

Credit Card Fraud Detection Using Machine Learning and Deploying Model in Streamlit App

A Thesis submitted to the Department of Computer Science and Engineering, Hajee Mohammad Danesh Science and Technology University in partial fulfillment of the requirements for the degree of B.Sc. (Engineering) in Computer Science and Engineering

Course Code: CSE 470

Course Title: Machine Learning and Pattern Recognition Sessional

By

Md. Mostafijur Rahman

Student ID: 1902073

Session: 2019

Al Mahmud Siam

Student ID: 1902062

Session: 2019



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

HAJEE MOHAMMAD DANESH SCIENCE AND TECHNOLOGY UNIVERSITY,
DINAJPUR-5200, BANGLADESH

September 2024

Table of Contents

Table of Contents.....	2
Abstract.....	3
Chapter 1.....	1
Introduction.....	1
1.1 Introduction.....	2
1.2 Research Motivation.....	3
1.3 Research Problem.....	3
1.4 Research Aims and Objectives.....	4
1.5 Contribution.....	5
1.6 Expected Outcomes.....	5
1.7 Thesis Organization.....	6
1.8 Conclusion.....	6
Chapter 2.....	8
Literature Review.....	8
2.1 Introduction.....	9
2.2 Challenges in Credit Card Fraud Detection.....	9
2.3 Removing Challenges in Credit Card Frauds Detection.....	10
2.4 Machine Learning Classifiers.....	11
2.9 Related Works.....	14
2.10 Conclusion.....	17
Chapter 3.....	18
Methodology.....	18
3.1 Dataset Description.....	19
3.2 Proposed framework.....	20
3.3 Algorithm on the proposed work.....	22
3.4 Features Selection Techniques.....	23
3.5 Deploying project to the Streamlit.....	26
Chapter 4.....	29
Result and discussion.....	29
4.1 Introduction.....	30
4.2 Model Performance Metrics.....	33
Chapter 5.....	40
Conclusion and Future Work.....	40
4.1 Conclusion and Future Work.....	41
Conclusion.....	41
Future Work.....	41

Abstract

At the current state of the fastest-growing world many industries like e-commerce, food and beverage, retail, transportation, healthcare, and so on are adopting credit card processing to exploit the facility of e-payment systems. Despite the many advantages of online payment, it has a higher risk of fraudulent transactions causing huge financial losses for both the customers and the companies. In this paper, we have focused on the comparative study of the performance of some developed methods using machine learning (ML) algorithms on an imbalanced dataset. As features of credit card frauds play an important role when machine learning is used for credit card fraud detection (CCFD) they must be chosen properly. In our work, after dividing the dataset into train and test samples, we apply the under sampling resampling techniques on the training dataset. In this paper we apply three machine learning algorithms and show the user interface using Streamlit. In user interface we select a specific model then measures the model classification report, confusion matrix and ROC curve

.Keywords – Under Sampling technique, Feature Selection, Machine learning algorithms, Imbalanced dataset, Credit Card Frauds detection (CCFD), Performance evaluation, Best model, Select methods, Streamlit

Chapter 1

Introduction

1.1 Introduction

The Internet has grown exponentially during the last decades. This has led to the proliferation and increases in the utilization of services like online bill payment, tap and pay, and e-commerce, among others [9]. Consequently, there has been an increase in fraudulent activity by criminals targeting credit card transactions. Generally speaking, credit card frauds are defined as criminal duplicity or international deception with the intention of making a profit—mostly financial. The alarming pace of rise in credit card frauds is a result of our rapidly rising reliance on online technologies [13]. To get around this obstacle, numerous models have been created and are being developed by many researchers which are mentioned in literature review sections having some limitations.

The aim of this study is to exhibit some detection strategies that are being developed to address this matter most effectively. In this work we assess an imbalanced dataset using some machine learning models and ascertaining the most congenial model for tracking out credit card frauds based on the number of predetermined performance criteria before selecting the important features by applying feature selection techniques to the dataset.

In most of the research findings discussed in literature review, the efficiency of models are varying while using the different machine learning algorithms. As we have used an imbalanced dataset from the European cardholders' September 2013 for balancing the dataset we have used a sampling approach called under sampling where the synthetic samples are generated for the minority class. We observed that the credit card transaction datasets are rarely available, highly imbalanced and skewed. Optimal feature (variables) selection for the models, suitable metric is the most important part to evaluate performance of techniques on skewed credit card fraud data [8]. To improve the model efficiency we used some feature selection techniques after normalization. After selection of important features we deployed the data to any of the machine learning algorithms. We use total three ML algorithms here for the purpose of classification of frauds and genuine transactions.

1.2 Research Motivation

Over the last decades of study, we have been following the news on information security and financial fraud as it is essential to all online and offline financial transaction systems. Although fraudulent transactions account for a relatively small percentage of most medium credit card transactions, as soon as a customer is unfortunate enough to have a credit card transaction, the loss of money to the business and a crisis of trust for the customer can ensue. Some reports show that Credit card fraud can easily accomplish their purpose. Large amounts of money can transact in a short period without any indication of risk and the owner's permission. Every fraudulent transaction can be legitimized by a fraudster's operation which makes fraud very challenging and difficult to detect [13]. As a result, we are sufficiently motivated to want to improve credit card fraud detection by training pass-through machine learning classification methods. The final purpose is to help this project to select a better model. The banks want to detect credit card transactions and quickly predict whether the trade is risky, regulators need to delay or hold the transaction, and the marketing needs to be blocked the next time the card is used a lot. We think we have ambitions to complete the fraud detection project. Besides, we hope we achieve an opportunity to realize the need for improved customer detection capabilities.

1.3 Research Problem

Imbalanced Datasets:

Credit card transactions are typically highly imbalanced, with a vast majority of transactions being legitimate. The challenge is developing effective models that can handle imbalanced datasets without compromising on fraud detection accuracy.

Data Privacy and Security:

Handling sensitive financial data requires robust measures for data privacy and security. Dataset can not give us the original data because of users' security in mind .Research is conducted to develop secure and privacy-preserving methods for handling credit card transaction data while maintaining the effectiveness of fraud detection.

Artificial Undersampling: As credit card fraud dataset is highly unbalanced so we use undersampling. The primary goal of this technique is to balance the class distribution by artificially decreasing the number of instances in the majority class. This undersampling is achieved by

introducing minority class a importance value

Understanding the designed Model results:

Here we apply the explainable AI to understand and interpret the decisions or predictions made by AI models. We use two methods here which are SHAP and LIME. We discussed in details in the one of the chapters.

Making the User Interface:

We use streamlit to show the user interface. It is perfect for data scientists and developers who want to showcase their work or create simple web applications for model testing, data exploration, and interactive presentations.

1.4 Research Aims and Objectives

The objective of this research is to evaluate the performance of the user's fraud detection model using different supervised machine algorithms to obtain a higher detection accuracy by comparing other methods, also to achieve this goal of the detection capability improvement, several objectives considered.

- Conduct regressions by collecting relevant research to identify problems with the current system in place, learn from the good experiences of other research, and also examine shortcomings
- Establish a pre-processed implementation to extract useful information and standardize the data by analyzing the existing dataset. Using Sampling Method (under sample) to solve the problem of highly unbalanced credit card data sets faced by the current fraud detection system.

- Evaluate the effectiveness of fraud detection by calculating the detection accuracy of different machine learning classification algorithms, logistic regression, decision tree and random forest and compare the results of this study in various aspects.
- After finding the best detection suitable for credit card spoofing detection, we can experimentally prove it by cross-validation and other methods.

1.5 Contribution

In this research paper we are trying to develop a model whose speed is very fast and its overall performance is very good. As this dataset contains a high range of samples so they need to develop a faster model as well as good performance. We take the dataset from Kaggle. The dataset is not balanced, which means some classes have a larger number of images and some have a very limited number of images. So, we try to use the undersampling Technique approach by which we can balance the dataset and prevent our model from overfitting. We also try to develop both Machine Learning and Deep Learning based models where we can provide a clear comparison between these two.

1.6 Expected Outcomes

The expected outcome of our thesis is to predict fraudulent transactions, minimizing financial losses for both individuals and financial institutions. Here are some general expected outcomes:

Fraud detection: The primary goal is to detect fraudulent transactions for both the cardholder and the financial institution. By accurately identifying suspicious activities, the system can intervene in real-time and block unauthorized transactions.

Reduce False Positives: While the system should effectively identify and block fraudulent transactions, minimizing false positives is also important. False positives occur when legitimate transactions are mistakenly flagged as fraudulent. Reducing false positives helps maintain a smooth and hassle-free experience for cardholders.

Data Security: Ensuring the security of credit card transaction data is an expected outcome. Fraud detection systems should be designed to handle sensitive information securely, complying with data protection and privacy regulations.

In summary, the expected outcomes of credit card fraud detection systems include identifying fraud, reducing financial losses, maintaining customer trust, complying with regulations, adapting to evolving threats, and improving the overall security and efficiency of financial transactions.

1.7 Thesis Organization

This thesis is structured into five chapters, each with a specific focus. A brief summary of these chapters is presented below.

Chapter 1 serves as an introduction to this thesis, highlighting its main objective, motivation, research problems, and contributions.

Chapter 2 provides an overview of important concepts and background knowledge relevant to the domain. It also shows different related works on the topic.

Chapter 3 outlines the complete methodology proposed in this thesis. It covers various aspects such as dataset description, preprocessing techniques, feature selection methods, performance measures, and the models employed to streamlit app.

Chapter 4 delves into the conducted experiments and presents detailed findings. It critically evaluates and discusses the outcomes of these experiments and visualization of result using streamlit app.

Chapter 5 concludes the work presented in this project and outlines potential directions for future research.

1.8 Conclusion

In conclusion, credit card fraud detection plays a pivotal role in safeguarding financial transactions and maintaining the integrity of the financial ecosystem. The continuous advancement of technology has not only facilitated legitimate financial transactions but has also given rise to increasingly sophisticated fraudulent activities. The implementation of robust fraud detection systems has become imperative to mitigate the risks associated with unauthorized transactions and protect both consumers and financial institutions.

First and foremost, the primary goal is to detect fraudulent transactions and minimize financial losses. By accurately distinguishing between legitimate and unauthorized activities, these systems contribute to the reduction of monetary losses associated with fraud. This, in turn, enhances the financial stability of both individual cardholders and the financial institutions that issue credit cards.

In conclusion, credit card fraud detection is not only about minimizing financial losses but also about creating a secure and trustworthy financial environment. It is a multifaceted approach that combines technology, analytics, and regulatory compliance to ensure the integrity of financial transactions in an ever-evolving digital landscape.

Chapter 2

Literature Review

2.1 Introduction

Credit card fraud is a pervasive and costly problem that affects financial institutions, businesses, and consumers worldwide. With the increasing reliance on electronic transactions and online shopping, the opportunities for fraudulent activities have grown exponentially. According to industry estimates, billions of dollars are lost each year due to credit card fraud, making it a significant concern for stakeholders across various sectors.

Detecting and preventing credit card fraud is a multifaceted challenge that requires a combination of advanced technologies, robust security measures, and effective risk management strategies. In recent years, there has been a surge in research efforts aimed at developing sophisticated fraud detection systems capable of identifying fraudulent transactions in real-time.

This literature review provides an overview of the current state-of-the-art techniques and methodologies employed in credit card fraud detection. It explores the evolution of fraud detection systems, from traditional rule-based approaches to more advanced machine learning and artificial intelligence (AI) algorithms. Additionally, this review discusses the key challenges and opportunities in the field and identifies emerging trends and future directions for research.

In conclusion, this literature review provides a comprehensive overview of the current landscape of credit card fraud detection. By synthesizing existing research and identifying gaps in knowledge, it aims to inform future research endeavors and contribute to the development of more robust and reliable fraud detection solutions.

2.2 Challenges in Credit Card Fraud Detection

Credit Card fraud detection has some of the challenges that complicate the fraud detection process.

1. Changing fraud patterns over time — This one is the toughest to address since the fraudsters are always in the lookout to find new and innovative ways to get around the systems to commit the act. Thus it becomes all-important for the deep learning models to be updated with

the evolved patterns to detect. This results in a decrease in the model's performance and efficiency. Thus the machine learning models need to keep updating or fail their objectives.

2. Class Imbalance — One major challenge is the highly imbalanced nature of the dataset, where the frequency of genuine cases far exceeds that of fraudulent cases. This class imbalance makes it difficult to identify positive examples (fraudulent cases) accurately, especially as more data is gathered and the proportion of positive examples decreases.

3. Model Interpretations — This limitation is associated with the concept of explainability since models typically give a score indicating whether a transaction is likely to be fraudulent or not — without explaining why.

4. Feature generation can be time-consuming — Subject matter experts can require long periods of time to generate a comprehensive feature set which slows down the fraud detection process.

2.3 Removing Challenges in Credit Card Frauds Detection

Addressing the challenges requires innovative approaches in algorithm development, data preprocessing, and model evaluation. Techniques such as learning, ensemble modeling, and feature selection techniques can help to mitigate the impact of changing fraud patterns and class imbalances. Additionally, efforts to enhance model interpretability, such as using explainable AI techniques, can improve trust in fraud detection systems. Moreover, leveraging advancements in computational efficiency and data processing can streamline feature generation tasks, enabling faster and more effective fraud detection. By tackling these challenges, organizations can build more robust and reliable credit card fraud detection systems to protect against financial fraud.

2.4 Machine Learning Classifiers

2.4.1 Logistic Regression (LR)

Logistic Regression is a widely used statistical method for binary classification problems, where the outcome variable is categorical and can take one of two possible values. Unlike linear regression, which is used for predicting continuous outcomes, logistic regression models the probability of a binary response using the logistic function. This transformation ensures that the output lies between 0 and 1, making it suitable for modeling probabilities. The logistic regression equation can be expressed as:

$$P(Y = 1|X) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_n X_n)}}$$

where $P(Y=1|X)$ represents the probability of the positive class, X_1, X_2, \dots, X_n are the independent variables, and $\beta_0, \beta_1, \dots, \beta_n$ are the coefficients to be estimated from the data. Logistic regression uses Maximum Likelihood Estimation (MLE) to estimate these coefficients, optimizing the fit of the model to the observed data. One of the strengths of logistic regression is its interpretability, as the estimated coefficients can be directly interpreted in terms of the odds ratio, giving insights into how each predictor influences the likelihood of the outcome. Moreover, logistic regression assumes a linear relationship between the predictors and the log-odds of the response, making it effective in many real-world applications such as medical diagnosis, credit scoring, and marketing. Despite its simplicity, logistic regression performs well when the classes are linearly separable and is often used as a baseline model for binary classification tasks. However, its performance can degrade in more complex settings, where non-linear decision boundaries are required, necessitating more sophisticated models like decision trees or neural networks.

2.4.1 Decision tree (DT)

Decision trees are a kind of supervised learning technique that is mostly employed in classification tasks having a predetermined goal variable [14]. Using this technique, a dataset is

divided into segments that are constructed with multiple branches with nodes that are connected by edges and create a tree-like structure with root nodes, internal nodes also called decision nodes and the leaf nodes. The leaf nodes indicate the final target variable. As it is a non-parametric supervised learning technique, it may effectively handle huge and intricate datasets without requiring any kind of complicated parametric framework. A DT contains the following types of nodes: root node, decision node, and leaf node. Fig. 2.1 shows how a decision tree looks like.

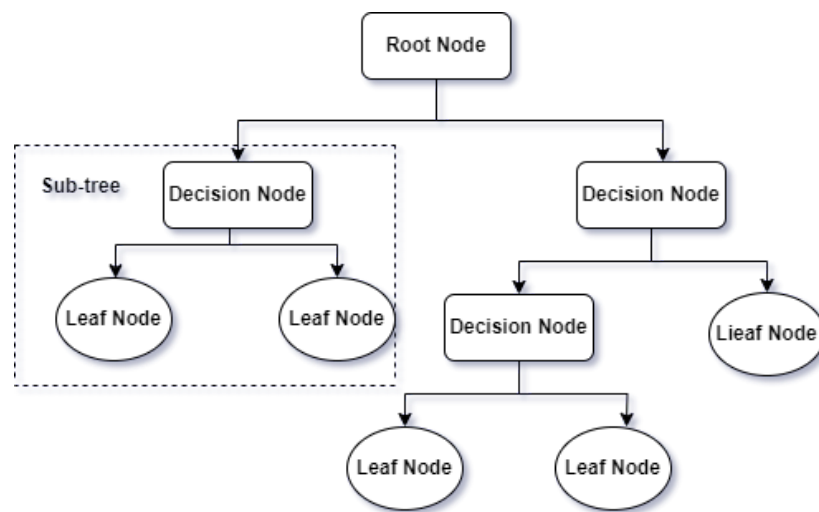


Fig. 2.1: Graphical presentation of Decision Tree

The underlying components of a decision tree are nodes and the branches and the required steps which are used in building a desired model are splitting, pruning and stopping. The branches which help to create the tree hierarchy represent the possible outcomes. The path from root nodes through decision nodes to leaf nodes indicates the decision rules for classification also called if-then rules [15]. To construct relatively pure nodes, a node is split into several sub-nodes, a process known as node splitting or simply splitting. This may be accomplished in a few different ways by determining the optimal split for a node. It uses entropy, Gini index Information Gain, and as a metric while splitting. These metrics are used for different target variables [16].

Entropy is a metric for measuring the impurity for a dataset that means how much variance the data has or randomness in data. The expressions used for computing entropy are

$$H(X) = - \sum_{i=0}^n P(x_i) \log \log P(x_i) \quad (2.1)$$

$$E(X) = -(Fraud) \log_2 p(Fraud) - p(Not\ F\ aud) \log_2 p(Not\ F\ raud) \quad [16] \quad (2.2)$$

The *information gain* is based on the reduction in entropy after a dataset is split on a specific feature. In splitting our main goal is to make the largest information gain with the lowest entropy. The formula used for calculating IG.

$$(X, Y) = (X) - (X|Y) \quad (4) \quad (2.3)$$

$$G(X, Y) = -p(F\ raud) \log_2 p(F\ raud) - p(Not\ F\ raud) \log_2 p(Not\ F\ raud) - \sum |Sv\ S| \text{entropy}(Sv) \quad [16] \quad (2.4)$$

The metric of splitting a decision tree is the *Gini Index/ Gini impurity*. It is a measure of how mixed or impure a dataset is. The Gini impurity ranges between 0 and 1. The formula for calculating the Gini Index is

$$\text{Gini} = 1 - \sum_{i=1}^n (x_i)^2 \quad (2.5)$$

2.4.2 Random Forest (RT)

Random Forest is a machine learning algorithm solving classification and regression-type problems consisting of multiple decision trees (DTs). It overcomes the limitation of high variance of DTs by applying row sampling with replacement and features sampling techniques randomly to the dataset to make two or more decision trees. After that, the DTs are aggregated and then applied voting techniques (taking a majority vote) to the test data to classify datasets. The figure shows the concept of the RT algorithm. The RT method resists overfitting and offers a reliable estimate of the generalization error [8]. It provides the ability for ensemble learning, which is the act of merging several classifiers to solve a challenging issue and enhance the model's performance [14]. The greater number of trees in the forest leads to higher accuracy prevents the problem of overfitting and takes less training time as compared to other algorithms

[30]. It operates effectively even with a big dataset and predicts output with high accuracy. In situations where a significant amount of data is absent, it can nevertheless retain accuracy.

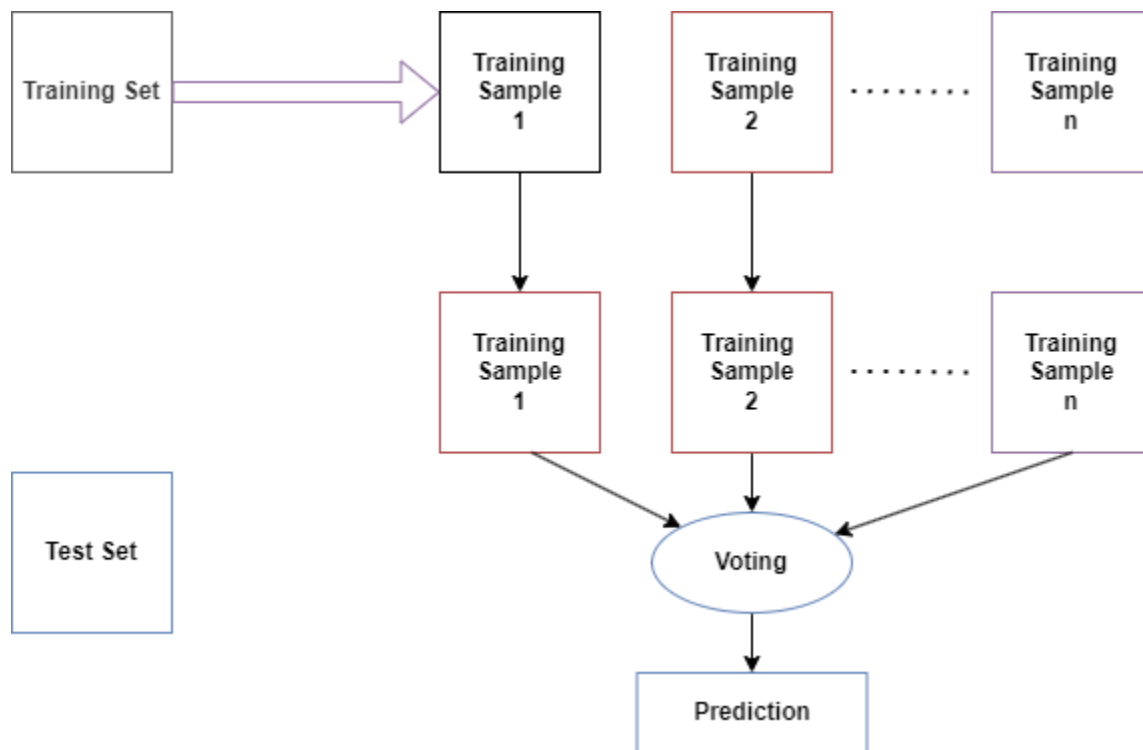


Fig. 2.2: Working mechanism of Random forest algorithm

2.9 Related Works

The paper [11] intends to illustrate the modeling of a data set using machine learning with Credit Card Fraud Detection. It identifies the fraudulent transactions with the highest percentage and minimizes the incorrect fraud classifications. It focuses on the analysis and pre-processing of PCA-transformed data and applies two anomaly detection algorithms. In this paper, it is possible to incorporate several algorithms as modules and combine their outputs to improve the product's accuracy.

Some supervised learning algorithms are applied to an imbalanced dataset to make a comparison of these algorithms and to differentiate between legitimate and fraudulent transactions in the study [10]. In this paper for performance evaluation sensitivity, precision, and time are used to

conclude. Here applying resampling techniques to the imbalanced datasets to deduce the imbalance ratio so that it is possible to apply and improve the accuracy of different algorithms.

As seen in paper [1], the comparative study of several techniques used in fraud detection is based on the design criterion. In this paper nine different algorithms are applied to the dataset and accuracy, speed, and cost metrics are used for performance evaluation. These algorithms can be used to build classifiers using ensemble or meta-learning techniques or hybrid approaches can be applied to achieve higher accuracy and variable classification cost in several classification problems.

In this paper [9], a machine learning (ML) based credit card fraud detection engine with feature selection based on genetic algorithms (GAs) is proposed. The suggested detection engine employs the following machine learning classifiers after selecting the optimal features: Decision Tree (DT), Random Forest (RF), Logistic Regression (LR), Artificial Neural Network (ANN), and Naive Bayes (NB). The suggested credit card fraud detection engine is assessed using a dataset created from European cardholders to verify the performance. Other feature selection techniques can be used to see the better model on the same dataset.

The goal of the study paper [2] is to distinguish between fraudulent and non-fraudulent transactions using a variety of machine learning methods, including support vector machine (SVM), k-nearest neighbor (kNN), and artificial neural network (ANN). In this case, the experiment of these three algorithms are compared and performance is tested to determine accuracy. It is concluded that artificial neural networks perform better than systems built with support vector machines and k-nearest neighbor algorithms in terms of prediction. More accuracy should be achieved in SVM and kNN by applying data pre-processing, normalization, and under-sampling in datasets.

In this paper [30] four machine-learning algorithms are applied to an imbalanced dataset. To detect fraud transactions and to find the best accuracy which are K-nearest neighbor, Random Forest, Logistic Regression, and Decision Tree with various techniques and measure the performance of techniques based on accuracy and precision with the highest value

The two most important classification models are discussed in the paper [24] which are decision tree(DT) and Support Vector Machine(SVM) by comparing their performance with real datasets. When the size of the datasets increased the SVM model outperformed the decision tree but the number of frauds less the DT. Other classification models can be perfectly applied with the same datasets and make better comparisons with other performance metrics.

In another comparative study [9], a machine learning (ML) based credit card fraud detection engine with feature selection based on genetic algorithms (GAs) is proposed. Here, the suggested detection engine employs the following machine learning classifiers after selecting the optimum features: Decision Tree (DT), Random Forest (RF), Logistic Regression (LR) and used the European cardholders datasets. Another different dataset can be applied to the system to check and improve the performance.

A hybrid solution is proposed using the neural network (ANN) in a federated learning framework in the paper [4]. It produces higher accuracy while ensuring privacy while using real-life datasets.

In the work [25], the use of two machine learning approaches with real-world financial datasets is demonstrated with noteworthy results. The methods are Bayesian Belief Network and Artificial Neural Network (ANN). Pruning methods can be used in ANN to remove connections and perceptron that aren't used in the training phase of the model, increasing its efficiency.

To achieve a better result, imbalanced or skewed data is preprocessed with the re-sampling (over-sampling or under-sampling) technique for better results and, then applies three machine learning algorithms namely: logistic regression, Naïve Bayes and K-nearest neighbor [28]. The performance of these algorithms is recorded with their comparative analysis based on accuracy, sensitivity, specificity, precision, F-measure, and area under the curve.

Various techniques of credit card fraud detection provide enhanced protection for credit card systems against a variety of frauds. Some techniques are decision tree, hidden markov model (HMM), logistic regression, and fuzzy logic-based system. These techniques both detect and prevent credit card fraudulent transactions by ensuring high security for the cardholders and protecting their personal information [5].

Representation of the comprehensive review of various methods used to detect credit card fraud such as the Markov Model, Decision Trees, Logistic Regression, Support Vector Machines (SVM), Genetic algorithm, Neural Networks(ANN, CNN, RNN) , Random Forests, Bayesian Belief Network are discussed in paper [18]. It indicates the pros and cons of every method used in the system.

2.10 Conclusion

In conclusion, credit card fraud detection presents a complex landscape marked by challenges such as evolving fraud patterns, class imbalances in datasets, limitations in model interpretability, and the time-consuming nature of feature generation. Overcoming these obstacles requires a comprehensive approach that integrates innovative algorithm development, robust data preprocessing techniques, and effective model evaluation strategies. By embracing advancements such as adaptive learning, ensemble modeling, explainable AI, and automation, stakeholders can enhance the accuracy, efficiency, and interpretability of fraud detection systems.

Chapter 3

Methodology

3.1 Dataset Description

The dataset that is used with this proposed approach is a real-world dataset downloaded from Kaggle. It was created from European cardholders' September 2013 MasterCard transactions. A total of 284,807 entries, or transactions, have been recorded in two days. Out of 284,807 transactions with 31 features namely- "Time," "V1" to "V28," "Amount," and "Class"—there are 492 cases of fraud and the rest were legitimate. Considering the number of fraudulent transactions, we can see that this dataset is highly imbalanced which means disparity occurs in the dependent variables [10], where only 0.173% of transactions are labeled as frauds. To solve the imbalanced dataset issue an oversampling technique is used in the proposed system. The features from "V1" to "V28," are transformed into numerical values using a PCA- the dimensionality reduction transformation to protect user identities and sensitive features. After exploring the dataset it is seen that there are no missing values in the dataset so here there is no need of preprocessing steps. There may be unwanted features that are not related to the independent feature i.e. "Class" feature in the dataset. To check this we use an advantageous technique called correlation technique. The figure shows the heatmap of the correlation matrix where it is obvious that there is no need to remove any features because all of the features are related to the "Class" feature [14]. Thus we don't need any preprocessing to the dataset.

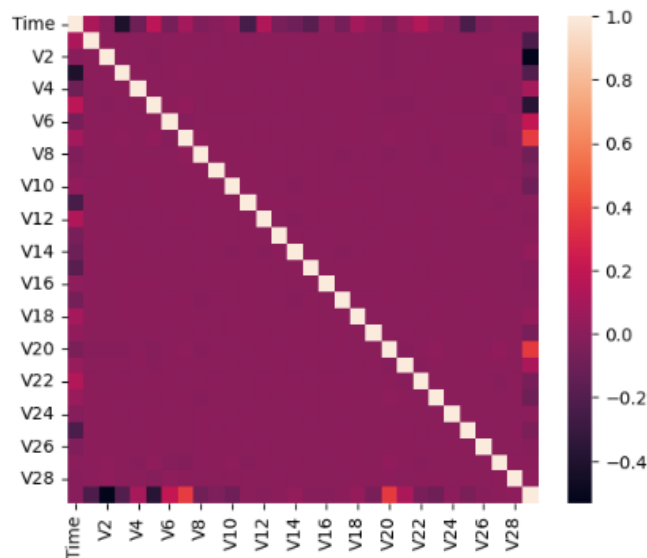


Fig. 3.1: Correlation matrix with heatmap

3.1.1 Sampling Techniques

Compared to legitimate transactions, there are much fewer instances of fraudulent transactions in our dataset. Frequently used methods for adjusting the class distribution include under sampling the majority class, oversampling the minority class, or a combination of those two [30]. As in our used dataset the fraudulent transactions are much smaller than the genuine transactions to overcome this, we conducted one under-sampling technique [12]. It is used here in which majority occurrences are decreased and artificial samples are produced for the minority class. The technique which improves the random under-sampling and is also used to overcome the overfitting problem. The figure shows the class distribution before and after applying the sampling technique.

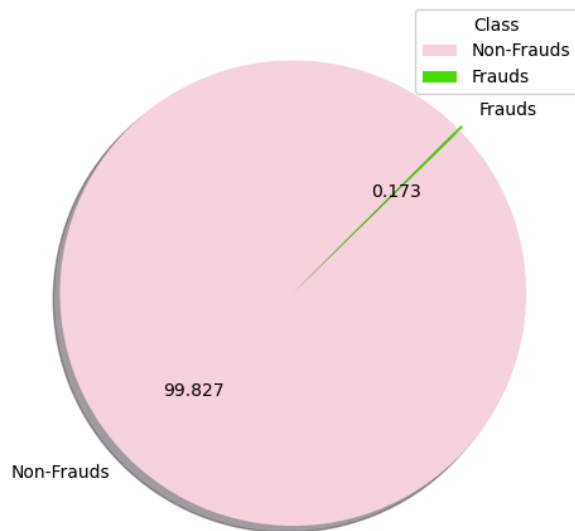


Fig. 3.2 (a): Before sampling

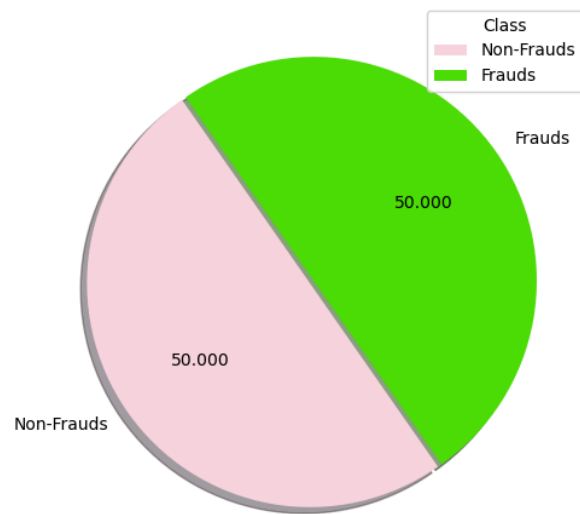


Fig. 3.2 (b): After sampling

Fig.3.2: Class Distribution before Sampling and after Sampling

3.2 Proposed framework

Fig. 3.3.1 shows the architecture of the suggested methodology. One of the sampling techniques such as the SMOTE technique is employed for sampling the downloaded dataset. After the sampling is done the dataset is split into two halves, one training set and the other testing set. Then one of the previously specified feature selection techniques is applied to the training dataset to make the dataset more presentable for performing any machine learning algorithms to the dataset for better performance in the model. After the model is developed the evaluation method is done using the testing dataset. For performance evaluation, some metrics are used that are shown in the previous section and by comparing the models performance we suggest the best one.

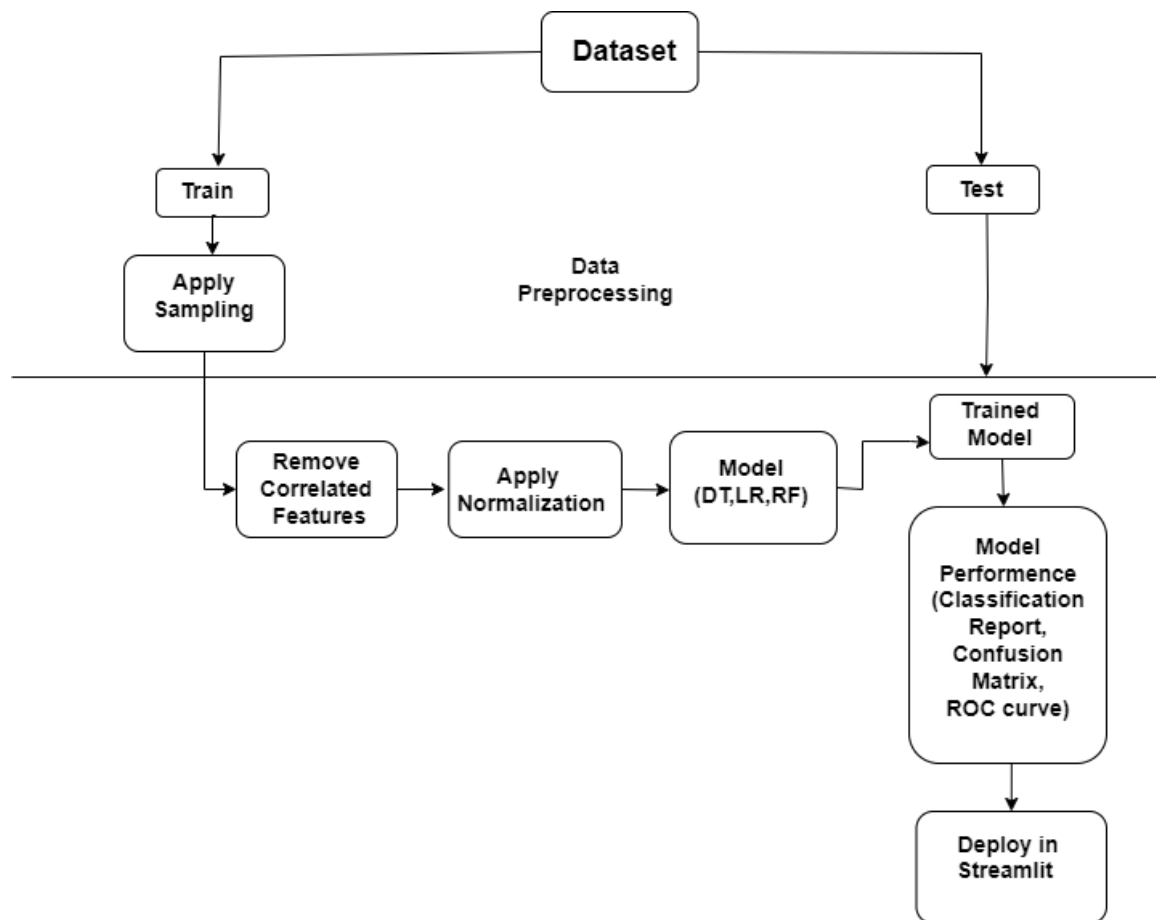


Fig. 3.3: The proposed framework

3.3 Algorithm on the proposed work

Algorithm in Mathematical Form

Input:

Dataset $D = \{(X, Y)\}$, where $X \in \mathbb{R}^{n \times d}$ (feature matrix) and $Y \in \{0, 1\}^n$ (binary labels)

Models = LR, DT, RF

Output:

Model Accuracy

Data Preprocessing:

Split dataset D into training set $(X_{\text{train}}, Y_{\text{train}})$ and testing set $(X_{\text{test}}, Y_{\text{test}})$. Select dependent (target) feature Y and independent features X .

Apply Under Sampling to Training Data:

Generate synthetic samples for the minority class using under sampling:

$$(X_{\text{SMOTE}}, Y_{\text{SMOTE}}) = \text{UNDER_SAMPLE}(X_{\text{train}}, Y_{\text{train}})$$

Remove Correlated Features:

Compute the Pearson correlation matrix $C \in \mathbb{R}^{d \times d}$:

$$C_{ij} = \frac{\text{Cov}(X_i, X_j)}{\sigma(X_i) \cdot \sigma(X_j)}$$

Remove features where $|C_{ij}| > \epsilon$ (threshold for correlation).

Normalize Features:

Normalize the selected features $X_{\text{normalized}}$ using min-max scaling or z-score normalization:

$$X_{\text{normalized}} = \frac{X_{\text{selected}} - \min(X_{\text{selected}})}{\max(X_{\text{selected}}) - \min(X_{\text{selected}})}$$

Train Multiple Models:

For each model $M \in \text{Models}$, train the model on the normalized features:

$$M_{\text{trained}} = M.\text{fit}(X_{\text{normalized}}, Y_{\text{SMOTE}})$$

Evaluate model performance using metrics such as accuracy, precision, recall, F1-score, and AUC:

$$\text{Performance}(M_{\text{trained}}) = \{\text{Accuracy, Precision, Recall, F1, ROC}\}$$

Model Performance Evaluation:

Evaluate the performance of the ensemble model on the test set:

$$\text{Performance}(y_{\text{ensemble}}, Y_{\text{test}}) = \{\text{Accuracy, Precision, Recall, F1, ROC}\}$$

Select Best Model and Method:

Select the model with the best performance based on the evaluation metrics:

Deploy Model on Streamlit APP:

We use streamlit python library packages to deploy the model.

3.4 Features Selection Techniques

3.4.1 Correlation

The Under sampling can lead to higher correlation between features due to several factors. By generating samples through interpolation between existing minority class instances, it creates new data points that are more similar to each other, thereby increasing feature correlation. This process reduces the density of the majority class in the feature space, which can result in stronger linear relationships among features. Additionally, the samples often exhibit reduced variability compared to the original sparse minority class samples, further amplifying correlations. The clustering effect that arises from closely packed synthetic samples can also contribute to this phenomenon. Consequently, while under sampling effectively addresses class imbalance, it is crucial to monitor and manage the resulting feature correlations, as they may impact the

performance of certain machine learning algorithms sensitive to multicollinearity. Applying feature selection or dimensionality reduction techniques post may be necessary to mitigate any adverse effects on model performance.

The main goal of correlation is to reduce the amount of features of the majority class. There used a fixed value θ which is used as the threshold defining the separation between evaluating features individually and in pairs. When 2 features correlation coefficient value are greater than the threshold value than they dependent with one another. Then we may remove any of them. On the other hand when 2 features correlated values are greater than the threshold value then they are evaluated as independent [33]. There are various methods to calculate correlation between different variables like Pearson correlation, Kendall Tau correlation, Spearman rank correlation. In this paper we use Pearson correlation method.

$$\text{Formula of correlation, } \text{Corr}(x,y) = \frac{\text{cov}(x,y)}{\sqrt{\text{var}(x)} \cdot \sqrt{\text{var}(y)}} \quad (3.1)$$

Where $\text{cov}(x,y)$ is covariance between x and y means how x and y are related to one another.

$$\text{cov}(x,y) = \frac{\sum (x_i - \bar{x}) \cdot (y_i - \bar{y})}{N} \quad (3.2)$$

where N is total no. of samples in the dataset

$$\text{var}(x) = \frac{\sum (x_i - \bar{x})^2}{N} \quad (3.3)$$

where \bar{x} is the mean of input data and x_i is each single data

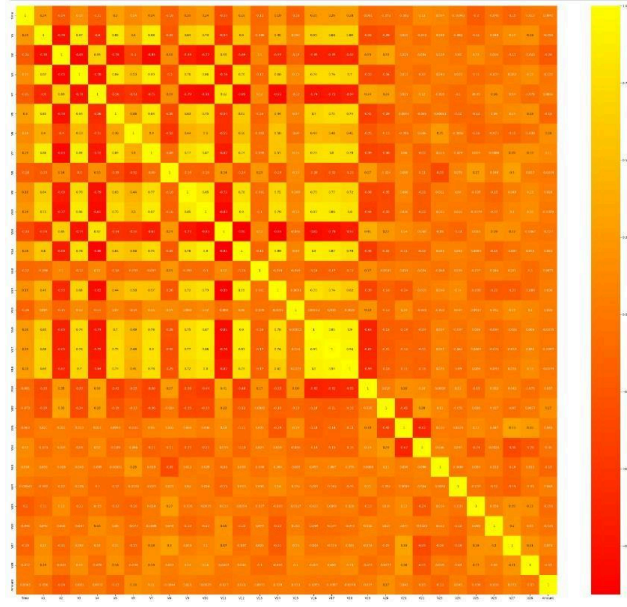


Fig. 3.5: Correlation between all train dataset

In this paper here we use the threshold value (θ) is 0.95. In Fig. 3.4.2.1. It shows the correlation of all 30 train features between each other. V17 crosses the threshold value that we set 0.95 is correlated with V16. Remove these two features from our train dataset. Now our dataset has 28 features available to train the model.

3.4.2 Normalization

Different features' value ranges and dimensions are different from each other, so those features can't be compared with each other and can't directly be used as input for the model [34].

In this case an essential method is required that converts the input data in a standard format. One of the most used worldwide techniques is normalization. In this method the input data is converted in the range between 0 and 1 and apply all of the input data before they are used for train the model.

In this paper for processing the data we used min-max normalization.

$$\text{Formula of normalization, } Y_{norm} = \frac{Y - Y_{min}}{Y_{max} - Y_{min}} \quad (3.4)$$

Where Y is the input data, Y_{min} is the minimum input value and Y_{max} is the maximum input value and Y_{norm} is the final output of converting input data to normal range [0,1].

3.5 Deploying project to the Streamlit

Streamlit is an open-source Python library that allows you to create interactive web applications for machine learning and data science projects with minimal effort. It is designed for rapid prototyping and data exploration, enabling data scientists and engineers to turn their data analysis scripts into shareable, web-based applications in a few simple steps. Here's an overview of its features and key aspects:

Key Features of Streamlit:

1. Easy to Use:

- Streamlit is simple to set up and use. You can turn a Python script into an interactive app by adding just a few lines of code.
- No need for front-end development experience or knowledge of JavaScript, HTML, or CSS.

2. Widgets and Interactivity:

- Streamlit provides built-in widgets such as sliders, buttons, checkboxes, and text inputs to make your app interactive. These widgets allow users to input data and parameters dynamically, and Streamlit automatically reruns your Python script whenever a widget changes.

3. Data Visualization:

- It supports a wide range of popular visualization libraries like **Matplotlib**, **Seaborn**, **Plotly**, **Altair**, and **Bokeh**. This makes it easy to display interactive charts and graphs directly in the app.
- You can create real-time data visualizations that update based on user inputs.

4. Support for Machine Learning Models:

- Streamlit can serve machine learning models directly in an app. You can input data, run predictions, and display results in real-time.
- It works seamlessly with libraries like **scikit-learn**, **TensorFlow**, **PyTorch**, and others.

5. Layout Customization:

- Streamlit provides functions to structure the layout of your app using columns, containers, and other layout options, making it easy to build dashboards.

6. **Markdown and Text Support:**

- You can easily include text, markdown, and LaTeX in your apps, allowing you to explain your data, models, and visualizations effectively.

7. **Deployment:**

- Streamlit apps can be deployed with ease using **Streamlit Cloud** (formerly Streamlit Sharing) or other platforms like **Heroku**, **AWS**, and **Google Cloud**.
- It allows you to share your app with others quickly without worrying about complicated server setup or configuration.

Streamlit Workflow Example:

Install Streamlit:

```
pip install streamlit
```

Create a simple app (e.g., `app.py`):

```
import streamlit as st

st.title("Simple Streamlit App")

st.write("This is an example of a Streamlit app.")

# Input widget

name = st.text_input("Enter your name:")

if st.button("Submit"):

    st.write(f"Hello, {name}!")
```

Run the app:

```
streamlit run app.py
```

1. This will launch a local web server, and you can view the app in your browser at <http://localhost:8501>.

Advantages:

- **Quick Prototyping:** Ideal for rapidly turning ideas into interactive apps.
- **Python-Only:** You don't need to learn any web development languages.
- **Interactivity:** Widgets and real-time updates make apps responsive.
- **Open-Source & Free:** Streamlit is free to use, and its code is open-source.

Disadvantages:

- **Limited UI Customization:** Streamlit is not as flexible as full web frameworks (e.g., Django, Flask) when it comes to customizing the front-end design and layout.
- **Performance Constraints:** For very large datasets or complex computations, Streamlit might not be as fast as a fully optimized web app.

Finally Streamlit is perfect for data scientists and developers who want to showcase their work or create simple web applications for model testing, data exploration, and interactive presentations.

Chapter 4

Result and discussion

4.1 Introduction

In this section, we present and analyze the outcomes of the credit card fraud detection model, followed by a demonstration of the user interface designed using Streamlit. The models that we design successfully identified fraudulent transactions with high precision, but there was a trade-off with recall, especially when tuning the model for better precision. The Streamlit interface provides an intuitive and interactive platform for end-users, including financial analysts and risk management teams, to detect potential fraud in real-time. Future improvements could include the integration of real-time detection systems and further refinement of the model using newer datasets or advanced algorithms.

4.1.1 Model Performance comparison

The credit card fraud detection model was developed using machine learning algorithms on a dataset containing historical transactions. Given the imbalanced nature of the data, where fraudulent transactions are significantly less frequent than non-fraudulent ones, performance metrics such as **precision**, **recall**, **F1-score**, and **ROC-AUC** were used to evaluate the model. These metrics provide insight into how well the model can distinguish between legitimate and fraudulent transactions.

- **Accuracy:** While accuracy is a basic measure, it can be misleading due to the data imbalance. For example, predicting all transactions as non-fraudulent would yield a high accuracy, but it would fail to identify fraud cases.
- **Precision and Recall:** Precision is crucial for fraud detection because it measures how many of the predicted frauds were actually fraudulent. Recall measures how well the model captures actual fraudulent cases.
- **F1-score:** A balance between precision and recall, the F1-score provides a single metric that accounts for both false positives and false negatives.

4.1.1 Streamlit App Interface for Credit Card Fraud Detection

To make the model accessible and easy to use, a Streamlit-based user interface was developed. This interface allows users to:

1. **View predictions:** After processing the data, the interface displays whether each transaction is predicted to be fraudulent or not. It highlights key metrics such as confidence scores for each prediction.
2. **Visualize results:** The UI includes several visualizations, such as histograms showing the distribution of predicted fraud cases and pie charts displaying the proportion of fraudulent versus legitimate transactions.

1. App Layout

The Streamlit app is structured with a clean and user-friendly layout that guides users through the process of uploading transaction data, processing it with the fraud detection algorithm, and viewing the results.

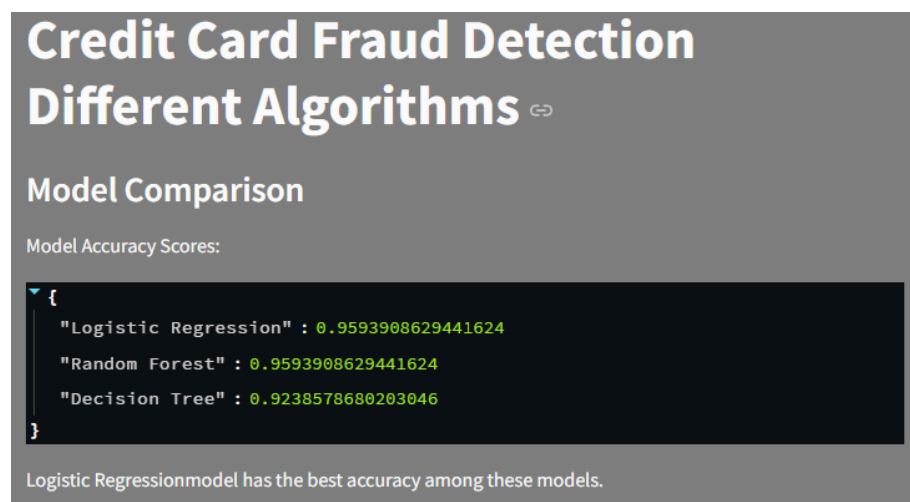


Fig. 4.1 Model Performance Accuracy Comparison

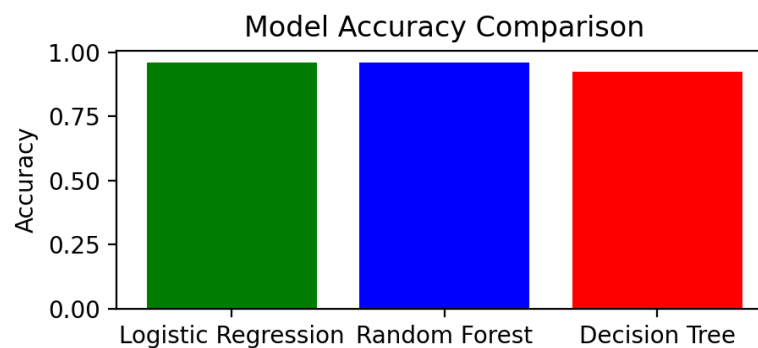


Fig. 4.2 Model Performance comparison

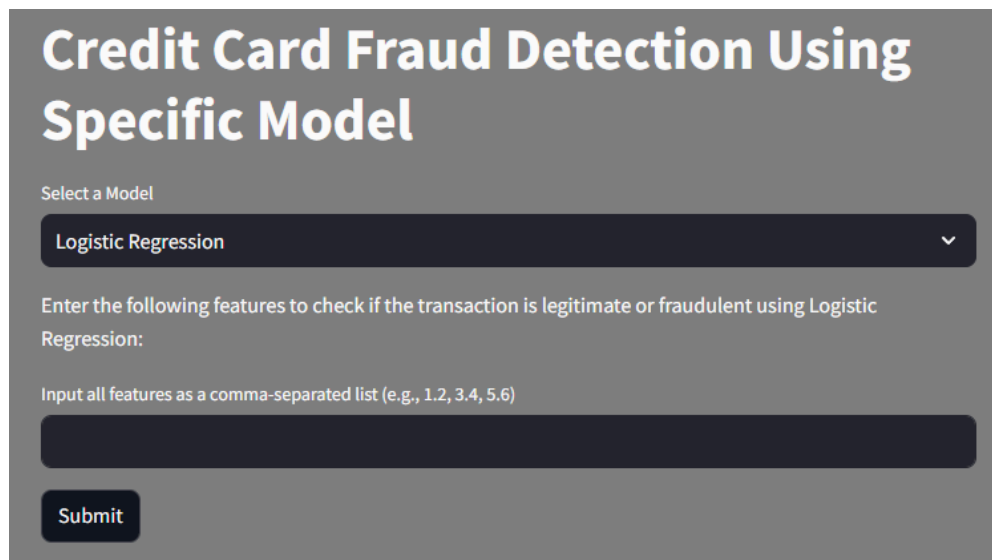
2. Input Section

- **File Upload Widget:**

- A file uploader allows users to upload their transaction data in CSV format. The app can include a sample dataset link for users to test the interface.
- The interface can validate the file type and provide feedback if an unsupported format is uploaded.

- **Input Preview:**

- After uploading, the app can display a preview of the uploaded data, showing the first few rows in a table format. This step allows users to verify that the correct data has been uploaded.



Credit Card Fraud Detection Using Specific Model

Select a Model

Logistic Regression

Enter the following features to check if the transaction is legitimate or fraudulent using Logistic Regression:

Input all features as a comma-separated list (e.g., 1.2, 3.4, 5.6)

Submit

Fig. 4.3 Input Field

3. Model Selection (Optional)

If your app supports multiple algorithms (e.g., Logistic Regression, Random Forest, etc.), include a dropdown menu where users can select their preferred model for fraud detection.

4. Prediction Button

- A button labeled “Submit” initiates the prediction process. When clicked, the app processes the input data using the selected algorithm.

6. Output Section

- **Prediction Results:**
 - After processing, the app displays the results in a new table, showing each transaction along with a prediction label (e.g., “Fraud” or “Not Fraud”) and a confidence score or probability for each prediction.
 - This section can also highlight transactions flagged as fraudulent for easy identification.
- **Visualizations:**
 - Include visual components like:
 - **Classification Report**
 - **Confusion Matrix**
 - **ROC curve**

4.2 Model Performance Metrics

Display key performance metrics (like precision, recall, and F1-score) used by the algorithm. This provides users with insight into the reliability of the predictions.

4.2.1 Classification Report

We employ a variety of parameters to assess the performance of a specific model. The trained dataset is used to apply the models, and then the outputs obtained with the use of each model are compared systematically to those produced by the other models [10]. A determination is made regarding the most appropriate model for the dataset or problem type based on these comparisons. In this work, we compare the several models under use using the four elementary matrices: Accuracy, precision, recall, F1-score. All evaluation metrics used in the proposed approach depend on a confusion matrix in one way or another [28]. A confusion matrix is a performance evaluation tool in machine learning, representing the accuracy of a classification model. It displays the number of true positives, true negatives, false positives, and false

negatives. A confusion matrix, which shows a classification model's accuracy, is a machine learning performance evaluation tool in which the number of false positives (FP), false negatives (FN), true positives (TP), and true negatives (TN) is shown.

Accuracy is the proportion of correctly anticipated results. The overall accuracy of the classifier can be calculated by adding the number of true positives and true negatives and dividing by the total number of predictions [26]. It is also known as the error rate and is calculated by the following formula:

$$Accuracy = \frac{TP+TN}{TP+FN+TN+FP} \quad (1)$$

Precision provides a measure of the percentage of positive predictions that are true positives. It shows how well a model predicts a certain class of interest [10]. In essence, precision is calculated by the ratio of the total number of true positives to the total number of positive predictions which is the addition of the number of true positives and the number of false positives. It is also called a positive predicted value. It is depicted mathematically in the following formula:

$$Precision = Positive Predicted Value = \frac{TP}{TP+FP} \quad (2)$$

Recall also known as True Positive Rate (TPR) and Sensitivity, is one of the most important evaluation metrics used in detecting fraudulent credit card transactions [28]. It is calculated as the ratio between the number of true positives to the total number of positive samples. It is shown in equation (3):

$$Recall = Sensitivity = TPR = \frac{True Positives}{Total Positives} = \frac{TP}{TP+FN} \quad (3)$$

F1 Score is used to show the accuracy of the test which means that it gives the accuracy of experiments performed [33]. It uses both precision and recall to compute the value. It is calculated by the following equation:

$$F1 = 2 \times \frac{Precision * Recall}{Precision + Recall} \quad (4)$$

Classification Report for LR:

	precision	recall	f1-score	support
0	0.925234	1.000000	0.961165	99.000000
1	1.000000	0.918367	0.957447	98.000000
accuracy	0.959391	0.959391	0.959391	0.959391
macro avg	0.962617	0.959184	0.959306	197.000000
weighted avg	0.962427	0.959391	0.959315	197.000000

Fig. 4.4 Classification Report for Logistic Regression

Classification Report for Random Forest:

	precision	recall	f1-score	support
0	0.950495	0.969697	0.960000	99.000000
1	0.968750	0.948980	0.958763	98.000000
accuracy	0.959391	0.959391	0.959391	0.959391
macro avg	0.959623	0.959338	0.959381	197.000000
weighted avg	0.959576	0.959391	0.959385	197.000000

Fig. 4.5 Classification Report for

Classification Report for Decision Tree:

	precision	recall	f1-score	support
0	0.945652	0.878788	0.910995	99.000000
1	0.885714	0.948980	0.916256	98.000000
accuracy	0.913706	0.913706	0.913706	0.913706
macro avg	0.915683	0.913884	0.913625	197.000000
weighted avg	0.915835	0.913706	0.913612	197.000000

Fig. 4.6 Classification Report for Decision Tree

4.2.2 Confusion Matrix

The **Confusion Matrix** is an essential tool for evaluating the performance of classification models. It provides a detailed breakdown of how well a model is able to classify instances of different categories or classes. In its basic form, it is a table that contrasts the actual labels of the dataset (true values) with the predicted labels generated by the model. This matrix is especially

valuable in binary classification tasks, but it can be extended to multiclass classification scenarios as well.

The Confusion Matrix consists of four key components: **True Positives (TP)**, **False Positives (FP)**, **True Negatives (TN)**, and **False Negatives (FN)**. Each of these measures offers insights into different types of errors and successes made by the model. Specifically:

- **True Positives (TP)**: Instances where the model correctly predicted the positive class.
- **False Positives (FP)**: Instances where the model incorrectly predicted the positive class, but the actual class was negative (also known as a Type I error).
- **True Negatives (TN)**: Instances where the model correctly predicted the negative class.
- **False Negatives (FN)**: Instances where the model incorrectly predicted the negative class, but the actual class was positive (also known as a Type II error).

These components help in calculating several important performance metrics, such as **accuracy**, **precision**, **recall**, **specificity**, and **F1 score**, all of which provide deeper insights into the model's ability to generalize across different datasets. For instance, **accuracy** reflects the overall correctness of the model by calculating the proportion of true results (both TP and TN) out of the total number of instances. **Precision** focuses on the model's ability to correctly identify positive results, while **recall** emphasizes its capacity to detect all actual positives. The **F1 score** harmonizes precision and recall into a single metric, particularly useful when the classes are imbalanced.

The **Confusion Matrix** is especially helpful in situations where the dataset is imbalanced (i.e., one class significantly outnumbers another). In such cases, relying solely on accuracy can be misleading. For example, in a fraud detection scenario where fraudulent transactions are rare, a model might achieve high accuracy by predicting most transactions as legitimate. However, the Confusion Matrix, along with metrics like precision and recall, would reveal the model's weaknesses in detecting fraud.

Confusion Matrix for Logistic Regression:

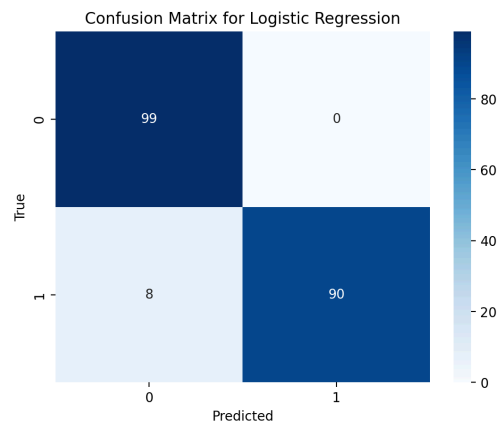


Fig. 4.7 Confusion Matrix for Logistic Regression

Confusion Matrix for Random Forest:

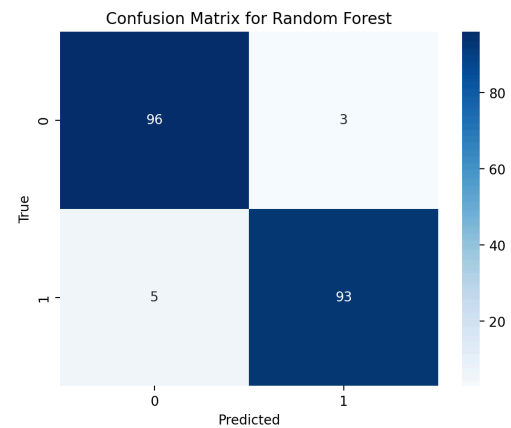


Fig. 4.8 Confusion Matrix for Random Forest

Confusion Matrix for Decision Tree:

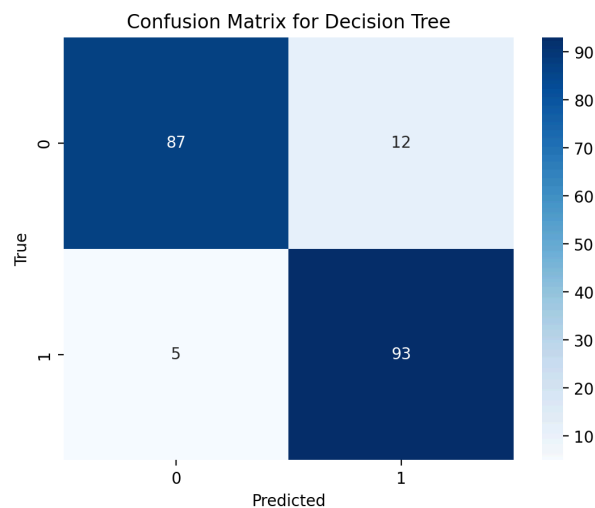


Fig. 4.9 Confusion Matrix for Random Forest

4.2.3 ROC Curve

The **ROC Curve** (Receiver Operating Characteristic Curve) is a graphical representation used to evaluate the performance of a binary classification model by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings.

- **True Positive Rate (TPR)**, also called sensitivity or recall, measures the proportion of actual positives that are correctly identified by the model. It is defined as:

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

Here, TP (True Positives) are correctly predicted positive cases and FP (False Positives) are actual positive cases that were incorrectly predicted as negative.

- **False Positive Rate (FPR)** measures the proportion of actual negatives that are incorrectly identified as positive by the model. It is defined as:

$$FPR = \frac{FP}{FP+TN}$$

- FPF (False Positives) are actual negative cases that were incorrectly predicted as positive.
- TNP (True Negatives) are correctly predicted negative cases.

The ROC curve is generated by plotting the **TPR** against the **FPR** at different classification thresholds. Each point on the curve represents a different threshold for deciding whether a given instance is classified as positive or negative. As the threshold changes, the balance between true positives and false positives shifts.

ROC Curve for Logistic Regression:

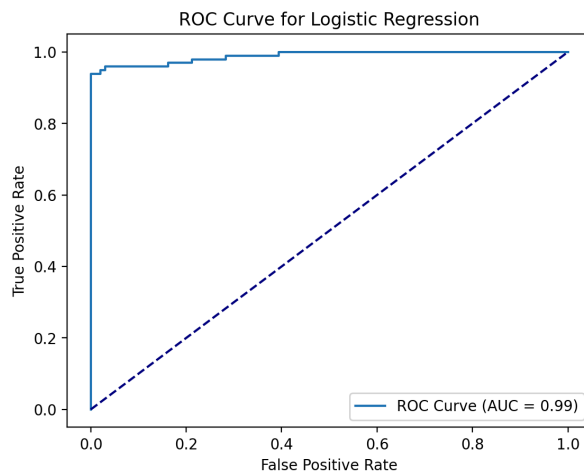


Fig. 4.10 ROC Curve for Logistic Regression

ROC Curve for Random Forest:

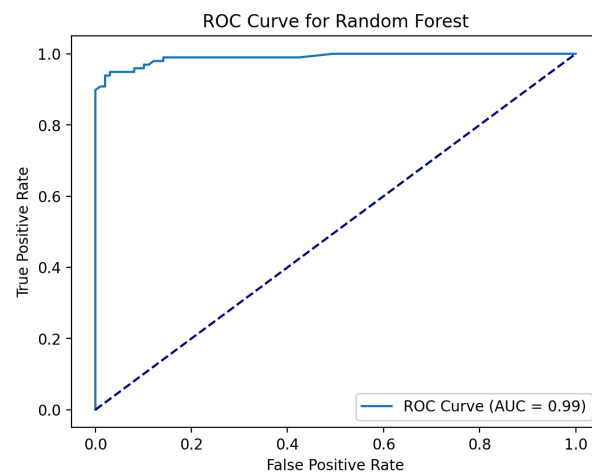


Fig. 4.11 ROC Curve for Random Forest

ROC Curve for Decision Tree:

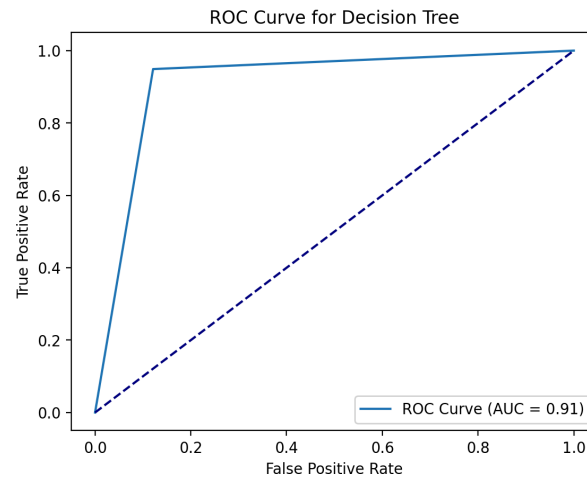


Fig. 4.12 ROC Curve for Decision Tree

Chapter 5

Conclusion and Future Work

4.1 Conclusion and Future Work

Conclusion

The credit card fraud detection project successfully demonstrates the application of machine learning algorithms to identify fraudulent transactions in a real-world financial context. Through the development of a user-friendly Streamlit interface, we have made it easier for users—such as financial analysts and risk management teams—to upload transaction data, obtain predictions, and visualize results. The model was evaluated using various metrics, revealing its effectiveness in distinguishing between legitimate and fraudulent transactions. By providing high precision and recall, the system aims to reduce financial losses associated with fraud while minimizing the inconvenience to genuine customers.

The combination of robust data processing and a streamlined interface not only enhances the operational efficiency of fraud detection processes but also empowers users with the ability to make informed decisions based on predictive analytics.

Future Work

While the current implementation demonstrates promising results, there are several avenues for future work to further enhance the system:

1. **Real-time Fraud Detection:** Developing a real-time detection system would allow for immediate alerts and responses to potentially fraudulent transactions, minimizing losses further.
2. **Incorporating Additional Features:** The model can be improved by integrating more features such as transaction location, merchant information, and customer behavior patterns. This additional context can enhance the model's accuracy and reliability.
3. **Model Optimization:** Exploring advanced algorithms, such as deep learning techniques or ensemble methods, could improve performance metrics. Continuous model tuning and validation using new data are essential for maintaining effectiveness.

4. **User Experience Enhancements:** Further improvements to the Streamlit interface, such as adding user authentication, detailed error handling, and customizable visualization options, can increase usability.
5. **Deployment and Scalability:** Transitioning the application from a local environment to a cloud-based platform would allow for greater scalability and accessibility, making the system available to a wider audience.
6. **Integration with Financial Systems:** Collaborating with financial institutions to integrate the fraud detection system into their existing transaction monitoring systems could provide a more seamless user experience and immediate impact on fraud prevention.

By addressing these areas, the project can evolve into a more comprehensive and effective tool for combating credit card fraud, ultimately contributing to the integrity of financial transactions and enhancing consumer trust.

References

- [1] M. Zareapoor, S. K. . Seeja.K.R, and M. Afshar Alam, “Analysis on Credit Card Fraud Detection Techniques: Based on Certain Design Criteria,” *Int. J. Comput. Appl.*, vol. 52, no. 3, pp. 35–42, 2012, doi: 10.5120/8184-1538.
- [2] A. RB and S. K. KR, “Credit card fraud detection using artificial neural network,” *Glob. Transitions Proc.*, vol. 2, no. 1, pp. 35–41, 2021, doi: 10.1016/j.gltp.2021.01.006.
- [4] R. Bin Sulaiman, V. Schetinin, and P. Sant, “Review of Machine Learning Approach on Credit Card Fraud Detection,” *Human-Centric Intell. Syst.*, vol. 2, no. 1–2, pp. 55–68, 2022, doi: 10.1007/s44230-022-00004-0.
- [5] B. Al Smadi and M. Min, “A Critical review of Credit Card Fraud Detection Techniques,” *2020 11th IEEE Annu. Ubiquitous Comput. Electron. Mob. Commun. Conf. UEMCON 2020*, pp. 0732–0736, Oct. 2020, doi: 10.1109/UEMCON51285.2020.9298075.
- [6] V. N. Dornadula and S. Geetha, “Credit Card Fraud Detection using Machine Learning Algorithms,” *Procedia Comput. Sci.*, vol. 165, pp. 631–641, 2019, doi: 10.1016/j.procs.2020.01.057.
- [7] M. Karim and R. M. Rahman, “Decision Tree and Naïve Bayes Algorithm for Classification and Generation of Actionable Knowledge for Direct Marketing,” *J. Softw. Eng. Appl.*, vol. 06, no. 04, pp. 196–206, 2013, doi: 10.4236/jsea.2013.64025.
- [8] V. Sushma, S. Neelamma, Y. Machaiah, and S. Fathima, “Credit Card Fraud Detection using Machine Learning,” *14th Int. Conf. Adv. Comput. Control. Telecommun. Technol. ACT 2023*, vol. 2023-June, no. 20, pp. 861–864, 2023, doi: 10.48175/ijarset-9488.
- [9] E. Ileberi, Y. Sun, and Z. Wang, “A machine learning based credit card fraud detection using the GA algorithm for feature selection,” *J. Big Data*, vol. 9, no. 1, 2022, doi: 10.1186/s40537-022-00573-8.

- [10] S. Khatri, A. Arora, and A. P. Agrawal, "Supervised Machine Learning Algorithms for Credit Card Fraud Detection : A Comparison," *2020 10th Int. Conf. Cloud Comput. Data Sci. Eng.*, pp. 680–683, 2020.
- [11] M.S P, A. Saini, S. Ahmed, and S. Sarkar, "Credit Card Fraud Detection using Machine Learning and Data Science," *Int. J. Eng. Res.*, vol. 08, Sep. 2019, doi: 10.17577/IJERTV8IS090031.
- [12] N. Tressa *et al.*, "Credit Card Fraud Detection Using Machine Learning," *2023 3rd Asian Conf. Innov. Technol. ASIANCON 2023*, pp. 488–493, 2023, doi: 10.1109/ASIANCON58793.2023.10270805.
- [13] F. Itoo, Meenakshi, and S. Singh, "Comparison and analysis of logistic regression, Naïve Bayes and KNN machine learning algorithms for credit card fraud detection," *Int. J. Inf. Technol.*, vol. 13, no. 4, pp. 1503–1511, 2021, doi: 10.1007/s41870-020-00430-y.
- [14] A. Karthikeyan and M. Jakkani, "Credit Card Fraud Detection on Imbalanced Dataset using Supervised Machine Learning Algorithms," *Int. J. Innov. Res. Sci. Eng. Technol. | An ISO*, vol. 10, no. 8, p. 11783, 2021, doi: 10.15680/IJIRSET.2021.1008156.
- [14] A. Karthikeyan and M. Jakkani, "Credit Card Fraud Detection on Imbalanced Dataset using Supervised Machine Learning Algorithms," *Int. J. Innov. Res. Sci. Eng. Technol. | An ISO*, vol. 10, no. 8, p. 11783, 2021, doi: 10.15680/IJIRSET.2021.1008156.
- [15] Y. Y. Song and Y. Lu, "Decision tree methods: applications for classification and prediction," *Shanghai Arch. Psychiatry*, vol. 27, no. 2, pp. 130–135, 2015, doi: 10.11919/j.issn.1002-0829.215044.
- [16] E. Odhiambo Omuya, G. Onyango Okeyo, and M. Waema Kimwele, "Feature Selection for Classification using Principal Component Analysis and Information Gain," *Expert Syst. Appl.*, vol. 174, no. January, p. 114765, 2021, doi: 10.1016/j.eswa.2021.114765.
- [17] M. Devika, R. Ravi Kishan, L. Sai Manohar, and N. Vijaya, "Credit Card Fraud Detection Using Logistic Regression," *2nd IEEE Int. Conf. Adv. Technol. Intell. Control. Environ.*

- Comput. Commun. Eng. ICATIECE 2022, vol. 11, no. 4, pp. 471–477, 2022, doi: 10.1109/ICATIECE56365.2022.10046976.
- [18] A. S. Rathore, A. Kumar, D. Tomar, V. Goyal, K. Sarda, and D. Vij, “Credit Card Fraud Detection using Machine Learning,” *Proc. 2021 10th Int. Conf. Syst. Model. Adv. Res. Trends, SMART 2021*, pp. 167–171, 2021, doi: 10.1109/SMART52563.2021.9676262.
 - [19] J. Kumar, A. K. Singh, and R. Buyya, “Ensemble learning based predictive framework for virtual machine resource request prediction,” *Neurocomputing*, vol. 397, pp. 20–30, 2020, doi: 10.1016/j.neucom.2020.02.014.
 - [20] A. Husejinović, “Credit card fraud detection using naive Bayesian and c4.5 decision tree classifiers,” *Period. Eng. Nat. Sci.*, vol. 8, no. 1, pp. 1–5, 2020.
 - [21] H. Z. Alenzi and N. O. Aljehane, “Fraud Detection in Credit Cards using Logistic Regression,” *Int. J. Adv. Comput. Sci. Appl.*, vol. 11, no. 12, pp. 540–551, 2020, doi: 10.14569/IJACSA.2020.0111265.
 - [22] Y. Sahin and E. Duman, “Detecting credit card fraud by ANN and logistic regression,” *INISTA 2011 - 2011 Int. Symp. Innov. Intell. Syst. Appl.*, pp. 315–319, 2011, doi: 10.1109/INISTA.2011.5946108.
 - [23] J. K. Pun, “Improving credit card fraud detection using a meta-learning strategy,” pp. 1–133, 2011.
 - [24] Y. Sahin and E. Duman, “Detecting credit card fraud by decision trees and support vector machines,” *IMECS 2011 - Int. MultiConference Eng. Comput. Sci. 2011*, vol. 1, pp. 442–447, 2011.
 - [25] S. Maes, K. Tuyls, and B. Vanschoenwinkel, “Credit Card Fraud Detection Using Bayesian and Neural Networks,” *Maciunas RJ, Ed. Interact. image-guided neurosurgery. Am. Assoc. Neurol. Surg.*, no. March, pp. 261–270, 1993.
 - [26] Abu Rbeian, Alsharif Hasan & Ashqar, Huthaifa. (2023). Credit Card Fraud Detection

Using Enhanced Random Forest Classifier for Imbalanced Data.

- [27] Samaneh Sorounejad, Z. Zojaji, R. E. Atani, and A. H. Monadjemi, "A Survey of Credit Card Fraud Detection Techniques: Data and Technique Oriented Perspective," pp. 1–26, 2016, [Online]. Available: <http://arxiv.org/abs/1611.06439>
- [28] N. S. Alfaiz and S. M. Fati, "Enhanced Credit Card Fraud Detection Model Using Machine Learning," *Electron.*, vol. 11, no. 4, 2022, doi: 10.3390/electronics11040662.
- [29] R. Powar and R. Dawkhar, "and Engineering Trends Credit Card Fraud Detection Using Machine," vol. 5, no. 9, pp. 41–46, 2020.
- [30] G. Goy, C. Gezer, and V. C. Gungor, "Credit Card Fraud Detection with Machine Learning Methods," *UBMK 2019 - Proceedings, 4th Int. Conf. Comput. Sci. Eng.*, no. March, pp. 350–354, 2019, doi: 10.1109/UBMK.2019.8906995.
- [31] N. Rtayli and N. Enneya, "Selection features and support vector machine for credit card risk identification," *Procedia Manuf.*, vol. 46, pp. 941–948, 2020, doi: 10.1016/j.promfg.2020.05.012.
- [32] J. K. Jaiswal and R. Samikannu, "Application of Random Forest Algorithm on Feature Subset Selection and Classification and Regression," 2016, doi: 10.1109/WCCCT.2016.25.
- [33] K. Michalak and H. Kwasnicka, "Correlation based feature selection method," *Int. J. Bio-Inspired Comput.*, vol. 2, no. 5, pp. 319–332, 2010, doi: 10.1504/IJBIC.2010.036158.
- [34] F. Zhou, H. Fan, Y. Liu, H. Zhang, and R. Ji, "Hybrid Model of Machine Learning Method and Empirical Method for Rate of Penetration Prediction Based on Data Similarity," *Appl. Sci.*, vol. 13, no. 10, 2023, doi: 10.3390/app13105870.
- [35] V. V. Madhav and K. A. Kumari, "Analysis of Credit Card Fraud Data using PCA," *IOSR J. Eng. www.iosrjen.org ISSN*, vol. 10, no. 1, pp. 2278–8719, 2020, [Online]. Available: www.iosrjen.org