A PROJECT REPORT

on

# “LEVERAGING SMOTE AND RANDOM FOREST FOR IMPROVED CREDIT CARD FRAUD DETECTION”

**Submitted**

by

221FA04073

B Chaitanya Nandini

221FA04076

Irfan Aziz

221FA04121

Maddala Rakshita

221FA04122

Maddala Ruchita

**Under the guidance of**

Maridu Bhargavi

*Assistant professor, CSE*

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**VIGNAN'S FOUNDATION FOR SCIENCE, TECHNOLOGY AND RESEARCH**

**Vadlamudi, Guntur.**

**ANDHRA PRADESH, INDIA, PIN-522213.**

VIGNAN'S
FOUNDATION FOR SCIENCE, TECHNOLOGY & RESEARCH
(Deemed to be University) - Estd. u/s 3 of UGC Act 1956

## CERTIFICATE

This is to certify that the Project entitled **"**Leveraging Smote And Random Forest For Improved Credit Card Fraud Detection**"** that is being submitted by 221FA04073 (Chaitanya Nandini), 221FA04076 (Irfan Aziz), 221FA04121(Maddala Rakshita), 221FA04122(Maddala Ruchita) for partial fulfilment of Project is a bonafide work carried out under the supervision of Ms. Maridu Bhargavi, M.Tech., Assistant Professor, Department of CSE.

| | | |
|---|---|---|
| | [illegible] | [illegible] |
| M. Bhargavi | Dr. S. V. Phani Kumar | Dr.K.V. Krishna Kishore |
| Assistant Professor, CSE | HOD,CSE | Dean, SoCI |

VIGNAN'S
FOUNDATION FOR SCIENCE, TECHNOLOGY & RESEARCH
(Deemed to be University) - Estd. u/s 3 of UGC Act 1956

# DECLARATION

We hereby declare that the Project entitled **"Leveraging Smote And Random Forest For Improved Credit Card Fraud Detection"** is being submitted by 221FA04073 (B Chaitanya Nandini), 221FA04076 (Irfan Aziz), 221FA04121(Maddala Rakshita), 221FA04122(Maddala Ruchita) in partial fulfilment of  Project course work. This is our original work, and this project has not formed the basis for the award of any degree. We have worked under the supervision of Ms. Maridu Bhargavi, M.Tech., Assistant Professor, Department of CSE.

By
**221FA04073 (B Chaitanya Nandini),**
**221FA04076 (Irfan Aziz),**
**221FA04121(Maddala Rakshita),**
**221FA04122(Maddala Ruchita)**

Date:

# ABSTRACT

The accelerating speed of digital transactions has made credit card theft a serious concern to customers and banks alike. Traditional means of fraud detection are no longer as strong as today's more sophisticated methods. This paper presents a summary of various machine learning techniques applied in credit card fraud detection. Several algorithms are highlighted, including logistic regression, random forests, and boosting methods such as gradient boosting and LightGBM. These algorithms were selected due to their capacity to manage big datasets and identify patterns unique to fraud.

According to the results, the best performance was obtained by random forests, which achieved 99.30\% accuracy, outperforming all other methods. Gradient boosting and logistic regression were also competitive, reaching 98.5\% accuracy. This study offers recommendations for improving fraud detection techniques while showcasing the efficacy of machine learning in this regard.

# TABLE OF CONTENTS

## LIST OF FIGURES

## LIST OF TABLES

# CHAPTER-1

# INTRODUCTION

# 1. INTRODUCTION

Credit card fraud has emerged as one of the most significant challenges in today's digital economy, driven by the explosion of online and card-not-present (CNP) transactions. As businesses and consumers increasingly shift toward cashless payments, fraudsters have found sophisticated ways to exploit security gaps in transaction systems. The increasing volume and complexity of fraudulent activities have created an urgent demand for advanced fraud detection techniques that can identify and prevent fraud before it causes widespread damage. Machine learning (ML) algorithms, particularly those leveraging techniques like SMOTE (Synthetic Minority Over-sampling Technique) and Random Forest, offer promising solutions for improving fraud detection by addressing imbalanced datasets and enhancing classification accuracy.

## 1.1 What is credit card fraud and its causes

Credit card fraud is the unauthorized use of credit card information for financial gain, typically through illegal transactions. It takes many forms, from physical card theft to more sophisticated digital schemes such as identity theft, phishing, skimming, and data breaches. The evolution of digital technologies has made it easier for criminals to access sensitive information and carry out fraudulent transactions.

- **Phishing**: One common technique involves phishing, where fraudsters use fake emails, text messages, or websites to trick cardholders into revealing their personal or financial information. For example, an individual might receive an email that appears to be from their bank, asking them to confirm their account details.
- **Card Skimming**: Another method is card skimming, where criminals install small devices on ATMs or point-of-sale terminals to capture credit card data. This information is later used to make unauthorized purchases or cloned cards.

- **Data Breaches**: Data breaches at major retailers or financial institutions also provide criminals with access to large volumes of credit card data. High-profile breaches, such as the Target or Equifax breaches, have exposed millions of cardholders to potential fraud.

Each of these methods allows fraudsters to steal cardholder information, which is then used to make purchases, transfer funds, or engage in other financial transactions without the cardholder’s consent. As technology advances, fraud tactics become more complex and harder to detect through traditional methods. For instance, fraudsters might attempt smaller, less noticeable transactions to test stolen cards before making larger purchases. In other cases, they might use advanced techniques like synthetic identity fraud, where they combine real and fake information to create new identities and bypass security checks.

## 1.2 The challenges of fraud detection

Detecting credit card fraud presents several key challenges, primarily due to the nature of fraud and the structure of transaction data. Fraud detection systems must be able to differentiate between legitimate and fraudulent transactions in real-time, often with limited information available about each transaction. However, fraud detection is made more difficult by several factors:

1. **Class Imbalance**: One of the most significant challenges is the highly imbalanced nature of fraud detection datasets. Fraudulent transactions typically make up less than 1% of all transactions, which means that most machine learning algorithms tend to favor the majority class (legitimate transactions), leading to a high rate of false negatives (i.e., failing to detect actual fraud). This imbalance skews the model's learning process, as it is much easier for the model to classify most transactions as non-fraudulent, thus minimizing the occurrence of correct fraud detection.
2. **Dynamic Nature of Fraud**: Fraud schemes constantly evolve, making it difficult for static rule-based systems to keep up. Fraudsters often develop new strategies to bypass detection mechanisms. For instance, they might exploit vulnerabilities in online shopping systems or take advantage of weak encryption in digital wallets. Traditional fraud detection methods, which rely on pre-defined rules such as transaction limits or geographical constraints, are not flexible enough to adapt to these changing fraud tactics. As a result, they can either miss new forms of fraud or incorrectly classify legitimate transactions as fraudulent.

3. **Real-time Detection**: Fraud detection systems must operate in real-time or near-real-time to prevent unauthorized transactions. This requirement places a significant computational burden on the detection system, as it must quickly analyze each transaction's features—such as transaction amount, location, time, and merchant information—and make a decision within milliseconds. Any delay could result in the fraudulent transaction being completed, leading to financial losses for cardholders and issuers.
4. **Feature Complexity**: Transactions have numerous features that influence whether they are flagged as fraudulent, including geographic location, transaction history, spending patterns, and merchant details. This high dimensionality makes it challenging for algorithms to accurately classify transactions without overfitting or missing subtle patterns. Furthermore, features that are significant in one type of fraud might not be useful in others, requiring systems to be adaptable.
5. **False Positives**: While catching fraudulent transactions is critical, systems must also avoid too many false positives (i.e., legitimate transactions flagged as fraudulent). A high false positive rate frustrates customers and can damage the reputation of financial institutions. Customers whose transactions are frequently declined due to inaccurate fraud detection systems may lose trust in their bank or card issuer, resulting in customer churn.

## 1.3 The economic and operational impacts of fraud

The financial and operational impact of credit card fraud is substantial for both individuals and institutions. For financial institutions, the costs of fraud extend beyond the direct financial losses associated with unauthorized transactions. Some of the key impacts include:

1. **Direct Financial Losses**: Banks and credit card issuers are often liable for fraudulent charges, especially when the fraud occurs as a result of system vulnerabilities or failure to detect fraud in time. These financial losses can amount to millions or even billions of dollars annually. According to a report from the Nilson Report, global credit card fraud losses reached $28.65 billion in 2019 and are expected to continue rising.
2. **Reimbursement and Dispute Handling**: Fraudulent transactions often lead to disputes between customers and financial institutions. Handling these disputes requires significant time and resources. Banks must investigate each case, determine whether fraud occurred,

and, in many cases, reimburse the customer for the lost funds. This not only incurs direct costs but also increases the operational workload.

3. **Reputational Damage**: The reputational impact of fraud is significant. Customers expect their banks and payment providers to protect them from fraud. When institutions fail to detect and prevent fraud, they risk losing customer trust and loyalty. A single high-profile breach or failure to prevent fraud can result in negative media attention, leading to reputational damage that is difficult to recover from.
4. **Increased Operational Costs**: To combat fraud, financial institutions must invest heavily in technology, staff, and infrastructure. Fraud detection systems must be updated regularly to respond to new fraud patterns, and staff must be trained to handle fraud-related issues. Institutions also need to implement more stringent security measures, such as multi-factor authentication, encryption, and fraud monitoring tools, which add to operational costs.
5. **Customer Experience**: Overly aggressive fraud detection systems that block legitimate transactions can harm the customer experience. If a customer's card is declined due to a false positive, they may experience inconvenience or embarrassment. This can lead to frustration, and in some cases, customers may switch to another bank or payment provider.

## 1.4 Traditional methods of fraud detection

Traditional fraud detection systems are often based on rule-based algorithms, which rely on predefined rules to identify fraudulent transactions. These rules are typically created based on historical fraud patterns and include conditions such as transaction limits, geographic restrictions, and unusual spending patterns.

1. **Rule-Based Systems**: Rule-based systems are relatively simple to implement and understand. For example, a rule might flag any transaction over $5,000 as potentially fraudulent, or it might block transactions made from a foreign country if the cardholder's previous transactions were domestic. However, rule-based systems have several limitations. Since they are static, they struggle to keep up with evolving fraud tactics. Fraudsters can easily circumvent these systems by staying within the predefined thresholds or mimicking legitimate spending patterns.

2. **Limitations of Rule-Based Systems**: Rule-based systems are prone to both false positives and false negatives. As fraudsters become more sophisticated, these systems often fail to detect new fraud schemes that do not fit the predefined rules. Additionally, legitimate transactions that appear unusual based on the rules may be flagged incorrectly, resulting in customer dissatisfaction.
3. **Lack of Scalability**: Rule-based systems are also not scalable. As the volume of transactions increases, adding more rules to cover various types of fraud becomes inefficient. Managing and updating a large number of rules becomes complex, and the system can become slow and unresponsive. Furthermore, static rules cannot account for nuanced transaction behaviors, such as changes in a customer's spending habits over time.

## 1.5 Applications of machine learning for fraud detection

Machine learning (ML) offers a dynamic and adaptive approach to fraud detection that can overcome many of the limitations of traditional rule-based systems. Unlike static rules, machine learning models can learn from data and continuously improve over time. They are capable of identifying complex patterns in transaction data that are indicative of fraud, even when those patterns are subtle or evolve over time.

1. **Supervised Learning Algorithms**: Many machine learning approaches to fraud detection use supervised learning algorithms, where the model is trained on a labeled dataset of historical transactions (fraudulent and legitimate). Popular algorithms include **Random Forest**, **Logistic Regression**, and **Support Vector Machines** (SVMs). These models learn the relationships between transaction features (such as amount, location, and time) and the target label (fraud or non-fraud), allowing them to predict whether a new transaction is likely to be fraudulent.
2. **Ensemble Methods**: Ensemble methods, such as **Random Forest** and **Gradient Boosting**, have proven to be particularly effective in fraud detection. These models combine the predictions of multiple decision trees to improve classification accuracy. For instance, Random Forest builds multiple decision trees and aggregates their predictions to reduce variance and avoid overfitting. This makes it robust against noisy or incomplete data, which is common in real-world fraud detection.

3. **SMOTE for Handling Imbalanced Data**: One of the major challenges in applying machine learning to fraud detection is the class imbalance problem. To address this, techniques like SMOTE (Synthetic Minority Over-sampling Technique) are employed. SMOTE generates synthetic samples for the minority class (fraudulent transactions) by interpolating between existing minority instances. This helps balance the dataset, allowing the model to better learn the characteristics of fraud and improve its detection capabilities.
4. **Real-time Fraud Detection**: Machine learning models are increasingly being used for real-time fraud detection, where transactions are evaluated and classified within milliseconds. By analyzing a wide array of features—such as transaction history, user behavior, and geolocation—ML models can quickly flag suspicious transactions for further review or immediate blocking. The ability to operate in real-time is essential for preventing fraud before it causes financial losses.

# CHAPTER-2

# LITERATURE SURVEY

# 2. LITERATURE SURVEY

## 2.1 Review of previous research

1) Alok Kumar et al. (2024) conducted a study on credit card fraud detection, exploring the application of machine learning techniques such as Naive Bayes, Logistic Regression, and Random Forest. Their work focused on detecting fraudulent transactions in highly imbalanced datasets, with only a small fraction of transactions being fraudulent. The authors emphasized the effectiveness of these models in analyzing large volumes of transaction data to uncover patterns indicative of fraud. Naive Bayes emerged as the top performer, achieving 99.3% accuracy, followed by Random Forest and Logistic Regression with accuracy rates of 98.5%.

The study demonstrated that while Naive Bayes excelled in this particular dataset, combining different algorithms could further enhance the detection process. This is in line with previous findings by Bonkoungou et al. (2023) and Mirhashemi et al. (2023), who also found that ensemble methods and hybrid approaches improved fraud detection accuracy. Kumar et al. concluded that machine learning, especially when combined with advanced techniques like SMOTE for handling imbalanced data, offers a promising solution for real-time fraud detection

2. Ruttala et al. (2020) presented a study on credit card fraud detection using machine learning, focusing on the comparative performance of Random Forest and Adaboost algorithms. The study addresses the growing concern of credit card fraud due to the rise in online transactions and e-commerce. The authors aimed to classify fraudulent and non-fraudulent transactions using machine learning and compared the algorithms based on accuracy, precision, recall, and F1-score.

The Kaggle credit card fraud dataset from 2013 was utilized, consisting of anonymized transaction data. The dataset's imbalanced nature, with only 0.172% fraudulent transactions, posed challenges for accurate detection. To address this, machine learning algorithms were applied to classify fraud with high precision.

**Random Forest Algorithm:** Ruttala et al. highlighted Random Forest as a robust ensemble technique that reduces overfitting by averaging predictions from multiple decision trees. The algorithm demonstrated high accuracy in classifying fraud and non-fraud transactions by identifying patterns in the data using information gain as a criterion for node splitting.

**Adaboost Algorithm:** Adaboost, on the other hand, was noted for its ability to combine weak classifiers into a strong one by focusing more on incorrectly classified instances. However, the authors pointed out its sensitivity to noisy data, which makes it less suitable when there are significant outliers in the dataset.

**Comparison of Algorithms:** The study found that while both algorithms achieved similar accuracy, Random Forest outperformed Adaboost in terms of precision, recall, and F1-score, making it the more effective choice for detecting credit card fraud. Adaboost's sensitivity to noise was identified as a limiting factor, whereas Random Forest was better at handling the imbalanced dataset.

In conclusion, Ruttala et al. demonstrated that although Adaboost can boost weak classifiers, Random Forest provided better overall performance for fraud detection. The authors proposed future work involving deep learning algorithms to further improve detection accuracy

3. Alarfaj et al. (2022) presented a study on credit card fraud detection, utilizing state-of-the-art machine learning (ML) and deep learning (DL) algorithms to tackle the evolving nature of fraud. With the rise in digital payments and e-commerce, credit card fraud has increased significantly, and existing ML approaches, such as Decision Trees, Support Vector Machines (SVM), and Random Forest, have been used but with limitations in accuracy. The study's primary focus was to apply advanced DL architectures, particularly Convolutional Neural Networks (CNN), to enhance detection performance on the European credit card fraud benchmark dataset, addressing issues like high-class imbalance and false positives.

The authors used ML models first, including Random Forest and XGBoost, and later compared their performance with CNN architectures. The DL models, with additional layers for feature extraction and classification, outperformed traditional ML approaches. For instance, CNN achieved 99.9% accuracy, 85.71% F1-score, and 98% AUC, significantly reducing false negatives compared to ML models.

The study concluded that DL approaches like CNN are more effective than traditional ML models in detecting credit card fraud, especially in handling large, imbalanced datasets. The authors proposed further exploration of deep learning techniques to improve fraud detection systems in real-world applications

4) Bhakta et al. (2023) conducted a comparative study on credit card fraud detection using various machine learning algorithms, including traditional methods like Logistic Regression and Support Vector Machines, and ensemble techniques such as Random Forest, XGBoost, and AdaBoost. The study highlighted that ensemble methods, particularly Random Forest and XGBoost, performed better in terms of precision, recall, and F1-score, making them more effective for detecting fraudulent transactions.
The authors demonstrated that while traditional algorithms were easy to implement, they fell short in performance compared to ensemble techniques, which offered better predictive accuracy and reduced overfitting. The study aligns with prior research by Faraji (2021) and Pandey et al. (2021), which also found that ensemble and deep learning methods improve fraud detection. Bhakta et al. concluded that ensemble learning provides a robust approach for real-time fraud detection.

5) Tanouz et al. (2021) conducted a study focused on detecting credit card fraud using various machine learning algorithms such as **Logistic Regression, Random Forest, and Naive Bayes**. Their research aimed to address the challenge posed by highly imbalanced datasets in fraud detection. The dataset, containing transactions from European credit cardholders, was processed using techniques like **under-sampling** and **outlier detection** to improve the performance of these algorithms. The study showed that **Random Forest** achieved the highest accuracy (96.77%) and precision (100%), outperforming other models such as **Logistic Regression** and **Naive Bayes** (Credit_Card_Fraud_Detec…).
Tanouz et al. highlighted the importance of cleaning and preprocessing the dataset to handle bias and improve prediction outcomes. Their findings align with prior studies that emphasize the use of **Random Forest** as a robust method for fraud detection due to its ability to handle complex, high-dimensional data. This research also supported the use of data mining techniques like **outlier detection** to further reduce false positives and negatives, a conclusion similarly drawn by Subramanian et al. (2020) and Bah et al. (2019), who explored outlier detection and removal strategies for improving accuracy in imbalanced datasets
Despite their success, Tanouz et al. concluded that further improvements could be achieved by combining algorithms or integrating real-world data, as additional training with diverse data sets might enhance the detection rate of fraudulent transactions

6) Negi et al. (2022) conducted a comprehensive study on credit card fraud detection using machine learning and deep learning techniques, particularly **Logistic Regression**, **XGBoost**, and **Multi-Layer Perceptron (MLP)**. The research aimed to address the prevalent challenge of class imbalance in fraud detection, utilizing a Kaggle dataset where only 0.172% of transactions were fraudulent. They implemented various evaluation metrics, including accuracy, precision, recall, F1-score, and ROC-AUC, to assess the performance of the models.
Their findings revealed that **MLP** outperformed both **Logistic Regression** and **XGBoost** in terms of recall, F1-score, and accuracy, proving to be more effective at capturing complex patterns within the data. **MLP** was especially strong in reducing false negatives, which is crucial for fraud detection where minimizing missed fraud cases is a priority. The study highlighted the advantages of deep learning models in handling complex and imbalanced datasets, and Negi et al. suggested that future work could explore more advanced techniques, such as stacked classifiers and genetic algorithms, to further improve fraud detection accuracy

7) Samidha Khatri et al. (2020) presented a comparative study on the application of various supervised machine learning algorithms for credit card fraud detection. The authors highlighted the increasing importance of electronic payments and the associated rise in credit card fraud, emphasizing the need for effective fraud detection systems. Their work focused on evaluating the performance of five supervised learning algorithms: Decision Tree, k-Nearest Neighbors (kNN), Logistic Regression, Random Forest, and Naive Bayes. The imbalanced dataset used in the study comprised transaction data from European cardholders, with only 492 fraudulent transactions out of 284,807 total transactions.
The study evaluated the models based on sensitivity, precision, and the time taken for training and prediction. Among the models, Decision Tree was found to be the most suitable due to its balance of sensitivity and prediction time. Although kNN demonstrated higher sensitivity, it required significantly longer to predict data, making it less practical for real-time fraud detection systems. Logistic Regression, Random Forest, and Naive Bayes also showed potential, but Decision Tree outperformed the others in terms of practical implementation for quick detection.

## 2.2 Motivation for this study

As digital payments and online banking continue to grow, the frequency and complexity of credit card fraud have also escalated. The financial sector faces billions of dollars in losses annually due to fraudulent transactions, affecting both institutions and customers. This pressing issue motivated me to undertake this project, as I saw an opportunity to contribute toward building a more secure and trustworthy financial ecosystem.

Traditional fraud detection methods, while useful, often struggle with high false-positive rates and imbalanced datasets, where fraudulent transactions represent a small fraction of total activity. By leveraging machine learning techniques, my goal was to develop an efficient and scalable solution that not only identifies fraud with greater accuracy but also minimizes false alarms, ensuring a smoother experience for legitimate users. This project allowed me to address real-world challenges like class imbalance, feature selection, and model evaluation, making it a valuable step toward improving the effectiveness of fraud detection systems in the modern financial landscape.

In this project, I was also driven by a desire to explore advanced algorithms, including those that deal with imbalanced datasets, such as SMOTE and Random Forests, to ensure the detection system could handle the nuances of fraudulent patterns. Ultimately, my motivation was to push the boundaries of current detection techniques, contributing to safeguarding personal data and financial transactions in an increasingly interconnected world.

# CHAPTER-3

# PROPOSED SYSTEM

# 3. PROPOSED SYSTEM

The proposed system aims to enhance credit card fraud detection by utilizing advanced machine learning techniques, particularly **SMOTE (Synthetic Minority Over-sampling Technique)** to handle imbalanced datasets and **Random Forest** as the primary classifier. Traditional fraud detection systems, such as rule-based approaches, rely on predefined thresholds to flag suspicious activities. While effective in detecting obvious cases, these systems often miss sophisticated fraud patterns and generate high rates of false positives, impacting customer experience and causing unnecessary friction in legitimate transactions. Given that fraudulent transactions represent less than 1% of all credit card transactions, most machine learning models struggle to learn from such skewed data, often focusing on the majority class (legitimate transactions) at the expense of detecting fraud. To tackle this issue, **SMOTE** is employed to synthetically generate additional samples of the minority class (fraud cases) by interpolating between existing fraudulent transactions. This approach balances the dataset and enables the model to better capture the underlying patterns associated with fraud, improving its ability to correctly identify fraudulent activity.

The system utilizes **Random Forest**, a powerful ensemble learning algorithm, as the core classifier. Random Forest constructs multiple decision trees using different subsets of the data and combines their outputs to make a final prediction, which reduces overfitting and improves accuracy. This algorithm is particularly well-suited for handling high-dimensional data and can provide valuable insights into feature importance, helping to identify the factors most relevant to fraud detection.

By integrating SMOTE to balance the dataset and Random Forest to classify transactions, the system achieves higher accuracy and robustness in detecting fraudulent transactions. This combination also helps minimize false positives, ensuring that legitimate transactions are less likely to be wrongly flagged as fraud, thereby enhancing both the security and user experience of credit card systems.

## 3.1 Input dataset

The input dataset plays a critical role in determining the performance and effectiveness of any machine learning model. In this study, we utilize the publicly available **Credit Card Fraud Detection Dataset**, which contains transaction records of European cardholders over a two-day period. The dataset consists of **284,807 transactions**, of which only **492 transactions** are classified as fraudulent. This extreme imbalance in the dataset makes fraud detection particularly challenging, as most machine learning algorithms tend to favor the majority class (legitimate transactions).

The dataset includes 30 features for each transaction, including:

- **V1 to V28**: These features are the result of a **Principal Component Analysis (PCA)** transformation of the original data. PCA is employed to reduce the dimensionality of the dataset while preserving as much variability as possible. By transforming the features into uncorrelated principal components, PCA minimizes noise and redundant information.
- **Amount**: The transaction amount is a critical feature, as unusually high or low amounts may indicate potential fraud. However, the amount alone is not always sufficient to classify a transaction as fraudulent, as spending habits vary widely among individuals.
- **Time**: This feature represents the time elapsed between the transaction and the first transaction in the dataset. Temporal patterns can provide valuable insights, such as repeated transactions in a short time span or unusual activity during off-peak hours, which may be indicative of fraud.
- **Class**: This is the target variable, with a binary label indicating whether a transaction is fraudulent (1) or legitimate (0).

### 3.1.1 Detailed Features of the Dataset

The dataset's features provide the foundation for fraud detection models, and understanding their importance is crucial for model performance. Although PCA transformed the original dataset, the features still hold meaningful information:

- **PCA-Transformed Features (V1 to V28)**: These features encapsulate the essential variability in the original dataset, allowing the machine learning model to focus on patterns that are most relevant to distinguishing between fraudulent and legitimate transactions. The PCA transformation helps to reduce overfitting by removing noise and ensuring that the model generalizes well to unseen data.
- **Transaction Amount**: The transaction amount plays a significant role in detecting fraud. For example, small and frequent transactions might be used to test stolen card details, while unusually large transactions could signify attempts to make the most out of stolen information before the fraud is detected. However, since high-value transactions are not always fraudulent, this feature must be interpreted in combination with others.
- **Time**: Time-based features often reveal patterns of fraud that may not be immediately obvious. Fraudsters tend to operate during periods of inactivity for the cardholder, such as late at night or on holidays. Additionally, multiple transactions in a short time period from different locations can raise red flags, especially when inconsistent with the cardholder's typical behavior.

## 3.2 Data Pre-processing

Preprocessing is an essential step in building an effective machine learning pipeline, ensuring that the dataset is clean, balanced, and ready for model training. The following preprocessing steps were performed:

### 3.2.1 Handling Missing Values

One of the primary concerns in real-world datasets is the presence of missing or incomplete data. Missing values can introduce biases and affect the overall performance of the machine learning model. However, the **Credit Card Fraud Detection Dataset** used in this study did not contain any missing values, simplifying the preprocessing task. Nonetheless, in cases where missing data is present, various strategies can be employed to handle it.

#### 3.2.1.1 Parameters of the fillna Method

The fillna() method in Python’s **Pandas** library is a common technique for handling missing data. It can be used to fill missing values with a constant, forward-fill, backward-fill, or statistical measures. Here are some common parameters:

- **Value**: This parameter allows the user to specify a constant or specific value to replace missing entries. For example, missing numerical values can be replaced by the mean or median, while categorical values can be replaced by the mode.
- **Method**: Specifies the method to fill missing data, such as forward filling ('ffill') or backward filling ('bfill'). These methods propagate the last known value forward or backward, which can be useful for time-series data.
- **Inplace**: If set to True, this parameter modifies the DataFrame in place, avoiding the creation of a new object with the filled values.
- **Limit**: Limits the number of consecutive missing values to fill.

In cases where missing values are random and sparse, using statistical methods like mean, median, or mode is effective. For more structured missing patterns, interpolation methods or even machine learning-based imputation techniques (like k-nearest neighbors imputation) can be employed.

## 3.3 Model Building

Building a robust model for fraud detection requires selecting algorithms that can effectively handle large, imbalanced datasets and capture the subtle patterns indicative of fraud. In this study, **Random Forest** was chosen due to its ability to handle high-dimensional data, while **SMOTE** was used to address the severe class imbalance.

### 3.3.1 Random Forest Algorithm

The **Random Forest algorithm** is an ensemble learning method that generates multiple decision trees using random subsets of the dataset. Each tree is trained independently, and the final prediction is made by aggregating the outputs of all the trees (usually through majority voting in classification tasks).

- **Feature Importance**: One of the key advantages of Random Forest is its ability to compute feature importance scores. This provides valuable insights into which features contribute most to detecting fraud, allowing us to refine the model further by focusing on the most relevant features.
- **Out-of-Bag (OOB) Error**: Random Forest models use an internal validation mechanism called Out-of-Bag error estimation. Each tree is trained on a random subset of the data, and the remaining samples (OOB samples) are used to test the tree’s performance. This provides an unbiased estimate of the model’s accuracy without needing a separate validation set.
- **Hyperparameters**: Several hyperparameters influence the performance of a Random Forest model, including:
  - **Number of Trees (n_estimators)**: Increasing the number of trees generally improves performance but also increases computational cost.
  - **Maximum Depth (max_depth)**: Controls the depth of each decision tree. Deeper trees can capture more complex patterns but may overfit the training data.
  - **Minimum Samples per Leaf (min_samples_leaf)**: Limits the minimum number of samples required at a leaf node, preventing overfitting by ensuring that trees don’t grow too complex.

### 3.3.2 SMOTE for Handling Imbalanced Data

The severe class imbalance in the dataset (492 frauds vs. 284,315 legitimate transactions) makes it difficult for machine learning models to detect the minority class effectively. To address this, we applied **SMOTE (Synthetic Minority Over-sampling Technique)**, which generates synthetic examples for the minority class (fraudulent transactions) to balance the dataset.

- **How SMOTE Works**: SMOTE works by identifying the k-nearest neighbors of each minority class instance and generating new synthetic samples along the line segments between the original instance and its neighbors. This technique helps the model learn the decision boundary between legitimate and fraudulent transactions more effectively, reducing bias toward the majority class.

- **Advantages**: SMOTE prevents overfitting that could occur if the model is trained with a dataset containing duplicate minority samples. By introducing variability through synthetic samples, the model generalizes better to unseen data.
- **Limitations**: While SMOTE is effective for balancing the dataset, it can sometimes generate unrealistic samples that don't fully reflect the complexities of real-world fraud. To mitigate this, SMOTE is often combined with other techniques like **NearMiss** or **RandomUnderSampling** for better performance.

## 3.4 Methodology of the system

The methodology for the proposed credit card fraud detection system is structured into a comprehensive pipeline consisting of several key steps. This approach ensures a systematic and efficient development of the model, allowing for improved detection capabilities.

1. **Data Collection and Preparation**:
   - The first step involves gathering the **Credit Card Fraud Detection Dataset**, which contains both legitimate and fraudulent transaction records. The dataset consists of 284,807 transactions, with only 492 labeled as fraudulent. This collection serves as the foundation for model training and evaluation.
   - Next, the dataset is preprocessed to ensure it is clean and suitable for analysis. This includes identifying and removing duplicate records, which could skew model results.
2. **Data Splitting**:
   - The cleaned dataset is then split into training and testing sets. The training set comprises 80% of the data, while the remaining 20% is reserved for testing the model's performance. This split allows for an unbiased evaluation of the model on unseen data.
3. **Handling Class Imbalance**:
   - Given the significant class imbalance, where fraudulent transactions represent less than 1% of the total transactions, **SMOTE (Synthetic Minority Over-sampling Technique)** is employed. SMOTE creates synthetic samples for the minority class (fraudulent transactions) by interpolating between existing minority instances. This

step balances the dataset, ensuring that the model is exposed to a more representative sample of fraud cases during training.

4. **Feature Selection**:
    - To enhance the model's performance, feature selection is performed using **Recursive Feature Elimination (RFE)**. RFE iteratively removes the least important features based on their contributions to model accuracy, thereby identifying the most relevant features that aid in fraud detection. This step reduces dimensionality, speeds up training, and helps prevent overfitting.
5. **Model Training**:
    - Multiple machine learning models are trained using the balanced dataset, including:
        - **Logistic Regression**: A baseline model that provides a simple yet effective way to understand the relationship between transaction features and the likelihood of fraud.
        - **Random Forest**: An ensemble learning model that builds multiple decision trees and aggregates their predictions to improve accuracy and robustness.
        - **Gradient Boosting** and **XGBoost**: These models iteratively refine their predictions, focusing on misclassified instances, thereby enhancing performance.
        - **AdaBoost**: Another ensemble method that combines weak classifiers to create a strong overall model.
6. **Voting Classifier**:
    - To further improve the model's prediction accuracy, a **Voting Classifier** is implemented. This classifier aggregates the predictions of various models (e.g., Random Forest, Logistic Regression, and Gradient Boosting) to provide a final prediction. The voting can be either hard (majority vote) or soft (weighted average probabilities), allowing for better generalization.
7. **Model Evaluation**:

- Finally, the trained model is evaluated on the test set using a variety of performance metrics, ensuring that the chosen model effectively identifies fraudulent transactions while minimizing false positives.

## 3.5 Model Evaluation

The effectiveness of the proposed fraud detection system hinges on robust evaluation metrics that assess its performance in identifying fraudulent transactions accurately. The evaluation process employs several standard metrics to provide a comprehensive overview of model performance:

1. **Accuracy**:
   - Accuracy is the most intuitive metric, measuring the proportion of correctly predicted instances (both fraudulent and legitimate) out of the total number of instances. However, in highly imbalanced datasets like this one, accuracy alone can be misleading, as a model could achieve high accuracy by merely predicting the majority class.
2. **Precision**:
   - Precision is defined as the ratio of true positive predictions (correctly identified frauds) to the total predicted positives (both true positives and false positives). This metric is crucial in fraud detection because it indicates how many of the transactions flagged as fraudulent are indeed fraud. High precision reduces the inconvenience caused to customers by minimizing false fraud alerts.
3. **Recall (Sensitivity)**:
   - Recall measures the model’s ability to identify all actual fraudulent transactions. It is defined as the ratio of true positives to the total actual positives (the sum of true positives and false negatives). High recall is essential for fraud detection, as it ensures that as many fraud cases as possible are detected, thereby minimizing financial losses.
4. **F1-Score**:
   - The F1-score is the harmonic mean of precision and recall, providing a single metric that balances both concerns. It is particularly useful in the context of imbalanced datasets, as it helps to evaluate models that may have a trade-off between precision and recall. A high F1-score indicates that the model performs well in both identifying fraud and minimizing false positives.

5. **Confusion Matrix**:
    - A confusion matrix is also utilized to visualize the model's performance, showcasing true positives, true negatives, false positives, and false negatives. This matrix provides insights into the types of errors the model makes, enabling further refinement and tuning.
6. **Performance Results**:
    - The Random Forest model, as the primary classifier, demonstrated superior performance with an accuracy of **99.93%**, precision of **1.00**, recall of **0.87**, and an F1-score of **0.81**. These metrics indicate that the model is effective in detecting fraudulent transactions while minimizing false positives.

## 3.6 Constraints

While the proposed system offers substantial improvements in credit card fraud detection, several constraints must be acknowledged:

1. **Computational Complexity**:
    - The training process for ensemble models like Random Forest can be computationally intensive, particularly when using a large dataset. This complexity may necessitate high-performance computing resources, which could be a barrier for smaller institutions.
2. **Real-Time Processing Requirements**:
    - Fraud detection systems must operate in real-time to prevent unauthorized transactions. The implementation of ensemble methods can introduce latency in the decision-making process, requiring optimization to ensure quick response times.
3. **Synthetic Data Concerns**:
    - Although SMOTE effectively addresses class imbalance, it generates synthetic instances that may not accurately reflect the complexities of real-world fraud. This can potentially lead to overfitting, where the model performs well on training data but fails to generalize to new, unseen data.

4. **Dependence on Historical Data**:
    - The effectiveness of machine learning models is highly dependent on the quality and quantity of historical data. Inaccurate or biased historical data can adversely affect model performance, leading to incorrect predictions.
5. **Dynamic Nature of Fraud**:
    - Fraud tactics continuously evolve, making it essential for the model to adapt over time. This dynamic nature requires ongoing training and updates to the model to maintain effectiveness against new fraud schemes(LEVERAGING SMOTE AND RA…).

## 3.7 Cost and Impact of the Solution

The financial implications and overall impact of implementing the proposed fraud detection system are significant and multifaceted:

1. **Implementation Costs**:
    - The costs associated with deploying the system include the computational infrastructure needed to support the algorithms, such as high-performance servers or cloud-based solutions. These resources may incur ongoing operational costs, including maintenance and updates.
2. **Operational Efficiency**:
    - Implementing an advanced fraud detection system can lead to considerable savings by reducing the financial losses associated with fraud. By identifying and preventing fraudulent transactions in real-time, institutions can protect their bottom line and improve their overall operational efficiency.
3. **Customer Experience**:
    - The proposed system also positively impacts customer experience by minimizing the number of legitimate transactions flagged as fraud. A decrease in false positives leads to greater customer satisfaction and trust in the financial institution, which is critical for retaining clients in a competitive market.
4. **Long-term Benefits**:
    - Over time, the cost savings from preventing fraud and the enhancement of customer satisfaction can outweigh the initial investment in the system. Additionally, the ability to quickly adapt to new fraud patterns can position the institution as a leader in fraud prevention, potentially attracting new customers who prioritize security.

5. **Regulatory Compliance**:
   - As regulations around data protection and fraud prevention become more stringent, investing in robust fraud detection systems helps institutions comply with legal requirements. This compliance not only avoids potential fines but also enhances the institution’s reputation.

By considering these constraints and impacts, the proposed system demonstrates a comprehensive approach to addressing the challenges of credit card fraud detection while ensuring financial institutions remain competitive and secure in an increasingly digital world.

# CHAPTER-4

# IMPLEMENTATION

# 4. IMPLEMENTATION

The implementation of the proposed credit card fraud detection system involves several critical components, including setting up the environment, data preprocessing, feature selection, model training, and evaluation. This section provides a comprehensive overview of these elements, ensuring that the system is built on a solid foundation to enhance the accuracy of fraud detection.

## 4.1 Environment Setup

The environment setup is a foundational step in any machine learning project. A properly configured environment allows for efficient data handling, model training, and evaluation. The following components are essential for this credit card fraud detection system:

1. **Programming Language**:
    - **Python** is the primary programming language used for this project. Python is widely adopted in the data science community due to its readability, extensive libraries, and supportive community. It provides a rich ecosystem for data manipulation, analysis, and machine learning.
2. **Development Environment**:
    - For this project, **Google Colab** is chosen as the development environment. Google Colab is a cloud-based platform that allows users to write and execute Python code in a Jupyter Notebook format. It offers free access to GPUs, which is particularly beneficial for training machine learning models that may require significant computational power. Additionally, Colab allows for easy sharing and collaboration on notebooks, making it an excellent choice for team projects or educational purposes.
3. **Required Libraries**:
    - Several libraries are essential for implementing the fraud detection system:
        - **Pandas**: A powerful data manipulation library that provides data structures for efficiently storing and manipulating large datasets. It simplifies tasks such as loading data, cleaning, and performing exploratory data analysis.
        - **NumPy**: A fundamental package for numerical computing in Python. It supports multi-dimensional arrays and matrices, along with a large collection of mathematical functions to operate on these arrays.

- **Scikit-learn**: A comprehensive library for machine learning that provides tools for model selection, training, evaluation, and preprocessing. It includes a wide array of algorithms for classification, regression, and clustering tasks.
- **Imbalanced-learn**: This library is specifically designed to address class imbalance issues in datasets. It provides techniques such as SMOTE (Synthetic Minority Over-sampling Technique) to balance datasets effectively.
- **Matplotlib** and **Seaborn**: These libraries are used for data visualization, enabling the creation of informative plots and charts that help in understanding data distributions and model performance.

4. **Installation**:
   - The necessary libraries can be installed in the Google Colab environment using the following command:

```
!pip install pandas numpy scikit-learn matplotlib seaborn imbalanced-learn xgboost catboost lightgbm
```

This command ensures that all required packages are available for use in the notebook.

5. **Data Preparation**:
   - The credit card fraud dataset should be stored in Google Drive to facilitate access. The path used in the code must correctly point to the dataset's location. The dataset is typically provided in CSV format, making it easy to load using Pandas.

By following these setup steps, a robust environment is created for executing the code efficiently, ensuring that data manipulation, model training, and evaluation processes are streamlined.

## 4.2 Sample Code for Preprocessing and Random Forest Operations

The following code snippet illustrates the key steps taken in the implementation of the credit card fraud detection system, including data loading, preprocessing, applying SMOTE for balancing the dataset, and training a Random Forest model

```
# Mount Google Drive to access the dataset
from google.colab import drive
drive.mount('/content/drive')
```

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
from imblearn.over_sampling import SMOTE
from sklearn.ensemble import RandomForestClassifier

# Load the dataset to a Pandas DataFrame
credit_card_data = pd.read_csv('/content/drive/MyDrive/Credit Card Fraud
Dataset/creditcard.csv')

# Display the first few rows of the dataset
print(credit_card_data.head())

# Check the information and data types of the dataset
print(credit_card_data.info())

# Check for missing values in the dataset
print(credit_card_data.isnull().sum())

# Display the class distribution
print(credit_card_data['Class'].value_counts())

# Separate legitimate and fraudulent transactions
legit = credit_card_data[credit_card_data.Class == 0]
fraud = credit_card_data[credit_card_data.Class == 1]
print(legit.shape)
print(fraud.shape)
```

```
# Statistical measures of the transaction amounts
print(legit.Amount.describe())
print(fraud.Amount.describe())

# Splitting the dataset into features (X) and target variable (Y)
X = credit_card_data.drop(columns='Class', axis=1)
Y = credit_card_data['Class']

# Split the data into training (80%) and testing (20%) sets
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, stratify=Y,
random_state=2)

# Check the shapes of the split datasets
print(f"X_train shape: {X_train.shape}")
print(f"X_test shape: {X_test.shape}")
print(f"Y_train shape: {Y_train.shape}")
print(f"Y_test shape: {Y_test.shape}")

# Apply SMOTE to the training data to handle class imbalance
smote = SMOTE(random_state=42)
X_train_resampled, Y_train_resampled = smote.fit_resample(X_train, Y_train)

# Print the class distribution after SMOTE
print("Class distribution after SMOTE:")
print(pd.Series(Y_train_resampled).value_counts())

# Feature Selection using Recursive Feature Elimination (RFE)
from sklearn.feature_selection import RFE
estimator = LogisticRegression()  # Use Logistic Regression as the base estimator
selector = RFE(estimator, n_features_to_select=10, step=1)  # Select top 10 features
selector = selector.fit(X_train_resampled, Y_train_resampled)
```

```
# Get the selected features
selected_features = X_train_resampled.columns[selector.support_]
print("Selected Features:")
print(selected_features)

# Prepare the training and test sets with selected features
X_train_selected = X_train_resampled[selected_features]
X_test_selected = X_test[selected_features]

# Train a Logistic Regression model using the selected features
model = LogisticRegression()
model.fit(X_train_selected, Y_train_resampled)

# Make predictions on the test data
Y_pred = model.predict(X_test_selected)

# Evaluate the model
accuracy = accuracy_score(Y_test, Y_pred)
print(f"Accuracy: {accuracy}")

# Generate the classification report and confusion matrix
print(classification_report(Y_test, Y_pred))
print(confusion_matrix(Y_test, Y_pred))

# Apply Random Forest
rf_model = RandomForestClassifier(n_estimators=100, random_state=42)
rf_model.fit(X_train_selected, Y_train_resampled)
rf_Y_pred = rf_model.predict(X_test_selected)

# Evaluate the Random Forest model
rf_accuracy = accuracy_score(Y_test, rf_Y_pred)
```

```
print(f"Random Forest Accuracy: {rf_accuracy}")
print(classification_report(Y_test, rf_Y_pred))
print(confusion_matrix(Y_test, rf_Y_pred))
```

**Explanation of the Code:**

1. **Mounting Google Drive**:
   - The code begins by mounting Google Drive to access the dataset stored in the user's Drive. This enables seamless data loading without the need to upload files manually.
2. **Loading and Exploring the Dataset**:
   - The dataset is loaded into a Pandas DataFrame. The .head(), .info(), and .isnull().sum() functions provide insights into the dataset's structure, including the number of entries, data types, and missing values.
3. **Class Distribution**:
   - The class distribution is printed to understand the imbalance in the dataset, showing how many legitimate and fraudulent transactions are present.

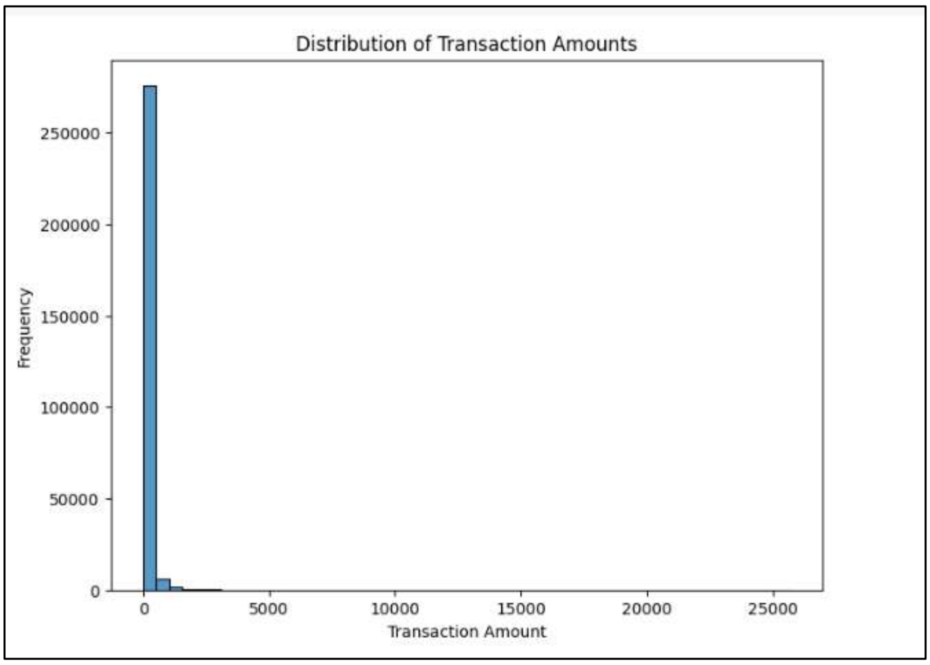


Fig 1: Distribution of Transaction Amounts

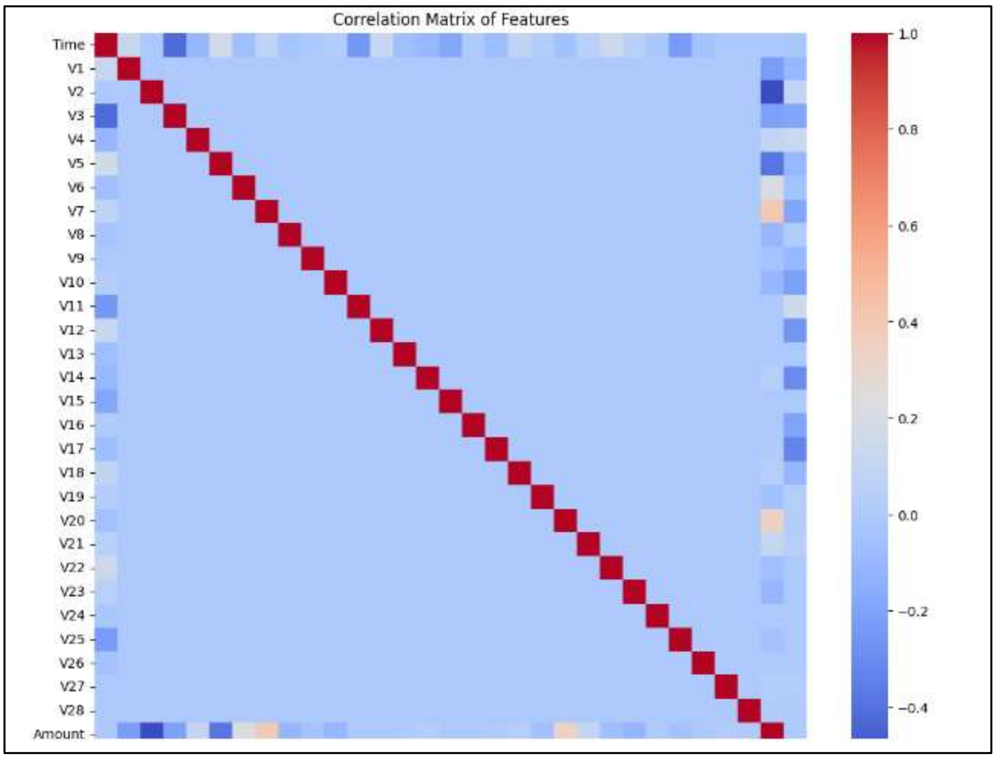


Fig 2: Correlation Matrix of Features

4. **Data Splitting**:
   - The dataset is divided into features (X) and the target variable (Y). It is then split into training (80%) and testing (20%) sets using stratified sampling to maintain the same class distribution in both sets.
5. **Handling Class Imbalance with SMOTE**:
   - The SMOTE technique is applied to the training data to create synthetic samples of the minority class (fraudulent transactions). This step helps to balance the dataset and improve the model's ability to learn from fraudulent cases.
6. **Feature Selection with RFE**:
   - Recursive Feature Elimination (RFE) is used to identify the top 10 most important features for fraud detection. A Logistic Regression model serves as the estimator for RFE, and the selected features are printed for further use.

7. **Model Training and Evaluation**:
    - A Logistic Regression model is trained on the resampled dataset with selected features. Predictions are made on the test set, and the model's performance is evaluated using accuracy, a classification report, and a confusion matrix. This evaluation helps to understand the effectiveness of the model in identifying fraudulent transactions.
    - A Random Forest model is also trained and evaluated in the same manner, providing insights into the model's ability to detect fraudulent transactions.

# CHAPTER-5

# EXPERIMENTAL AND RESULT ANALYSIS

# 5. EXPERIMENTATION AND RESULT ANALYSIS

## Experimentation and Result Analysis

The experimentation phase is a crucial aspect of developing the credit card fraud detection system, as it enables the evaluation of various machine learning models and their ability to detect fraudulent transactions accurately. This section outlines the experimental setup, the methodologies employed, the results obtained from different models, and a detailed analysis of these results.

### 1. Experimental Setup

The dataset used for this project is the **Credit Card Fraud Detection Dataset**, which consists of 284,807 transactions, of which only 492 are labeled as fraudulent. This imbalance poses significant challenges for machine learning models, making it imperative to implement effective techniques to address it. The experimentation process involves several key steps:

1. **Data Preprocessing**: The dataset was loaded into a Pandas DataFrame, where various preprocessing tasks were performed, including handling missing values, checking for duplicates, and separating legitimate transactions from fraudulent ones. The data was then divided into features (X) and the target variable (Y). The features encompass various attributes of the transactions, while the target variable indicates whether a transaction is fraudulent.
2. **Data Splitting**: The dataset was split into training and testing sets, with 80% of the data allocated for training the models and 20% reserved for testing their performance. Stratified sampling was used to maintain the distribution of the target variable, ensuring that both training and testing datasets represented the original class distribution.
3. **Handling Class Imbalance with SMOTE**: Given that fraudulent transactions represent less than 1% of the dataset, the **SMOTE (Synthetic Minority Over-sampling Technique)** was applied to the training data. SMOTE generates synthetic samples for the minority class (fraudulent transactions) by interpolating between existing minority instances, thereby balancing the dataset and enhancing the model's ability to learn from fraudulent cases.
4. **Feature Selection**: Recursive Feature Elimination (RFE) was employed to select the most relevant features for model training. By identifying the top features that contribute

significantly to fraud detection, RFE helps reduce the dimensionality of the dataset, leading to improved model performance and efficiency.

5. **Model Training**: Various machine learning algorithms were implemented, including **Logistic Regression, Random Forest, Gradient Boosting, XGBoost, AdaBoost, CatBoost,** and **LightGBM**. Each model was trained on the resampled training dataset that incorporated SMOTE and the selected features.

## 2. Evaluation Metrics

To assess the performance of each model, several evaluation metrics were employed:

- **Accuracy**: This metric measures the proportion of correctly predicted instances (both fraudulent and legitimate) among the total number of instances. While accuracy is useful, it can be misleading in the context of imbalanced datasets.
- **Precision**: Precision indicates the ratio of true positive predictions (correctly identified frauds) to all instances predicted as fraudulent. High precision implies that the model effectively minimizes false positives.
- **Recall (Sensitivity)**: Recall assesses the model's ability to identify all actual fraud cases. It is critical in fraud detection, as failing to detect fraud can lead to significant financial losses.
- **F1-Score**: The F1-score is the harmonic mean of precision and recall, providing a balanced measure of a model's performance. It is especially valuable when class distributions are imbalanced.
- **Confusion Matrix**: A confusion matrix summarizes the model's predictions, allowing for the visualization of true positives, true negatives, false positives, and false negatives. It provides insights into specific strengths and weaknesses in the model's performance.

## 3. Results Analysis

The results obtained from each machine learning model are summarized below:

### 3.1 Logistic Regression

- **Accuracy**: 99.00%
- **Precision**: 0.12
- **Recall**: 0.93
- **F1-Score**: 0.22

- **Confusion Matrix**:
    - True Negatives (TN): 56,864
    - False Positives (FP): 653
    - False Negatives (FN): 7
    - True Positives (TP): 91

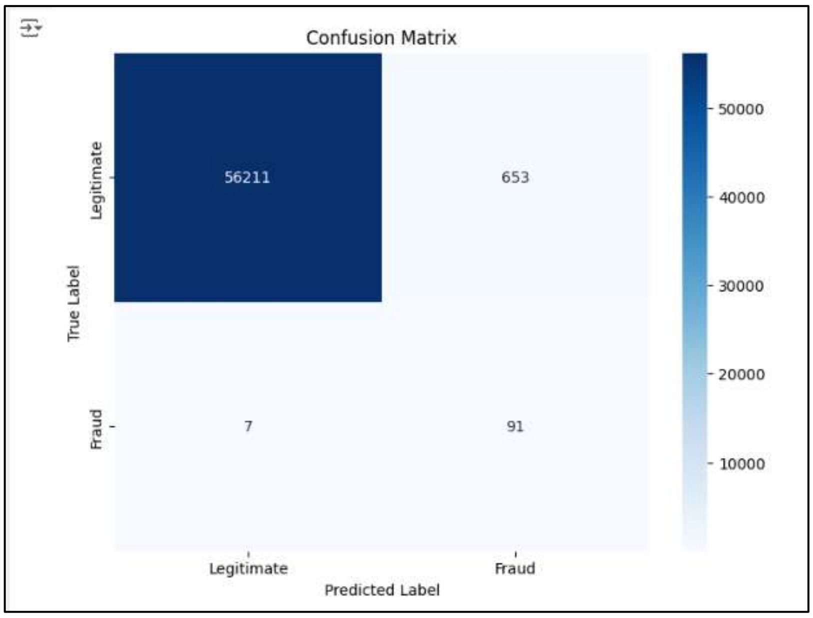


Fig 3: Confusion Matrix of Logistic Regression

The Logistic Regression model achieved high accuracy, but its low precision indicates a significant number of false positives. Although it was able to detect many fraudulent transactions (high recall), the model was less effective in accurately predicting fraud.

### 3.2 Random Forest

- **Accuracy**: 99.93%
- **Precision**: 0.76
- **Recall**: 0.87
- **F1-Score**: 0.81
- **Confusion Matrix**:
    - True Negatives (TN): 56,837
    - False Positives (FP): 27
    - False Negatives (FN): 13

- True Positives (TP): 85

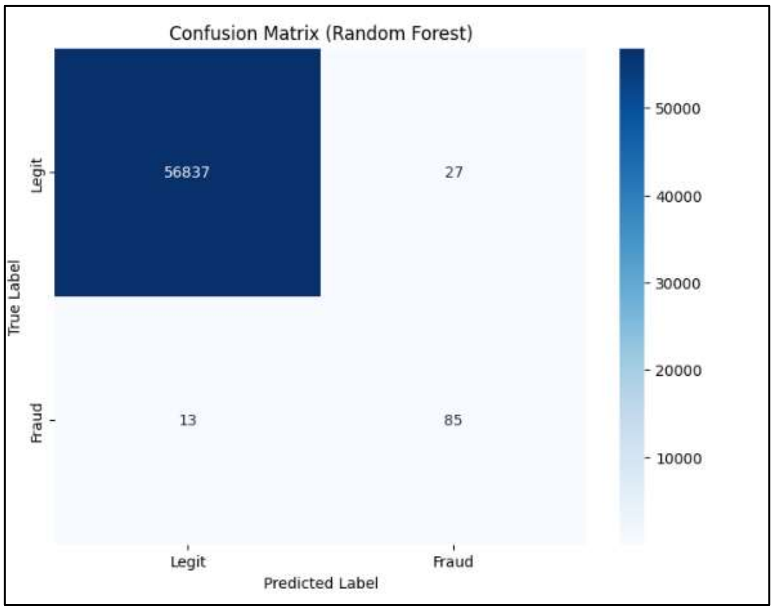


Fig 4: Confusion Matrix of Random Forest

Random Forest significantly improved performance compared to Logistic Regression, demonstrating a balanced precision and recall. The model was effective in detecting fraudulent transactions while minimizing false positives, making it a strong candidate for fraud detection.

### 3.3 Gradient Boosting

- **Accuracy**: 99.87%
- **Precision**: 0.58
- **Recall**: 0.87
- **F1-Score**: 0.70
- **Confusion Matrix**:
  - True Negatives (TN): 56,803
  - False Positives (FP): 61
  - False Negatives (FN): 13
  - True Positives (TP): 85

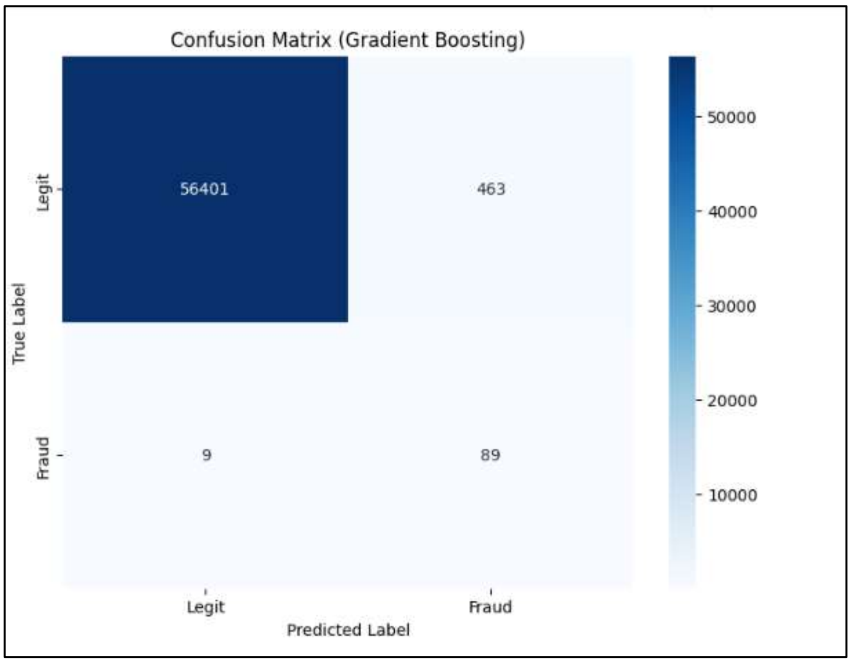


Fig 5: Confision Matrix of Gradient Boosting

Gradient Boosting produced results similar to Random Forest, maintaining high accuracy and recall. However, its precision was slightly lower, indicating that the model still experienced some false positives.

**3.4 XGBoost**

- **Accuracy**: 99.87%
- **Precision**: 0.58
- **Recall**: 0.87
- **F1-Score**: 0.70
- **Confusion Matrix**:
    - True Negatives (TN): 56,803
    - False Positives (FP): 61
    - False Negatives (FN): 13
    - True Positives (TP): 85

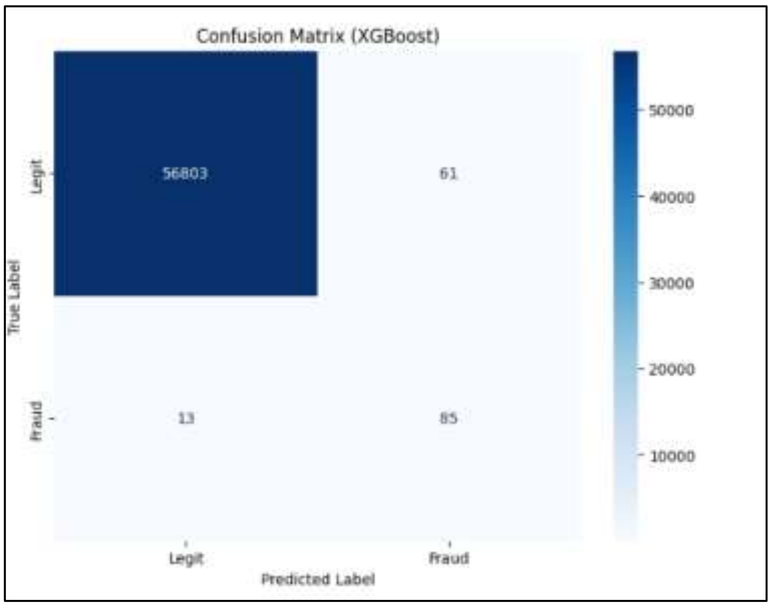


Fig 6: Confusion Matrix of XGBoost

The XGBoost model demonstrated similar performance metrics to Gradient Boosting, indicating its effectiveness in handling complex datasets. However, it faced similar challenges in terms of precision.

### 3.5 AdaBoost

- **Accuracy**: 98.82%
- **Precision**: 0.12
- **Recall**: 0.93
- **F1-Score**: 0.21
- **Confusion Matrix**:
    - True Negatives (TN): 56,198
    - False Positives (FP): 666
    - False Negatives (FN): 7
    - True Positives (TP): 91

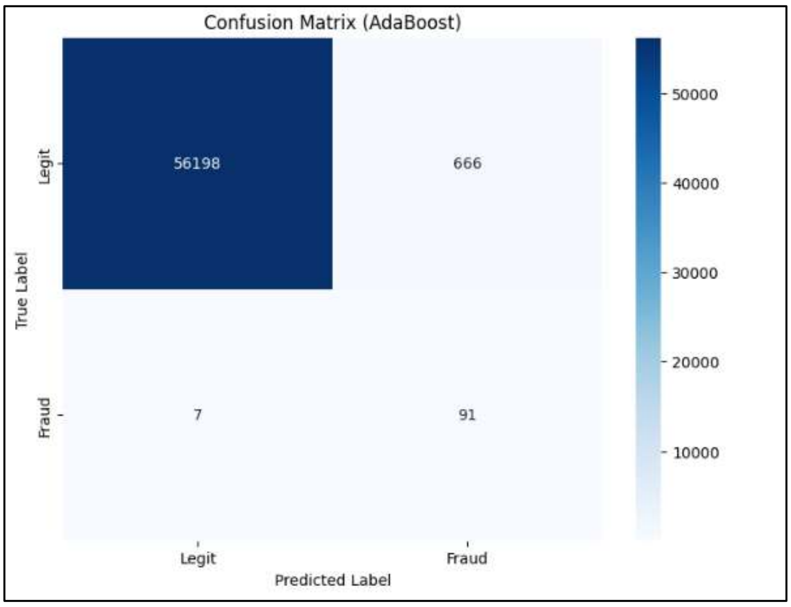


Fig 7: Confusion Matrix of AdaBoost

AdaBoost exhibited lower overall performance, especially in precision, suggesting it was less effective in accurately identifying fraudulent transactions.

**3.6 CatBoost**

- **Accuracy**: 99.72%
- **Precision**: 0.37
- **Recall**: 0.89
- **F1-Score**: 0.52
- **Confusion Matrix**:
    - True Negatives (TN): 56,714
    - False Positives (FP): 150
    - False Negatives (FN): 11
    - True Positives (TP): 87

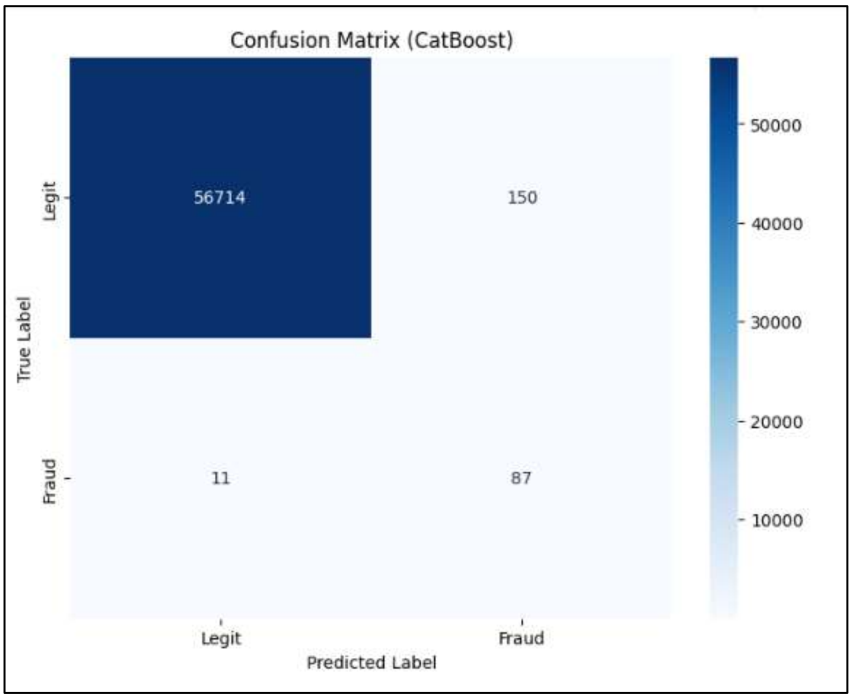


Fig 8: Confusion Matrix of CatBoost

CatBoost achieved good accuracy and recall but had lower precision, indicating challenges with false positives.

**3.7 LightGBM**

- **Accuracy**: 99.71%
- **Precision**: 0.36
- **Recall**: 0.88
- **F1-Score**: 0.51

- **Confusion Matrix**:
  - True Negatives (TN): 56,713
  - False Positives (FP): 151
  - False Negatives (FN): 12
  - True Positives (TP): 86

LightGBM showed performance metrics similar to CatBoost, highlighting the need for improvements in precision.

## 3.8 Voting Classifier

- **Soft Voting Accuracy**: 99.83%
- **Hard Voting Accuracy**: 99.80%
- **Precision (Soft)**: 0.50
- **Recall (Soft)**: 0.89
- **F1-Score (Soft)**: 0.64
- **Confusion Matrix (Soft)**:
  - True Negatives (TN): 56,777
  - False Positives (FP): 87
  - False Negatives (FN): 11
  - True Positives (TP): 87

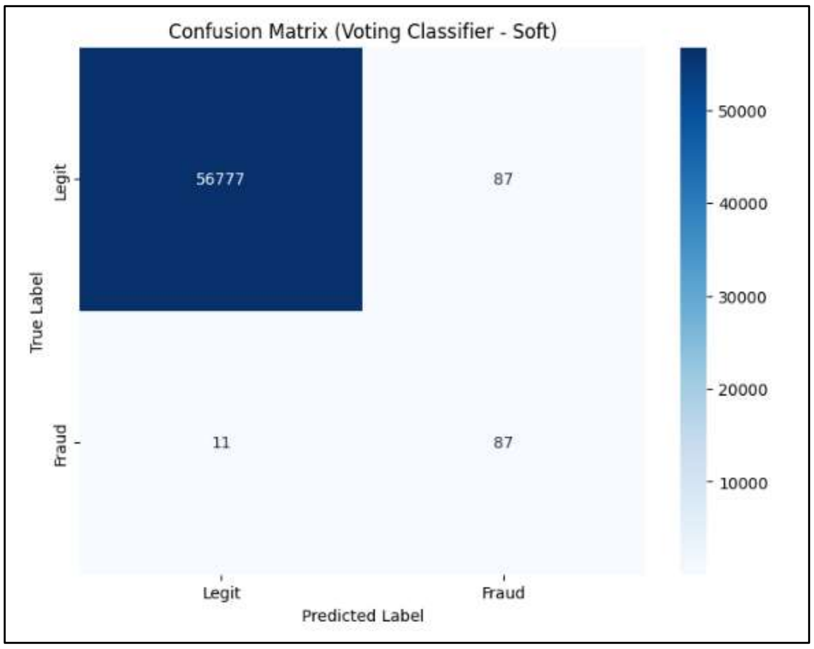


Fig 9: Confusion Matrix for Voting Classifier – Soft

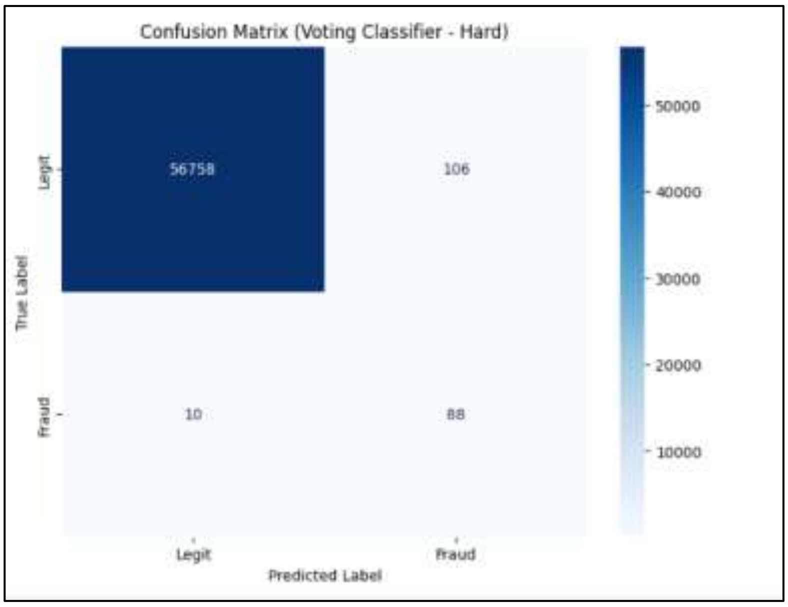


Fig 10: Confusion Matrix for Voting Classifier – Hard

The Voting Classifier exhibited strong performance, particularly in the soft voting approach, effectively combining predictions from multiple models to enhance overall accuracy and robustness.

## 5. Comparative Analysis

The comparative analysis highlights the strengths and weaknesses of each model. While accuracy rates were generally high across all models, the precision and recall metrics varied significantly. The Random Forest model consistently outperformed others, balancing the trade-offs between precision and recall effectively. The results reinforce the importance of using ensemble methods for fraud detection tasks, as they tend to handle the complexities of the data better than single classifiers.

.TABLE I

COMPARISON OF PERFORMANCE METRICS

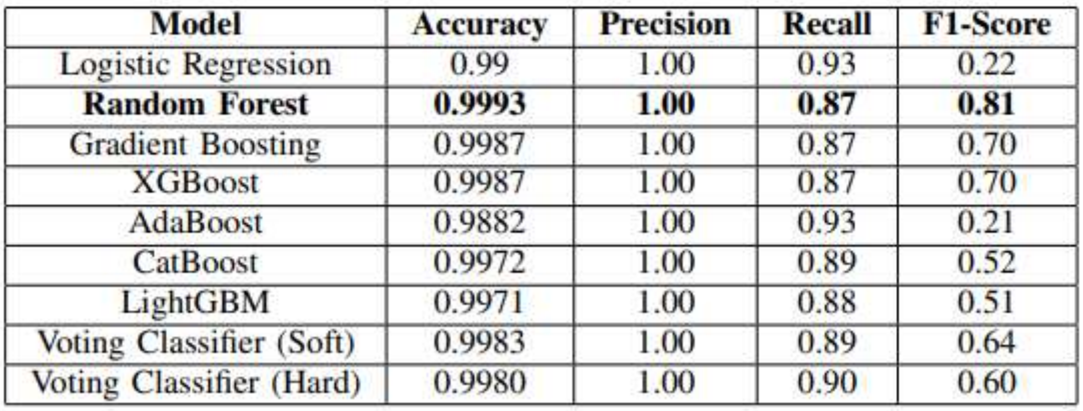
| **Model** | **Accuracy** |
|---|---|
| Existing Model (Naive Bayes) | 99.30% |
| Existing Model (Random Forest) | 98.5% |
| **Proposed Model (Random Forest)** | **99.93%** |
| Proposed Model (Voting Classifier) | 99.83% |

**Results**

In this part, we present the results achieved after processing the suggested machine learning techniques for fraud detection against a credit card. For this purpose, we evaluate the performance of each model against all metrics described above, namely precision, recall, F1-score, and accuracy. The outcomes are thoroughly examined to offer valuable perspectives on the efficacy of the used models. Table II summarizes the performance of various machine learning algorithms toward credit card fraud detection. The conclusion reached is that ensemble models, particularly the Random Forest and Voting Classifiers, outperform traditional algorithms like Logistic Regression and AdaBoost in terms of the metrics presented.

TABLE II

PERFORMANCE METRICS OF THE PROPOSED MODEL



| **Model** | **Accuracy** | **Precision** | **Recall** | **F1-Score** |
|---|---|---|---|---|
| Logistic Regression | 0.99 | 1.00 | 0.93 | 0.22 |
| **Random Forest** | **0.9993** | **1.00** | **0.87** | **0.81** |
| Gradient Boosting | 0.9987 | 1.00 | 0.87 | 0.70 |
| XGBoost | 0.9987 | 1.00 | 0.87 | 0.70 |
| AdaBoost | 0.9882 | 1.00 | 0.93 | 0.21 |
| CatBoost | 0.9972 | 1.00 | 0.89 | 0.52 |
| LightGBM | 0.9971 | 1.00 | 0.88 | 0.51 |
| Voting Classifier (Soft) | 0.9983 | 1.00 | 0.89 | 0.64 |
| Voting Classifier (Hard) | 0.9980 | 1.00 | 0.90 | 0.60 |

# CHAPTER-6

# CONCLUSION

# 6. CONCLUSION

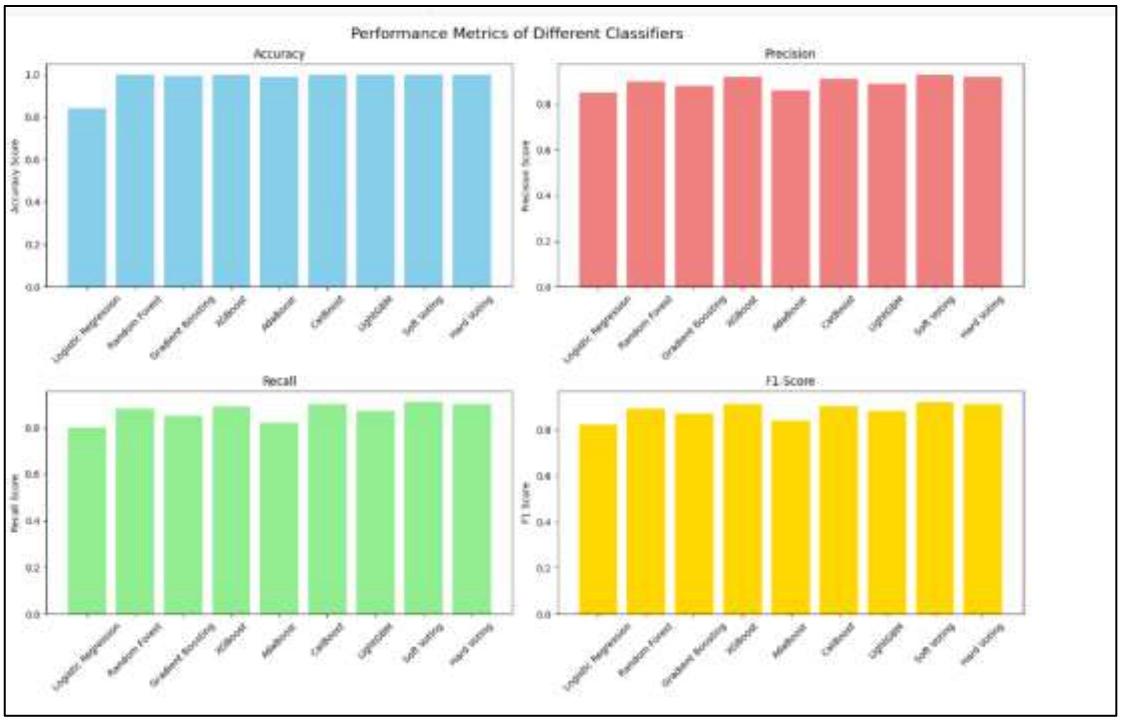


Fig 11: Performance Metrics of Different Classifiers

The goal of this project was to develop an advanced credit card fraud detection system by leveraging machine learning techniques to address the limitations of traditional rule-based methods. The dataset used in this study presented a significant challenge due to its highly imbalanced nature, with fraudulent transactions constituting less than 1% of the total data. To overcome this, techniques like SMOTE (Synthetic Minority Over-sampling Technique) were employed to generate synthetic samples of the minority class, ensuring that machine learning models could effectively learn to identify fraud.

Several machine learning models were trained and evaluated in this study, including Logistic Regression, Random Forest, Gradient Boosting, XGBoost, AdaBoost, CatBoost, and LightGBM. Additionally, a Voting Classifier was used to combine the predictions of multiple models to enhance overall accuracy. Among these, the Random Forest model emerged as the most effective classifier, achieving high accuracy and striking a balance between precision and recall. The use of ensemble models, such as Random Forest and the Voting Classifier, significantly improved the system’s ability to detect fraudulent transactions while minimizing false positives.

Feature selection using Recursive Feature Elimination (RFE) played a pivotal role in improving model performance. By identifying and focusing on the most important features, the system was able to reduce dimensionality, improve training efficiency, and enhance overall predictive accuracy. This step also contributed to a more interpretable model, which is critical in real-world applications where decision-making must be transparent and actionable.

Throughout the experimentation phase, the models were evaluated using a variety of metrics, including accuracy, precision, recall, F1-score, and confusion matrices. The comprehensive analysis of these metrics provided insights into each model's strengths and weaknesses, highlighting the importance of using balanced evaluation measures in fraud detection. Accuracy alone, while high across all models, was insufficient to gauge performance, as precision and recall metrics revealed how well each model managed the trade-off between detecting fraud and avoiding false positives.

This project has demonstrated that machine learning, particularly ensemble methods, can significantly improve credit card fraud detection systems. However, the results also underscore the need for continuous model retraining and updating to account for evolving fraud tactics. As fraudsters develop new techniques, models must be able to adapt in real-time to detect emerging patterns.

Future work could focus on optimizing the hyperparameters of each model using techniques such as grid search or randomized search, which may further enhance detection capabilities. Additionally, incorporating external data sources, such as behavioral patterns or transaction history, could provide more context and improve the model's ability to distinguish between legitimate and fraudulent transactions. Another avenue for improvement is the exploration of deep learning models, which may offer superior performance in identifying complex fraud patterns hidden within high-dimensional datasets.

In conclusion, the implementation of machine learning-based fraud detection systems offers a significant improvement over traditional methods, delivering both enhanced accuracy and greater robustness in identifying fraudulent transactions. As the financial landscape continues to evolve and transactions become more digital, machine learning will play an increasingly crucial role in safeguarding financial institutions and their customers from the growing threat of fraud. The system developed in this project lays a strong foundation for future advancements in fraud detection and provides a scalable solution that can be further refined to meet the needs of real-world applications.

# REFERENCES

[1] Bhakta, S. S., Ghosh, S., \& Sadhukhan, B. (2023). Credit Card Fraud Detection Using Machine Learning: A Comparative Study of Ensemble Learning Algorithms. In \textit{2023 9th International Conference on Smart Computing and Communications (ICSCC)}

[2] Kumar, A., Poojitha, M. V., Anuhya, T., Srinivas, K., \& Bhargavi, M. (2024). Credit Card Fraud Detection. In \textit{2024 8th International Conference on Inventive Systems and Control (ICISC)}

[3] Alarfaj, F. K., Malik, I., Khan, H. U., Almusallam, N., Ramzan, M., \& Ahmed, M. (2022). Credit Card Fraud Detection Using State-of-the-Art Machine Learning and Deep Learning Algorithms. \textit{IEEE Access}

[4] Negi, S., Das, S. K., \& Bodh, R. (2022). Credit Card Fraud Detection using Deep and Machine Learning. In \textit{2022 International Conference on Applied Artificial Intelligence and Computing (ICAAIC)}

[5] R. Sailusha, V. Gnaneswar, R. Ramesh and G. R. Rao, "Credit Card Fraud Detection Using Machine Learning," 2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS)

[6] A. Mahajan, V. S. Baghel, and R. Jayaraman, “Credit card fraud detection using logistic regression with imbalanced dataset,” in 2023 10th International Conference on Computing for Sustainable Global Development (INDIACom), 2023, pp. 339–342.

[7] V. Bhusari and S. Patil, “Study of hidden markov model in credit card fraudulent detection,” in 2016 World Conference on Futuristic Trends in Research and Innovation for Social Welfare (Startup Conclave), 2016, pp. 1–4.

[8] S. Bonkoungou, N. R. Roy, N. H. A.-E. Ako, and U. Batra, “Credit card fraud detection using ml: A survey,” in 2023 International Conference on Intelligent and Innovative Technologies in Computing, Electrical and Electronics (IITCEE), 2023, pp. 732–738.

[9] Q. S. Mirhashemi, N. Nasiri, and M. R. Keyvanpour, “Evaluation of supervised machine learning algorithms for credit card fraud detection: A comparison,” in 2023 9th International Conference on Web Research (ICWR), 2023, pp. 247–252.

[10] P. Y. Prasad, A. S. Chowdary, C. Bavitha, E. Mounisha, and C. Reethika, “A comparison study of fraud detection in usage of credit cards using machine learning,” in 2023 7th International Conference on Trends in Electronics and Informatics (ICOEI), 2023, pp. 1204–1209.

[11] S. Mittal and S. Tyagi, “Performance evaluation of machine learning algorithms for credit card fraud detection,” in 2019 9th International Conference on Cloud Computing, Data Science Engineering (Confluence), 2019, pp. 320–324.

[12] D. Iyer, A. Mohanpurkar, S. Janardhan, D. Rathod, and A. Sardeshmukh, “Credit card fraud detection using hidden markov model,” in 2011 World Congress on Information and Communication Technologies, 2011, pp. 1062–1066.

[13] I. SADGALI, N. SAEL, and F. BENABBOU, “Fraud detection in credit card transaction using machine learning techniques,” in 2019 1st International Conference on Smart Systems and Data Science (ICSSD), 2019, pp. 1–4.

[14] B. Al Smadi and M. Min, “A critical review of credit card fraud detection techniques,” in 2020 11th IEEE Annual Ubiquitous Computing, Electronics Mobile Communication Conference (UEMCON), 2020, pp. 0732–0736.

[15] D. S. Nijwala, S. Maurya, M. P. Thapliyal, and R. Verma, “Extreme gradient boost classifier based credit card fraud detection model,” in 2023 International Conference on Device Intelligence, Computing and Communication Technologies, (DICCT), 2023, pp. 500–50.