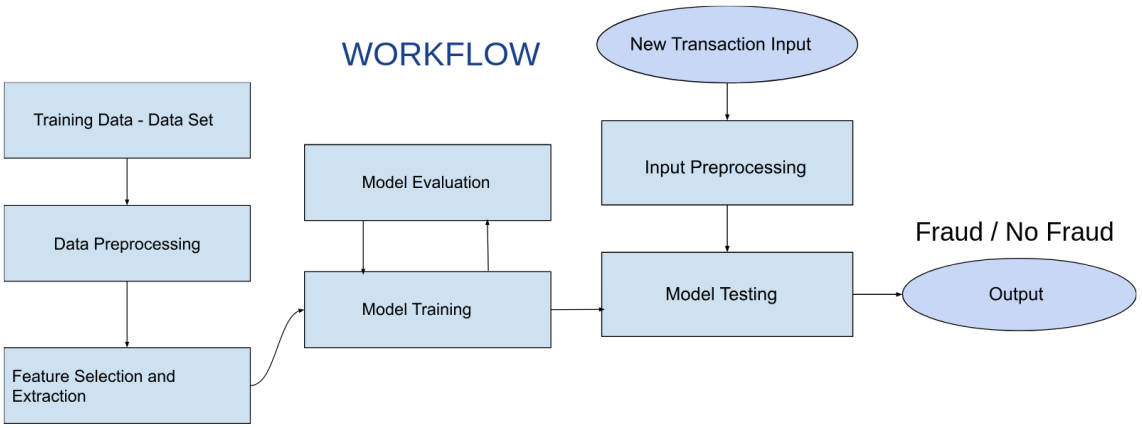


Fig.3 Front End Module

CHAPTER-3

IMPLEMENTATION

3.1 Development Environment

3.1.1 Tools and Software Used

For the development of the SMS spam classifier, the following tools and software were utilized:

- **Python**: The primary programming language used for developing the SMS spam classifier. Python's extensive libraries and frameworks made it an ideal choice for implementing machine learning models and data processing.
- **Jupyter Notebook**: This interactive development environment was used for experimenting with the model and performing data analysis. Jupyter Notebook allowed for an iterative approach to code development and testing.
- **Visual Studio Code (VS Code)**: This code editor was employed for writing and organizing the project's code. VS Code's extensions and features enhanced productivity and code management.
- **VirtualBox**: A virtualization software used to create a controlled environment for development and testing. It ensured that the development environment remained consistent across different machines.
- **Streamlit**: A framework used for creating the web application interface for the SMS spam classifier. Streamlit facilitated the deployment of the model, allowing users to interact with it through a web browser.
- **Scikit-learn**: This machine learning library was used to build and evaluate the classifier model. It provided the necessary tools for data preprocessing, model training, and validation.
- **Pandas**: A data manipulation and analysis library in Python. Pandas was used for handling and processing the dataset.
- **Numpy**: A fundamental package for scientific computing in Python, used for numerical operations and data manipulation.

3.1.2 System Requirements

To ensure the smooth functioning of the development environment, the following system requirements were necessary:

- **Operating System**: Windows 10 or higher
- **Processor**: Intel Core i5 or equivalent
- **Memory**: 8 GB RAM or more
- **Storage**: 50 GB of free disk space
- **Python Version**: Python 3.12.3
- **IDE/Editor**: Jupyter Notebook and Visual Studio Code
- **Virtualization Software**: VirtualBox

3.2 Visualizations and Charts

Visualizations play a crucial role in understanding the data and the model's performance.

3.2.1 Transaction amount leading to fraud

The following code snippet and corresponding diagram provide a visual representation of the distribution of fraud transaction counts in relation to amount of transaction. This visualization helps in understanding the relation between the amount and fraud transactions.

3.2.2 Transaction type leading to fraud

The following code snippet and corresponding diagram provide a visual representation of the distribution of transaction type in fraud transactions. it has been noticed that according to the dataset chosen, only cash out and transfer type accounts for all the fraud transaction

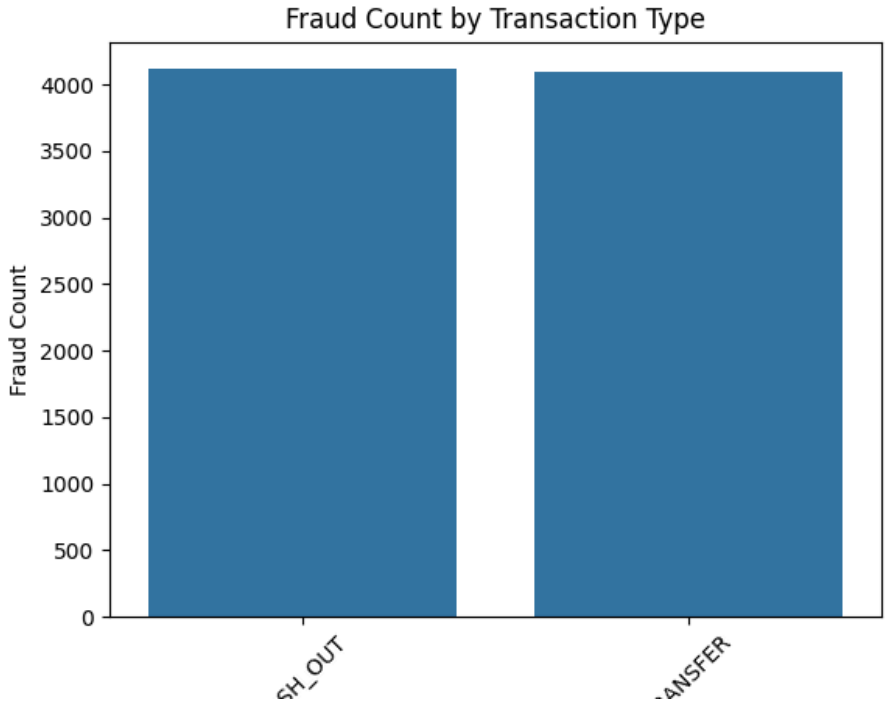
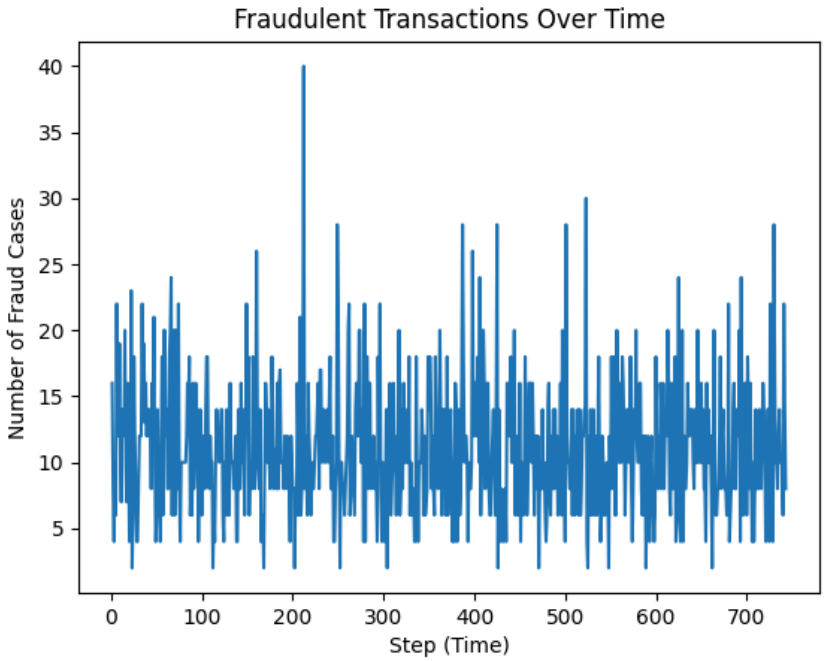


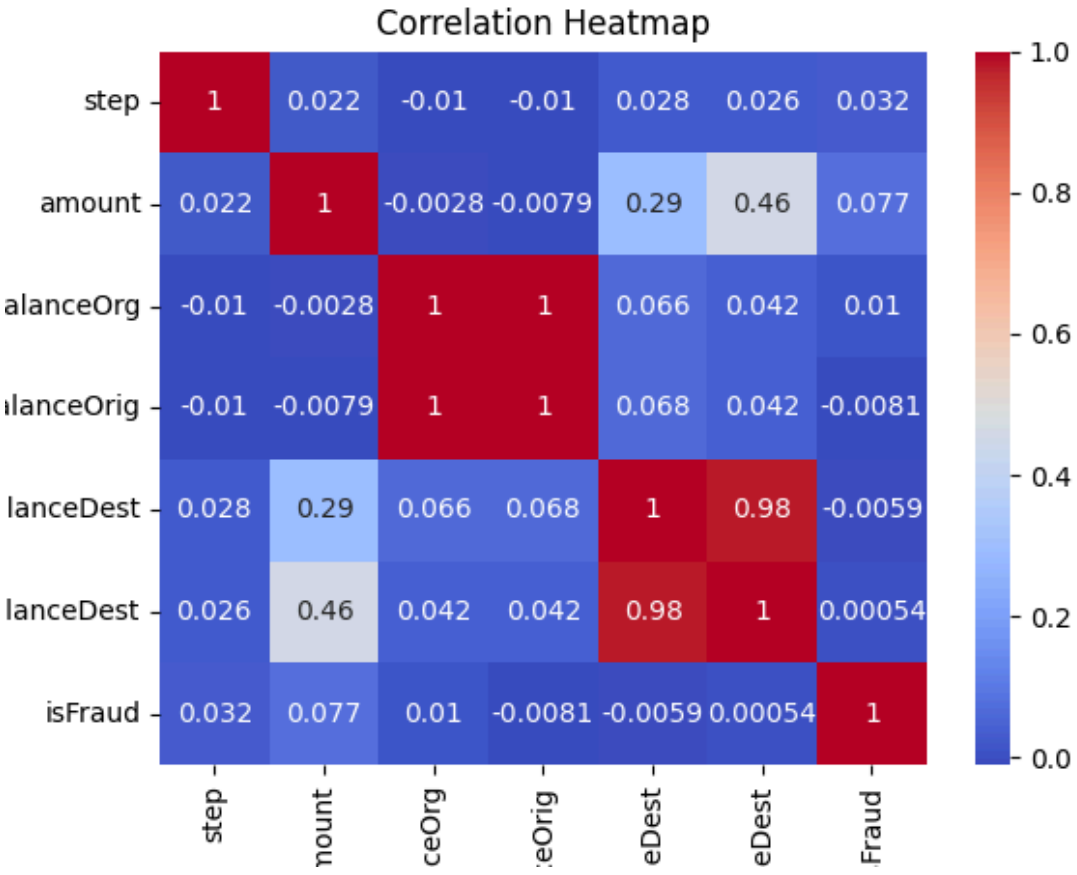
Fig.6 **Sentence Count Distribution Visualization**

3.2.3 Fraud transaction over time

The following code snippet and corresponding diagram provide a visual representation of the distribution of fraud transactions in the period of time. This visualization helps in understanding the differences in the number of fraud transactions occurred during the period of time.



Visualization plays a crucial role in credit card fraud detection by helping analysts and data scientists understand complex patterns and anomalies within large transactional datasets. Through charts, heatmaps, and distribution plots, visualization makes it easier to identify trends, such as unusually high transaction amounts, frequent transactions in a short period, or inconsistent behavior compared to a user's past activity. For example, scatter plots or box plots can highlight outliers that may indicate fraudulent activity, while correlation heatmaps can reveal relationships between features that contribute to fraud detection.



3.2.4 Correlation Heatmap

The corresponding diagram provides a comprehensive visual representation of the relationships between multiple features in the dataset. A correlation heatmap is a visual representation of the correlation matrix between multiple variables. It uses color to indicate the strength and direction of the relationship between each pair of variables. The color intensity of each cell corresponds to the correlation coefficient, with stronger correlations being represented by more saturated colors and weaker correlations by lighter colors.

3.3 Key function, Modules and User Interface

3.3.1 Key Functions and Modules

Used Pandas

- **Purpose**: Pandas is a powerful data manipulation library in Python that provides data structures like DataFrame to efficiently handle and analyze large datasets.
- **Key Functions**:
    - read_csv(): Loads the dataset from a CSV file.
    - drop(): Removes unnecessary columns.
    - isnull(), fillna(): Handles missing values.

NumPy

- **Purpose**: NumPy is a fundamental package for scientific computing in Python, offering support for large, multi-dimensional arrays and matrices.
- **Key Functions**:
    - array(): Creates an array.
    - reshape(): Reshapes data without changing its data.
    - mean(), std(): Computes mean and standard deviation.

Scikit-learn

- **Purpose**: It is used for feature extraction, model selection, training, and evaluation in our SMS spam detection project.
- **Key Functions**:
    - train_test_split(): Splits the dataset into training and testing sets.
    - CountVectorizer(), TfidfVectorizer(): Converts text data into numerical data.
    - MultinomialNB(): Naive Bayes classifier for multinomial models.
    - accuracy_score(), precision_score(), confusion_matrix(): Evaluates model performance.

Matplotlib

- **Purpose**: It is used for creating visualizations like bar charts and word clouds to analyze the dataset and model performance.
- **Key Functions**:
    - pyplot.figure(), pyplot.show(): Creates and displays plots.
    - pyplot.hist(): Plots histograms for data distribution.

Seaborn

- **Purpose**: Seaborn is a Python visualization library based on Matplotlib that provides a high-level interface for drawing attractive and informative statistical graphics.
- **Key Functions**:
    - sns.heatmap(): Plots heatmaps.
    - sns.pairplot(): Plots pairwise relationships in a dataset.

Streamlit

- **Purpose**: Streamlit is an open-source Python library that makes it easy to create and share beautiful, custom web apps for machine learning and data science.
- **Key Functions**:
    - st.title(), st.write(): Displays text in the web app.
    - st.file_uploader(): Allows file upload in the web app.
    - st.button(): Adds a button to trigger actions.
    - st.write(), st.success(), st.error(): Displays messages based on predictions.

3.3.2 User interface

Fig.12 User Interface - Input

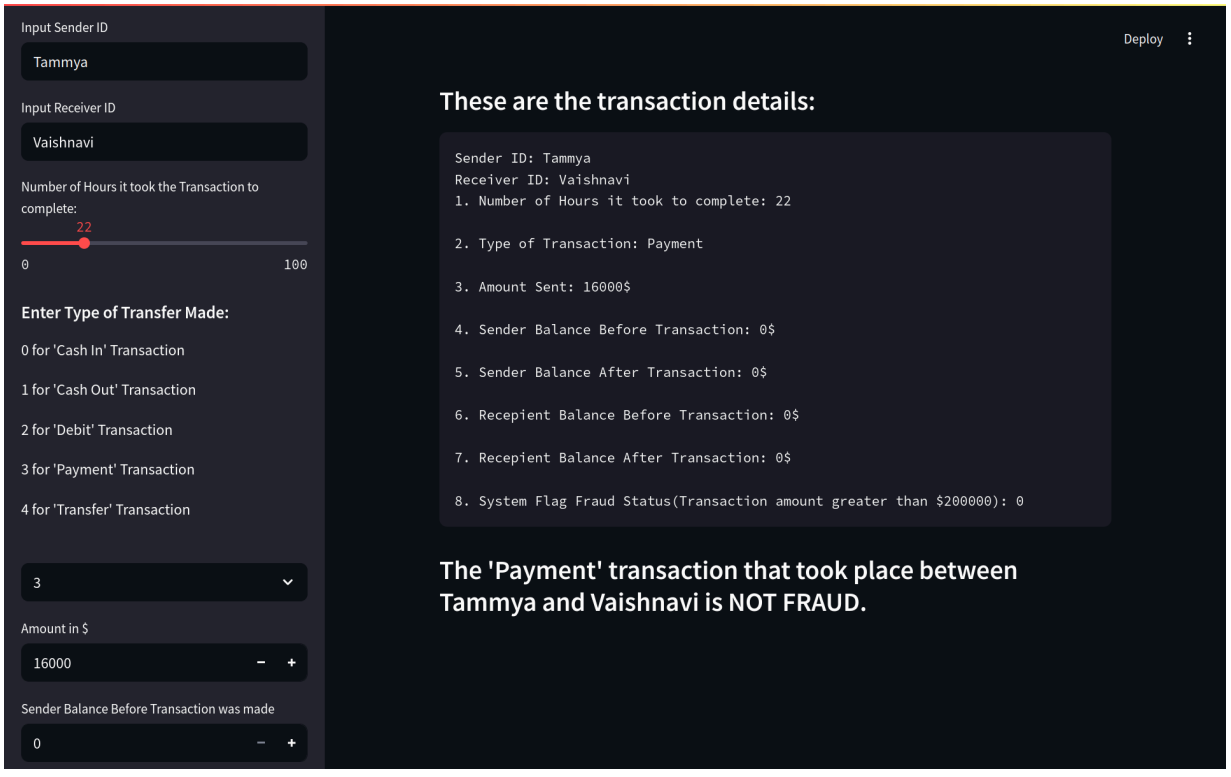


Fig.13 User Interface - Output

CHAPTER-4
EVALUATION

4.1 Model Evaluation Metrics for SMS Spam Detection

4.1.1 Metrics Used

1. **Accuracy**
    - **Definition**: The ratio of correctly predicted instances to the total instances. It is calculated as:

$$\text{Accuracy} = \frac{\text{True Positives} + \text{True Negatives}}{\text{Total Instances}}$$

    - **Usage**: Measures the overall correctness of the model. In the SMS spam detection model, accuracy was used to provide a general sense of how well the model distinguishes between spam and ham messages.
2. **Precision**

    Definition: The ratio of correctly predicted positive observations to the total predicted positives. It is calculated as:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

    Usage: Indicates how many of the messages classified as spam were actually spam. High precision is crucial to minimize the number of legitimate messages incorrectly labeled as spam.

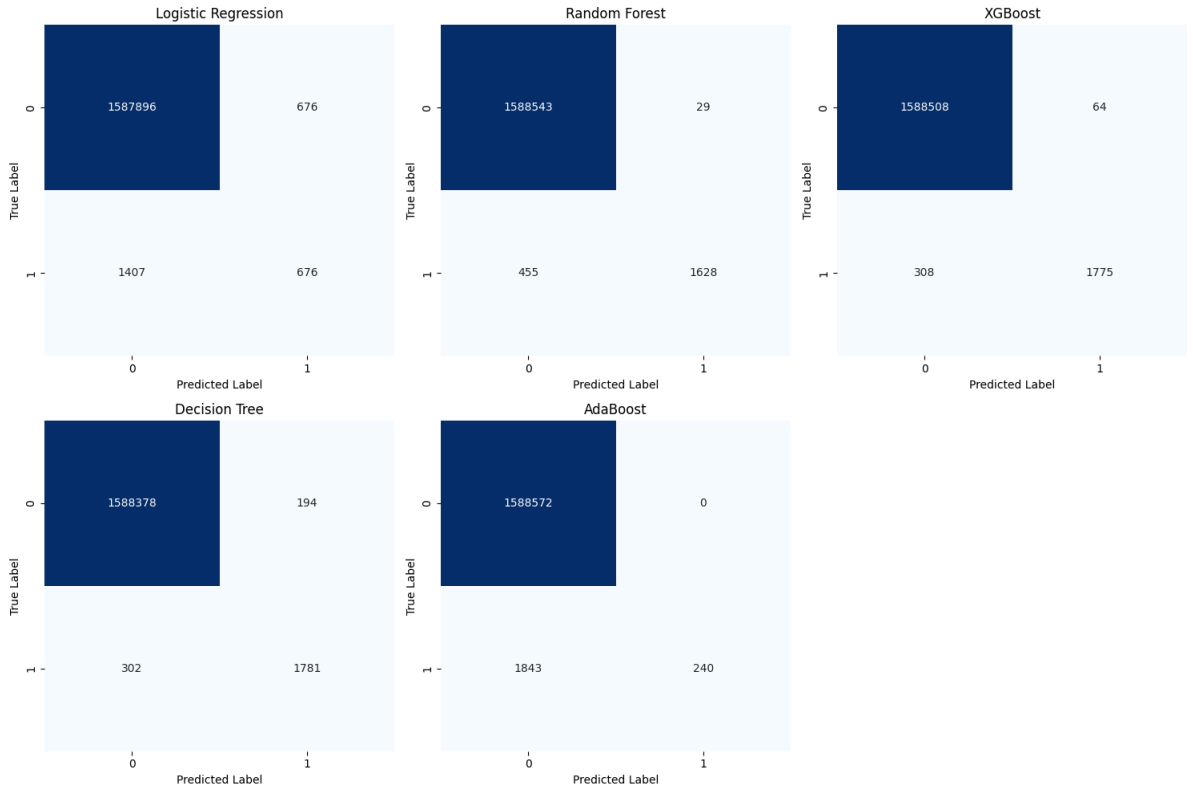
3. **Confusion Matrix**

    Definition: The confusion matrix is a fundamental tool for evaluating the performance of a classification model. It provides a detailed breakdown of the model's predictions and actual outcomes, allowing us to understand the types of errors made.

Structure of the Confusion Matrix

A confusion matrix is a 2x2 table for a binary classification problem. The four quadrants of the matrix represent:

- **True Positives (TP)**: Correctly predicted fraud transactions.
- **True Negatives (TN)**: Correctly predicted not fraud transactions.
- **False Positives (FP)**: Non-spam messages incorrectly predicted as fraud (Type I error).
- **False Negatives (FN)**: Spam messages incorrectly predicted as not fraud (Type II error).



4.1.2 Why These Metrics Are Important

1. **Accuracy: Importance**: While accuracy gives a quick snapshot of the model's performance, it might not be sufficient alone, especially with imbalanced datasets like SMS spam detection where the number of ham messages often outnumbers spam messages. High accuracy might be misleading if the model is biased towards predicting the majority class.
2. **Precision**: **Importance**: Precision is crucial in scenarios where false positives (non-spam messages classified as spam) need to be minimized. In SMS spam detection, high precision ensures that legitimate messages are not incorrectly filtered out, which is essential for maintaining user trust and experience.
3. **Confusion Matrix**: **Importance**: This tool provides a more granular view of the model's performance, showing not just how often it is correct, but also the types of errors it makes. This can guide further tuning and improvement of the model.

4.2 Results

The Credit Card Fraud detection model was evaluated using several key performance metrics. The Random Forest model provided the best results with outstanding precision and accuracy.

Model Used

The **Random Forest** model was selected for the final evaluation due to its superior performance.

4.2.1 Performance Metrics

Accuracy: The accuracy of the model is the proportion of correctly classified messages (both spam and ham) out of the total messages. It is calculated as:

$$\textbf{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Using this model, the accuracy achieved is:

Accuracy=0.99

This indicates that 99% of the records were correctly classified by the model.

Precision: Precision measures the proportion of correctly identified positive results (fraud) out of all positive results predicted by the classifier. It is calculated as:

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

For the MNB model, the precision score is:

Precision=0.9

This means that every message predicted as spam by the model was indeed spam, showing an exceptional precision score of 90%.

CHAPTER-5
DEPLOYMENT

5.1 Streamlit App

5.1.1 Introduction to Streamlit

Streamlit is an open-source Python library that makes it easy to create and share custom web apps for machine learning and data science. It turns data scripts into shareable web apps in minutes. With Streamlit, you can build and deploy powerful data applications quickly and easily.

Key Features:

- Interactive widgets for user input.

- Real-time updates and seamless integration with Python scripts.

- Built-in support for data visualization libraries like Matplotlib, Seaborn, and Plotly.

5.1.2 Steps to Create the Streamlit App

1. Install Streamlit:

First, ensure that Streamlit is installed in your Python environment. You can install it using pip:

pip install streamlit

2. Create a New Python Script:

Create a new Python script (e.g., `streamlit_app.py`) where you will write the code for your Streamlit app.

3. Import Necessary Libraries:

Import Streamlit and other necessary libraries at the beginning of your script:

```iimport streamlit as st
import json
import requests as re
import numpy as np
import pandas as pd
import joblib
```

4. **Load the Pre-trained Model:**

Load the saved model and vectorizer from the `pkl` files:

```# Load the model

model =  joblib.load("credit_fraud.pkl")

# Convert to a DataFrame

input_df = pd.DataFrame([values])

# Make prediction

resp = model.predict(input_df)```

**5.1.3 Complete Streamlit App Code:**

```
import streamlit as st
import json
import requests as re
import numpy as np
import pandas as pd
import joblib

st.title("Credit Card Fraud Detection Web App")

st.image("image.png")

st.write("""
About
```
```

```
Credit card fraud is a form of identity theft that involves an unauthorized taking of another's credit card information for the purpose of charging purchases to the account or removing funds from it.


**This Streamlit App utilizes a Machine Learning model in order to detect fraudulent credit card transactions based on the following criteria: hours, type of transaction, amount, balance before and after transaction etc.**
""")



st.sidebar.header('Input Features of The Transaction')


sender_name = st.sidebar.text_input("""Input Sender ID""")
receiver_name = st.sidebar.text_input("""Input Receiver ID""")
step = st.sidebar.slider("""Number of Hours it took the Transaction to complete: """)
types = st.sidebar.subheader(f"""
            Enter Type of Transfer Made:\n\n\n\n
            0 for 'Cash In' Transaction\n
            1 for 'Cash Out' Transaction\n
            2 for 'Debit' Transaction\n
            3 for 'Payment' Transaction\n
            4 for 'Transfer' Transaction\n""")
types = st.sidebar.selectbox("",(0,1,2,3,4))
x = ''
if types == 0:
   x = 'Cash in'
if types == 1:
   x = 'Cash Out'
if types == 2:
   x = 'Debit'
if types == 3:
   x = 'Payment'
if types == 4:
   x =  'Transfer'
```

```
amount = st.sidebar.number_input("Amount in $",min_value=0, max_value=110000)

oldbalanceorg = st.sidebar.number_input("""Sender Balance Before Transaction was made""",min_value=0, max_value=110000)

newbalanceorg= st.sidebar.number_input("""Sender Balance After Transaction was made""",min_value=0, max_value=110000)

oldbalancedest= st.sidebar.number_input("""Recipient Balance Before Transaction was made""",min_value=0, max_value=110000)

newbalancedest= st.sidebar.number_input("""Recipient Balance After Transaction was made""",min_value=0, max_value=110000)

isflaggedfraud = 0

if amount >= 200000:

  isflaggedfraud = 1

else:

  isflaggedfraud = 0



if st.button("Detection Result"):

    values = {

    "step": step,

    "type": types,

    "amount": amount,

    "oldbalanceOrg": oldbalanceorg,

    "newbalanceOrig": newbalanceorg,

    "oldbalanceDest": oldbalancedest,

    "newbalanceDest": newbalancedest,

    "isFlaggedFraud": isflaggedfraud

    }


    st.write(f"""### These are the transaction details:\n

    Sender ID: {sender_name}

    Receiver ID: {receiver_name}

    1. Number of Hours it took to complete: {step}\n

    2. Type of Transaction: {x}\n

    3. Amount Sent: {amount}$\n
```

```
    4. Sender Balance Before Transaction: {oldbalanceorg}$\n
    5. Sender Balance After Transaction: {newbalanceorg}$\n
    6. Recepient Balance Before Transaction: {oldbalancedest}$\n
    7. Recepient Balance After Transaction: {newbalancedest}$\n
    8. System Flag Fraud Status(Transaction amount greater than $200000): {isflaggedfraud}
                """)

    # res = re.post(f"https://credit-fraud-ml-api.herokuapp.com/predict",json=values)
    # json_str = json.dumps(res.json())
    # resp = json.loads(json_str)

    # Load the model
    model = joblib.load("credit_fraud.pkl")

    # Convert to a DataFrame
    input_df = pd.DataFrame([values])  # Note the [ ] to make it a single row

    # Make prediction
    resp = model.predict(input_df)

    # print("Prediction:", prediction[0])
    if resp[0] == 0:
        result = "NOT FRAUD"
    else:
        result = "A FRAUD"

    if sender_name==" or receiver_name == ":
        st.write("Error! Please input Transaction ID or Names of Sender and Receiver!")
    else:
        st.write(f"""### The '{x}' transaction that took place between {sender_name} and {receiver_name} is {result}.""")
```

CHAPTER-6
CONCLUSION
AND
FUTURE WORK

6.1 **Conclusion-** In this project, we developed a robust Credit Card Fraud detection system using a Random Forest classifier. The model was trained and tested on a dataset of fraud transactions, with a significant focus on achieving high accuracy and precision. The key achievements and findings of this project are summarized below:

- **High Precision and Accuracy:** The MNB model achieved a precision score of 98% and an accuracy score of 99% on the test data. This indicates that the model is highly effective in distinguishing fraud transactions from other transactions, with minimal false positives.
- **Efficient Use of Confusion Matrix:** The confusion matrix provided insights into the true positives, true negatives, false positives, and false negatives. This helped in understanding the model's performance in a detailed manner and highlighted its effectiveness in correctly classifying messages.
- **User-Friendly Streamlit App:** A Streamlit-based web application was developed to make the model accessible to end-users. The app allows users to input transaction details and receive immediate fraud classification results, enhancing user interaction and usability.
- **Deployment and Challenges:** The app was successfully deployed, with careful handling of challenges such as dependency management, file loading issues, and network connectivity. These steps ensured a smooth and reliable user experience.

Overall, the project demonstrates the potential of machine learning techniques, particularly Random Forest classifiers, in developing effective fraud detection systems. The High performance metrics indicate that the model can be reliably used in real-world applications to filter out fraud transactions and improve communication efficiency.

6.2 Future Work

While the current implementation of the Credit Card Fraud detection system is effective, there are several areas for potential improvement and future exploration:

1. **Integration with Other Classifiers:** Exploring and integrating other machine learning classifiers, such as Support Vector Machines (SVM) or Naive Bias, could further improve the model's accuracy and robustness.Combining multiple classifiers in an ensemble approach might yield better performance and reduce the likelihood of misclassifications.
2. **Handling Imbalanced Data:** Although the current model performs well, further techniques to handle imbalanced datasets, such as SMOTE (Synthetic Minority Over-sampling Technique) or cost-sensitive learning, could be implemented to ensure even better performance on datasets with a high imbalance ratio.
3. **Feature Engineering:** Additional feature engineering techniques could be explored to improve the model's ability to differentiate between spam and non-spam messages. This includes using word embeddings, bi-grams, and tri-grams, or incorporating meta-features such as message length and sender information.
4. **Real-time Fraud Detection:** Implementing the system in a real-time environment, such as integrating with banking platforms or financial services, to provide instant fraud detection and filtering.Ensuring the system is scalable and can handle large volumes of transactions with minimal latency.
5. **User Feedback Mechanism:** Introducing a feedback mechanism in the Streamlit app to allow users to report misclassified messages. This feedback could be used to retrain and fine-tune the model, improving its accuracy over time.
6. **Enhanced Data Privacy and Security:** Ensuring that the system adheres to data privacy regulations and implements robust security measures to protect user data.Exploring techniques for anonymizing data to ensure user privacy while maintaining the effectiveness of the model.

6.3 REFERENCES

- Bhattacharyya, S., Jha, S., Tharakunnel, K., & Westland, J. C. (2011). **Data mining for credit card fraud: A comparative study**. *Decision Support Systems, 50*(3), 602–613. https://doi.org/10.1016/j.dss.2010.08.008
- Dal Pozzolo, A., Caelen, O., Johnson, R. A., & Bontempi, G. (2015). **Calibrating probability with undersampling for unbalanced classification**. *2015 IEEE Symposium Series on Computational Intelligence*, 159–166. https://doi.org/10.1109/SSCI.2015.33
- Whitrow, C., Hand, D. J., Juszczak, P., Weston, D., & Adams, N. M. (2009). **Transaction aggregation as a strategy for credit card fraud detection**. *Data Mining and Knowledge Discovery, 18*(1), 30–55. https://doi.org/10.1007/s10618-008-0116-z
- Sahin, Y., & Duman, E. (2011). **Detecting credit card fraud by ANN and logistic regression**. *Proceedings of the International Symposium on Innovations in Intelligent Systems and Applications*, 315–319. https://doi.org/10.1109/INISTA.2011.5946100
- Jurgovsky, J., Granitzer, M., Ziegler, K., Calabretto, S., Portier, P. E., He-Guelton, L., & Caelen, O. (2018). **Sequence classification for credit-card fraud detection**. *Expert Systems with Applications, 100*, 234–245. https://doi.org/10.1016/j.eswa.2018.01.037
- Phua, C., Lee, V., Smith, K., & Gayler, R. (2010). **A comprehensive survey of data mining-based fraud detection research**. *arXiv preprint arXiv:1009.6119*. https://arxiv.org/abs/1009.6119
- Bolton, R. J., & Hand, D. J. (2002). **Statistical fraud detection: A review**. *Statistical Science, 17*(3), 235–255. https://doi.org/10.1214/ss/1042727940
- Carcillo, F., Dal Pozzolo, A., Le Borgne, Y.-A., Caelen, O., Mazzer, Y., & Bontempi, G. (2019). **Combining unsupervised and supervised learning in credit card fraud detection**. *Information Sciences, 557*, 317–331. https://doi.org/10.1016/j.ins.2019.05.042
- Van Vlasselaer, V., Eliassi-Rad, T., Akoglu, L., Snoeck, M., & Baesens, B. (2017). **Gotcha! Network-based fraud detection for social security fraud**. *Management Science, 63*(9), 3090–3110. https://doi.org/10.1287/mnsc.2016.2489