A Comprehensive Study on Credit Card Fraud Detection: Utilizing Feature Fusion Engineering and Explainable AI

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 452

Course Title: Project and Thesis Sessional

By

Al Mahmud Siam
Student ID: 1902062
Session: 2019

Md. Mostafijur Rahman
Student ID: 1902073
Session: 2019

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
HAJEE MOHAMMAD DANESH SCIENCE AND TECHNOLOGY UNIVERSITY,
DINAJPUR-5200, BANGLADESH

September 2024

Table of Contents

List of Figures

List of Tables

List of Algorithms

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 SMOTE resampling techniques on the training dataset. In this paper we apply five machine learning algorithms: decision tree, random forest, KNN, ANN, and Extreme Gradient Boosting on 4 methods (PCA, GA, ensemble learning: voting and a hybrid feature selection technique) and select the best method based on its performance. After the selected method, choose the best model based on its Accuracy, Precision, Recall, F1-score, AUC-ROC curve and precision recall curve and time. To understand and interpret the decisions or predictions made by ML models we use a method called SHAP of Explainable AI (XAI).

Keywords – Sampling technique, Feature Engineering, Machine Learning, Imbalanced dataset, Credit Card Frauds detection, Explainable AI(XAI)

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 a imbalanced dataset from the from European cardholders' September 2013 for balancing the dataset we have used a sampling approach called Synthetic Minority Oversampling Technique (SMOTE) is an oversampling technique 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 which are Principal Component Analysis (PCA), Genetic Algorithm (GA), K-fold cross-validation, ensemble learning, and correlation. After selection of important features we deployed the data to any of the machine learning algorithms. Total seven supervised ML algorithms are used 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 Oversampling: As credit card fraud dataset is highly unbalanced so the use SMOTE technique. The primary goal of SMOTE is to balance the class distribution by artificially increasing the number of instances in the minority class. This oversampling is achieved by introducing synthetic examples, which do not correspond to actual observations but are generated to represent potential instances of the minority class.

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.

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 (SMOTE) 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 decision tree, random forest, XGBoost ,KNN, ANN 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 Synthetic Minority Oversampling Technique (SMOTE) 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.

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.

Chapter 4 delves into the conducted experiments and presents detailed findings. It critically evaluates and discusses the outcomes of these experiments.

Chapter 5 concludes the work presented in this thesis 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

This research focuses on the application of the following supervised ML algorithms for credit card fraud detection: Decision Tree (DT), Random Forest (RF), Artificial Neural Network (ANN), k-nearest neighbors (KNN), and Extreme Gradient Boosting classifier(XGBC). ML systems are trained and tested using large datasets. Ileberi, E. et al. [45] used 5 ML algorithms (ANN, DT, RF, NB, LR) for their research work. In paper [46], authors Sohony, I. et al. applied an ensemble majority voting algorithm for their research proposal. In this study, the authors used 3 feed-forward neural networks and 2 RF algorithms for their research work. Varmedja, D. et al. [47] used 4 supervised ML algorithms (LR, NB, ANN and RF) for their research work to detect credit card fraud. Yılmaz, A. A. [48] used 5 supervised ML algorithms to detect credit card fraud. In this paper the author used RF, DT, LR, NB and ANN for their research work in this field. Khare, N., & Sait, S. Y. [49] used 4 ML algorithms for detect credit card fraud. In this

paper, the authors used LR, SVM, DT, and RF to detect credit card fraud. In this paper, we used 5 ML algorithms to detect credit card fraud.

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 the tree looks like.

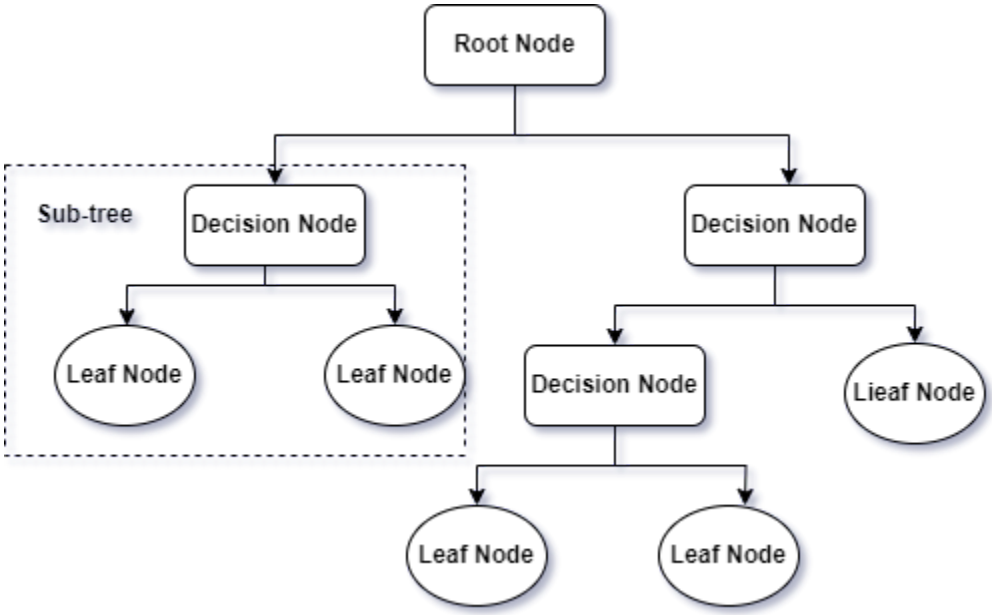


Fig. 2.1: Graphical presentation of Decision Tree

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

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

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

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.

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

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

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

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

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.

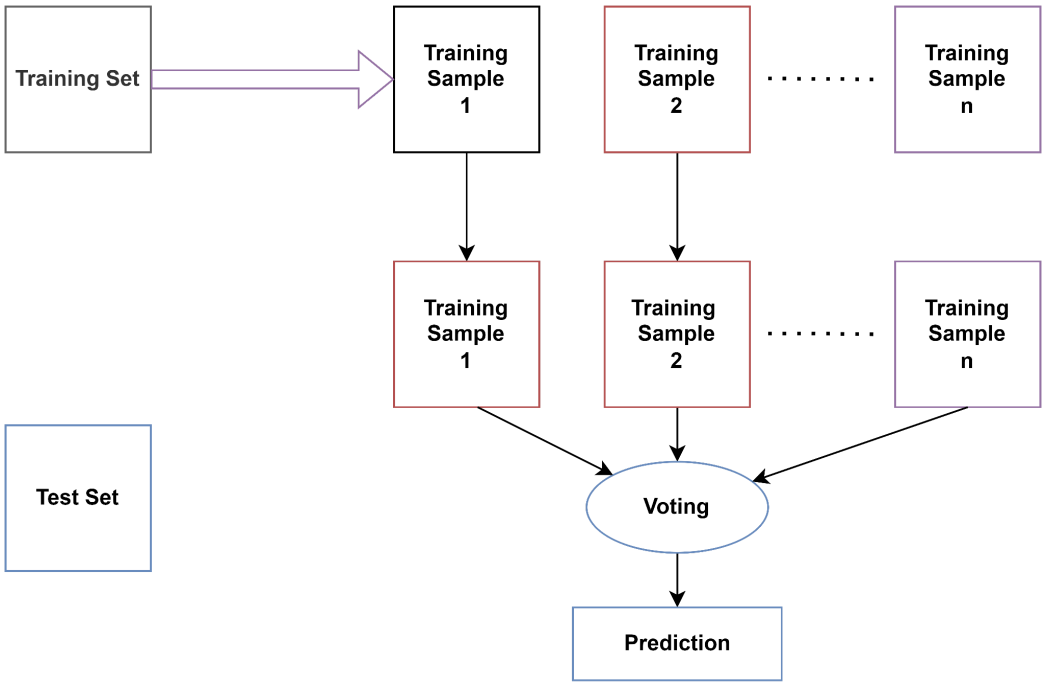


Fig. 2.2: Working mechanism of Random forest algorithm

2.4.3 K-Nearest Neighbor (KNN)

K-Nearest Neighbor is a type of simplest but very powerful supervised machine learning technique which is responsible for both classification and regression analysis on problems. It is a non-parametric algorithm that does not make assumptions on underlying data and is also called a lazy learner algorithm [10]. It classifies the data based on similarity measures by majority votes using the k-value that refers to the number of nearest neighbors. The K-value is selected perfectly for better performance and the similarity between two data is measured by Euclidean Distance or Manhattan distance or Minkowski distance. These distance measures are calculated as follows:

Euclidean Distance: $d(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$ (2.6)

Manhattan distance: $d(x, y) = \sum_{i=1}^{n} |x_i - y_i|$ (2.7)

Minkowski distance: $d(x, y) = \left(\sum_{i=1}^{n} |x_i - y_i|^c \right)^{\frac{1}{c}}$ (2.8)

Minkowski's distance is a generalization of the Euclidean and Manhattan distance [14]. For continuous attributes, the Euclidean distance is a useful distance metric, and optimizing the distance metrics can help the KNN algorithm perform better [29]. Both real and fraudulent data samples were needed for this technique's training. This approach is fast, helps in improving detection, and has a high false alert rate [4].

Finding the abnormalities in the targets may be possible by doing over-sampling and data separation in datasets. KNN helps credit card fraud detection with less processing power and memory and faster techniques for any number of datasets. When compared to other anomaly-based methods, KNN yields more accurate and efficient outcomes [13].

2.4.4 Artificial Neural Network (ANN)

An artificial neural network (ANN) is a network of interconnected nodes created to mimic how the human brain works. Every node in neighboring layers has a weighted connection to several other nodes. Individual nodes get input from other linked nodes, and they compute output by combining the weights using an activation function [27]. The fundamental components of the most basic ANN are an input layer, a single hidden layer, and an output layer. Shapes and architectures vary for these components. The amount of features in a particular dataset determines the size of the input layer, task difficulty determines the size of the hidden layer, and problem type determines the size of the output layer [4]. The most well-known neural networks used are feed-forward neural networks, also known as multilayer perceptrons [22]. It has a minimum of three node layers: the input, hidden, and output layers. Every node uses the activation function. The activation function adds bias after calculating the weighted sum of its inputs. This enables us to determine which neurons need to be eliminated and not taken into

account in external connections [30]. Figure 1 depicts a graphical representation of a simple ANN with 4 nodes in the input layer, a hidden layer with 5 nodes, and an output layer with 1 node.

In credit card fraud detection, initially, the network is trained to employ the cardholder's usual conduct. The transactions that seemed to be fraud are being back propagated then categorized as fraudulent and non-fraudulent across the network. This method was discovered to be highly effective whilst a neural network can handle data at high rates without reprogramming [18].

2.4.5 XGBoost

XGBoost stands for eXtreme Gradient Boosting used for classification, regression, and problems related to ranking. This approach is an advanced version of the Gradient boosting approach Chen & Guestrin, (2016), which improves the results compared to the Gradient boosting approach. It uses ensemble techniques to improve results. Ensemble techniques are used to modify existing classification models to handle imbalanced class distributions. In ensemble learning, classification problems are solved by training multiple learners. Its central concept is to combine numerous weak learners into keen learners to boost the classifier's performance. Figure 4 shows the idea of ensemble learning techniques.

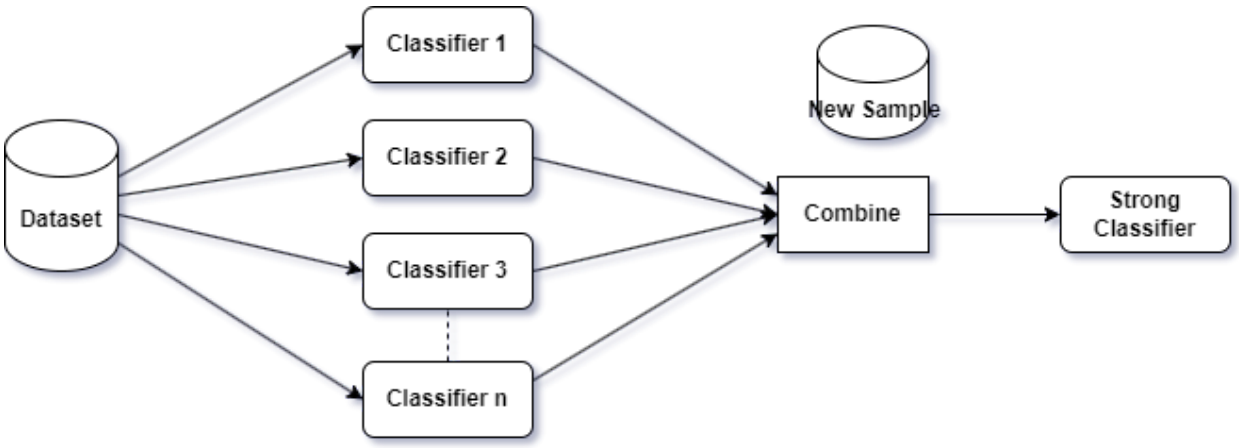


Fig. 2.3: Concept of ensemble learning technique using XGboost

Boosting is also an ensemble technique. It combines various weak learners to build a keen learner that provides improved results compared to individual learners. In each boosting step, weak learners are sequentially trained and correct their predecessor by adding weights to

previously misclassified samples. In boosting, the technique bootstrapping is used to avoid variance and overfitting. The XGBoost model uses boosting techniques for classification and predictions.

XGBoost adopts a more generalized method to control overfitting and contributes to improving the results. XGBoost works on parallel computing; hence it is fast as compared to the standard Gradient boosting approach. It can handle missing data and includes cross-validation features used to determine a boost round in each run. XGBoost needs a few parameters to tune for getting better results

2.5 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), Artificial Neural Network (ANN), and Naive Bayes (NB) 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].

The random forest (RF) technique is used in this study [26] to solve the given problem. Managing imbalanced datasets by using the hyperparameters approach and the over-sampling technique (SMOTE) to improve the random forest classifier's performance. To compare more techniques based on recall, accuracy, F1 score, and ROC, further methods would be used.

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.

Two distinct important strategies for pre-processing data to attain high detection accuracy while handling noisy, imbalanced, and outlier-filled data are mean-based and clustering-based techniques [21]. The logistic regression voting classifier and the support vector machine classifier are two well-known classifiers that provide superior results in terms of accuracy, sensitivity, and error rate [21]. Response time is a classifier's weak point, but it is possible to boost speed while also taking real-time data security and privacy into account.

2.6 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. Collaboration, standardization of evaluation metrics, and ethical considerations are essential for advancing the field responsibly. Through these efforts, organizations can build more resilient fraud detection systems, ultimately mitigating financial losses and maintaining consumer trust in electronic transactions.

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.

Table 1. Dataset Statistics

Number of variables	31
Number of observations	284807
Missing cells	0
Duplicate rows	773
Total size in memory	67.4 MiB
Numeric variables	30
Categorical variables	1

The dataset statistics and variable types shown in Table. 1 provide an overview of the dataset's structure and content. Notably, there are no missing cells, which means every entry in the dataset is complete. However, there are 773 duplicate rows, constituting 0.3% of the total data.

The total size of the dataset in memory is 67.4 MB.Regarding the variable types, the dataset primarily consists of numeric variables—30 out of the 31 are numeric. There is only one categorical variable, indicating that most of the data is numerical in nature. This division of variable types is essential for determining the appropriate statistical and machine learning methods to analyze the data.

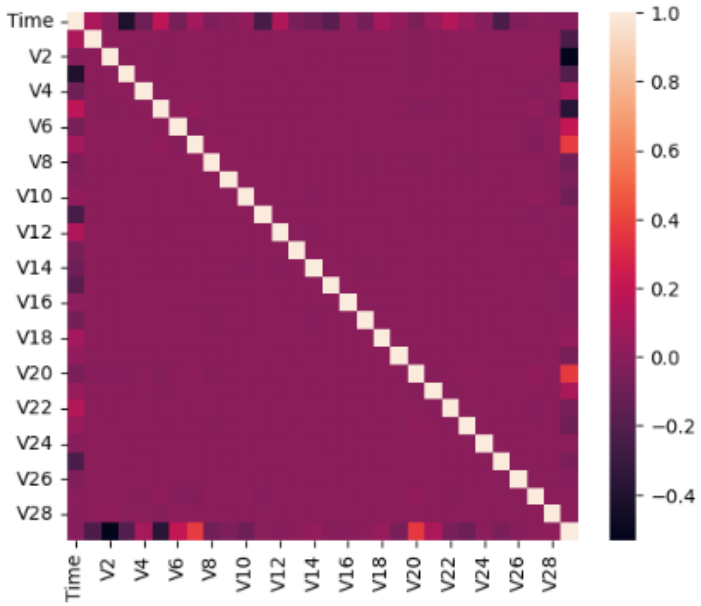
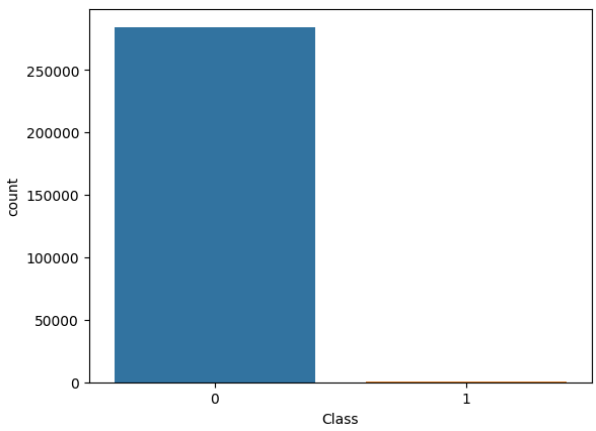


Fig. 3.1: Correlation matrix with heatmap

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 oversampling technique [12].

The Synthetic Minority over-sampling technique (SMOTE) is amongst of the most dominant techniques that are used to address the issue of class imbalance that is found in datasets such as the ones used to build credit card fraud detection ML based models [44]. The SMOTE method generates samples of a specific class by connecting a data point with its k nearest neighbors. The SMOTE method generates synthetic data points that are not a direct replica of the minority class instance [45].



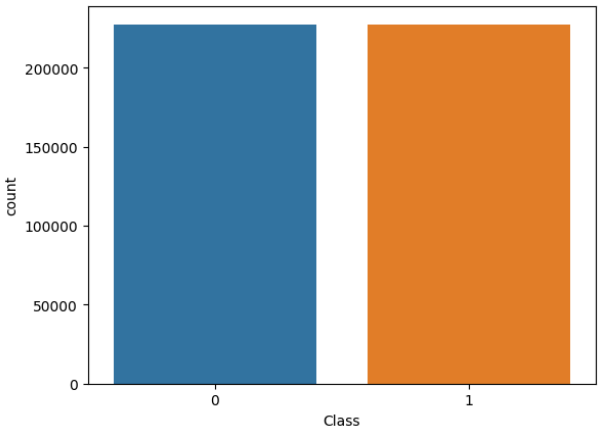


Fig. 3.2 (a): Class Distribution before Sampling Fig. 3.2(b): Class Distribution after Sampling

3.2 Proposed framework

Fig. 3.4 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 hybrid feature selection techniques, Genetic algorithm and ensemble majority voting 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. On the otherhand PCA technique is applied before the SMOTE methodology because if we SMOTE it before PCA then there occur data leakage problem. After the model is developed the evaluation method is done using the testing dataset. For performance evaluation, seven performance metrics are used those are accuracy, precision, recall, f1 score, AUC score, PR score and time. After that comparing the models performance, we suggest the best one. Finally, XAI techniques are applied to interpret and explain the model's behavior and performance.

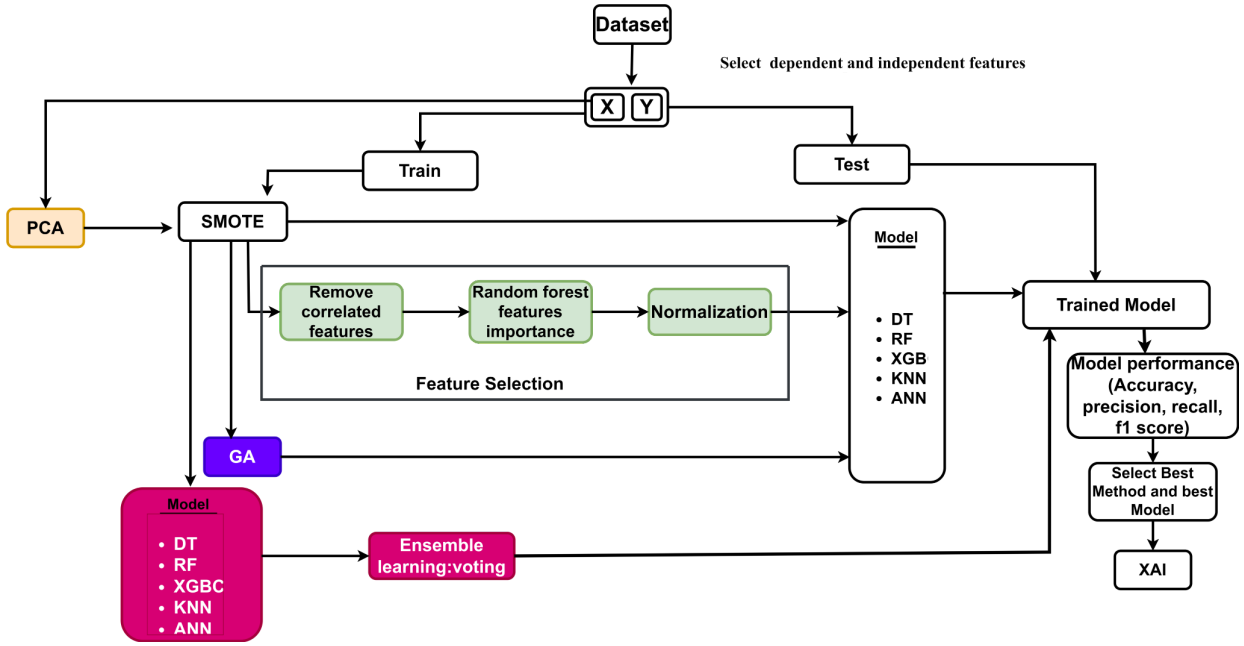


Fig. 3.3: The proposed framework

3.3 Algorithms on the proposed work

Algorithm 1 Data preprocessing

1: **Input:** Dataset $D = \{(Xi, Yi)\}^{n}_{i=1}$ where $Xi \in \mathbb{R}^{m}$ and $Yi \in \{0,1\}$ // n is no. of features and m is no. of samples

2: Models ← [DT,RF,XGBC,ANN,KNN]

3: PerformanceMetrics ← [Accuracy, Precision, Recall, F1 Score, AUC Score, PR Score, Time] // Measure models performance

4: M_{Best} ← [] // Store the best score base on the performance metrics

5: **Output:** Best model M_{Best} and feature selection method

7: $X = \{X_1, X_2, \ldots, X_n\}$, $Y = \{Y_1, Y_2, \ldots, Y_n\}$ // Split dataset D into feature matrix X and label vector Y

8: $(X_{train}, Y_{train}), (X_{test}, Y_{test})$ = **TrainTestSplit**(X, Y)

// Split data into training and testing sets

9: $(X_{SMOTE}, Y_{SMOTE}) = SMOTE(X_{train}, Y_{train})$ //balance the dataset

Algorithm 2 Principal Component Analysis (PCA)

10: $X_{PCA} =$ **PCA**(X) // Apply PCA to the X

11:$(X_{train}, Y_{train}),(X_{test}, Y_{test})$=**TrainTestSplit**$(X_{PCA}, Y)$

// Split data into training and testing sets

12: $X_{SMOTE\_PCA}, Y_{SMOTE\_PCA}$ = **SMOTE**(X_{train}, Y_{train})

// Apply SMOTE to balance the classes of Y

13: Models.***fit*** $(X_{SMOTE\_PCA}, Y_{SMOTE\_PCA})$ //Fit the models

14: $Y_{pred} \leftarrow$ Models.**predict**(X_{test}) //Predict the models

15: M_{Best}.**add**(Y_{pred}, Y_{test}) // Store best models according to all 5 models

Algorithm 3 Feature Selection

16:$S = \{X_{SMOTE_i} \mid \forall j \neq i, |C_{ij}| < \tau, i = 1, 2, \ldots, n\}$ // Remove highly correlated features, where C_{ij} represents the correlation between features X_i and X_j, and τ is the correlation threshold.

17: $S' \leftarrow$ **RandomForestImporatance**(S) // Select best features

18: $S_{Normalize} \leftarrow$ **Min-Max Scaling**(S') // Normalize the selected features

19: Models.***fit*** $(S_{Normalize}, Y_{SMOTE})$ //Fit the models

20: $Y_{pred} \leftarrow$ Models.**predict**(X_{test}) // Predict the models

21: M_{Best}.**add**(Y_{pred}, Y_{test}) //Store best models according to all 5 models

Algorithm 4 Genetic Algorithm

22: B←[] // An empty array to save selected features

23: Set, k←Total no. of iterations

24: Compute the initial Population PI with X

25: Compute the fitness method

26: Calculate optimized fitness value, q

27: i← 1
28: Update the list B until i is less than k:
29: Conduct the crossover
30: Compute mutations
31: Calculate the fitness
32: Generate optimize fitness score, q
33: Update the list B
34: Models.fit(B,Y_{SMOTE})
35: Y_{pred} ← Models.predict(X_{test})
36: M_{Best}.add(Y_{pred}, Y_{test})

Algorithm 5 Ensemble Majority Voting

37: Set, i ← 1
38: **for** each i in Models **do**
39: EnsembleModel.**fit**($X_{SMOTE,}Y_{SMOTE}$, voting = 'hard')
// Fit ensemble model with selected features
40: end **for**
41: Y_{pred}← EnsembleModel.predict(X_{test}) // Make predictions on the test set
42: M_{Best}.**add**(Y_{pred}, Y_{test})

Algorithm 6 Apply XAI

43: XAI ($X_{SMOT\ E}$, $Y_{SMOT\ E}$) //Interpretability through XAI

3.4 Methods Used in Models

3.4.1 Features Selection Techniques

3.4.1.1 Random Forest Importance

Random forest is a group of decision trees which we sometimes call decision trees. Classification trees are constructed by random forests. It also adds a new classification tree to the forest. One must add it to each individual tree. Each classification tree's leaf nodes consist of the target variable and intermediate nodes consist of other dependent variables. In a forest

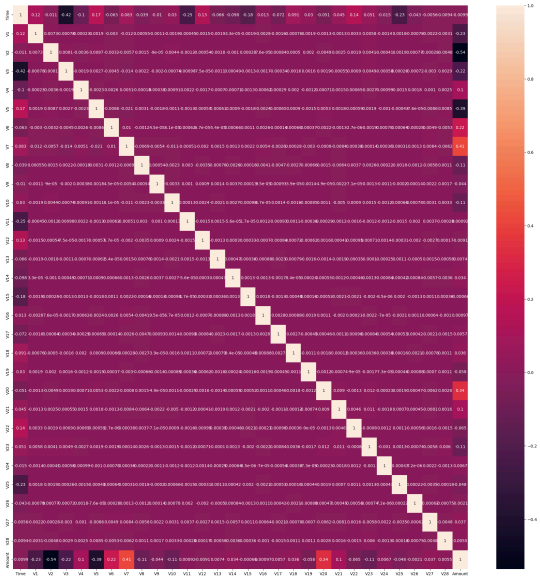
each tree provides its target variable and receiving the most target variable from all trees in a forest selects it as the final target [32].

But in the random forest the error rate will increase if the correlation between any two trees is increasing [32].

So before applying random forest in our dataset to select the best features we need to apply correlation to find out correlated features.

3.4.1.2 Correlation

The Synthetic Minority Over-sampling Technique (SMOTE) can lead to higher correlation between features due to several factors. By generating synthetic samples through interpolation between existing minority class instances, SMOTE creates new data points that are more similar to each other, thereby increasing feature correlation. This process enhances the density of the minority class in the feature space, which can result in stronger linear relationships among features. Additionally, the synthetic 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 SMOTE 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-SMOTE may be necessary to mitigate any adverse effects on model performance.



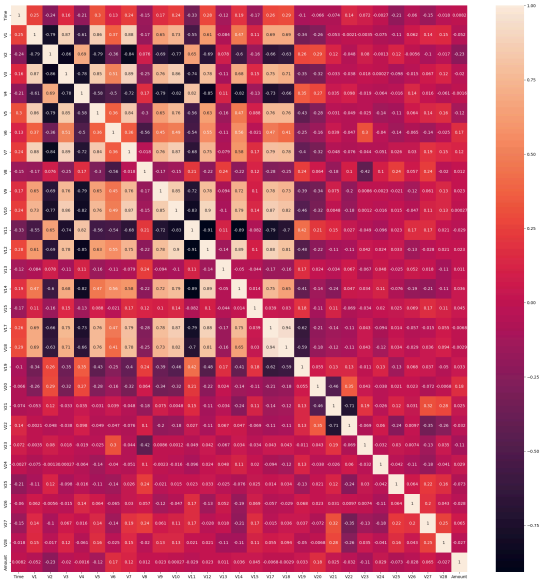


Fig 3.4(a): Correlation before creating SMOTE Fig.3.4(b): Correlation after creating SMOTE

The main goal of correlation is to reduce the amount of features. 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.

Formula of correlation,

$$\text{Corr(x,y)} = \frac{cov(x,y)}{\sqrt{var(x)}.\sqrt{var(y)}} \qquad (3.1)$$

Where cov(x,y) is covariance between x and y means how x and y are related to one another.

$$\text{cov(x,y)} = \frac{\Sigma\left(x_i - \bar{x}\right)\cdot\left(y_i - \bar{y}\right)}{N} \qquad (3.2)$$

where N is total no. of samples in the dataset

$$\text{var(x)} = \frac{\Sigma\left(x_i - \bar{x}\right)^2}{N} \qquad (3.3)$$

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

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 V16 from our train dataset. Now our dataset has 29 features available for train the model.

3.4.1.3 Feature selection Using Random Forest Importance

After applying correlation on the whole dataset except the Class column now we find out the top most important feature. For selecting top important features in the dataset we apply a random forest model. The random forest model receives different contributions from each feature in the dataset. Then we exclude less significant features. First apply a random forest model in the selected features of the dataset that contain 28 features that have no correlation between them. Then make a random forest feature importance plot to determine which features are most significant. Finally we select a threshold value whose feature importance value is more than the threshold select those features. In this paper we used threshold value 0.015 which means the importance having more than 1.5% impact on the model.

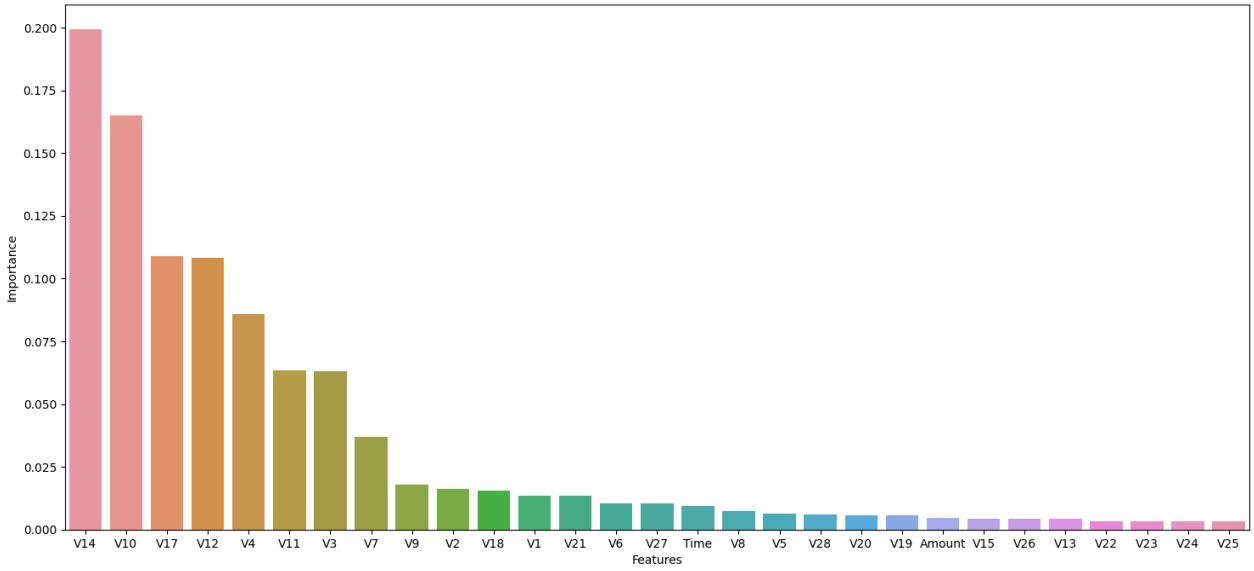


Fig. 3.5: Random Forest Feature Importance

In Fig. 3.6 we see top 14 features 'V14', 'V10', 'V17', 'V12', 'V4', 'V11', 'V3', 'V7', 'V9', 'V2','V18', 'V1', 'V21', 'V6' have an importance value of more than 0.0105.

3.4.1.4 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 applies 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.

Formula of normalization, $Y_{norm} = \frac{Y - Ymin}{Ymax - Ymin}$ (3.4)

Where Y is the input data, $Ymin$ is the minimum input value and $Ymax$ is the maximum input value and $Ynorm$ is the final output of converting input data to normal range [0,1].

This process improves the training speed and model performance better than before and reduces the dimension and value range of different features [10].

3.5.2 Principal Component Analysis

PCA is a 'Feature Extraction technique whose goal is to create a new, smaller set of features that contain the most useful data. It is a Dimensionality Reduction Algorithm. Its goal is to minimize the amount of input features while preserving as much of the original data as feasible. By utilizing theis technique we first submit our dataset on the PCA. PCA then generates new features. We refer to these new features as PCs. Our new features create lower dimensionality than the original dataset [35]. The Fig. 3.7 shows the overall working flow of the PCA.

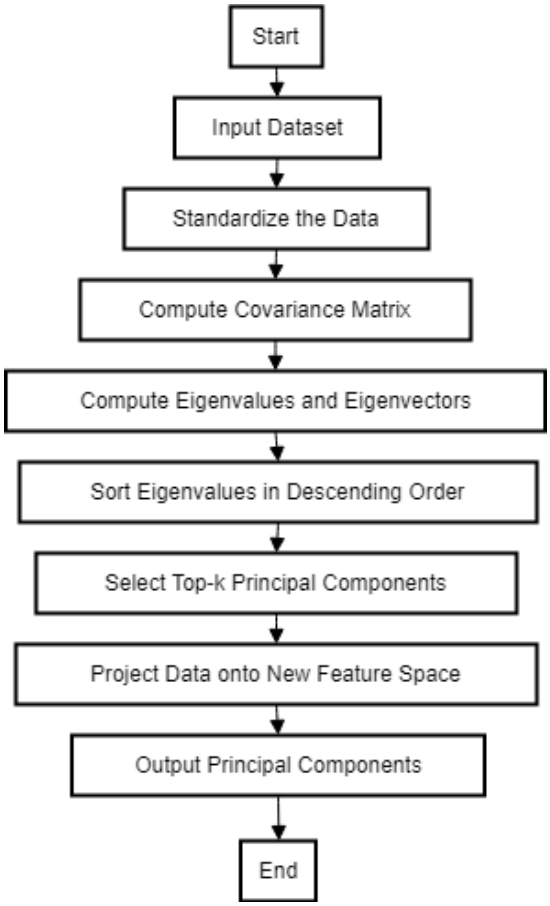


Fig. 3.6: Flow chart of PCA

Step of PCA Algorithm:

Step 1: Standardization

Here we first calculate the mean of every variable(features) of our dataset. Then we minus this mean form every data of each variable(features). Then we divide it by standard deviation.Our final value is

$$Z = \frac{x-\mu}{\sigma} \tag{3.5}$$

where x is input value , μ is the mean of each features and σ is standard deviation

So variable mean is 0 standard deviation is 1. Which means each feature contributes equally to the analysis.

Step 2: Correlation Matrix

In this step here we calculate the covariance matrix of the variables. The main goal of this step is to check the relationship between the variables. By using this we understood the variables differ from the mean. For 3 variables a, b, c the covariance matrix is a 3×3 form.

$[Cov(a,a)\ Cov(a,b)\ Cov(a,c)\ Cov(b,a)\ Cov(b,b)\ Cov(b,c)\ Cov(c,a)\ Cov(c,b)\ Cov(c,c)\ ]$

Step 3: Compute Eigenvalues and Eigenvectors

Let A be our covariance matrix and I is the identity matrix of the same shape square.

We need to find out eigenvalues λ and eigenvector X .Where AX = λX .

To find out the eigenvalues λ and eigenvector X we follow the steps:

Setp-1: First we calculate λI .

Step-2: Then we calculate A- λI

Step-3: Then we find out the determinant of step 2

det(A- λI).

From step 3 we find eigenvalues λ and eigenvector X.

Eigenvalues λ represent the magnitude of variance and eigenvectors X indicate its direction.

Step -4: Choose the number of principal components: Now setting eigenvalues in decreasing order. Number of principal components depends on the number of eigenvectors. We chose those eigenvectors whose corresponding eigenvalues are high.

Step-5: Final Data:

Multiply the selected principal component (eigenvectors) by the standardized data which we get in step1.

Final data = (eigenvectors) * standardized data

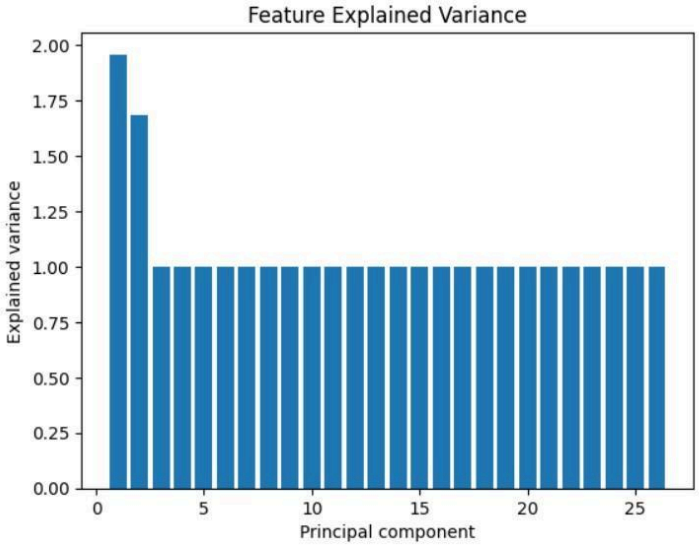


Fig. 3.7: Relation between explained variance and principal component

[1.95826326 1.6836999 1.00000351 1.00000351 1.00000351 1.00000351
 1.00000351 1.00000351 1.00000351 1.00000351 1.00000351 1.00000351
 1.00000351 1.00000351 1.00000351 1.00000351 1.00000351 1.00000351
 1.00000351 1.00000351 1.00000351 1.00000351 1.00000351 1.00000351
 1.00000351 1.00000351]

In Fig. 3.8. we plot a bar chart that shows the explained variance against principal components. It represents after the dimensionality reduction how much information is retained from the original dataset. For each principal component it shows the explained variance. This figure shows that all of the principal components are not significant because the first principal component explained variance is 1.95, second principal component explained variance is 1.68, the third to next 24 principal components explained variance is 1.00000351. Gradually next principal components explained variance is decreasing or equal.

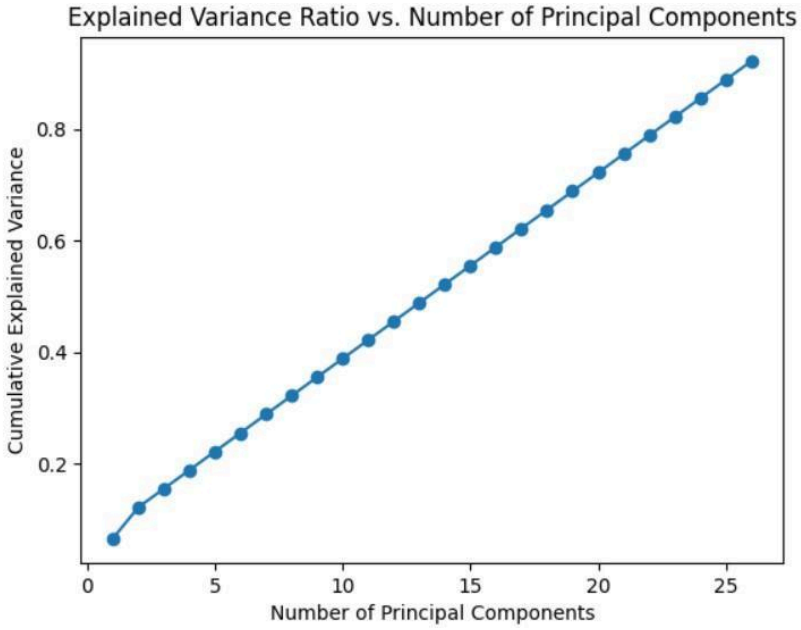


Fig. 3.8: Relation between cumulative explained variance and principal components

So we select those principal components which can retain more than 90% variance of the original dataset. Fig. 3.9. shows the cumulative explained variance vs the number of principal components. Here the first 26 principal components retain more than 90% variance of the original dataset. PCA reduced 30 features to 26 principal components. Therefore we select 26 principal components as [PC1 ,PC2...PC26]. Required time is also reduced.

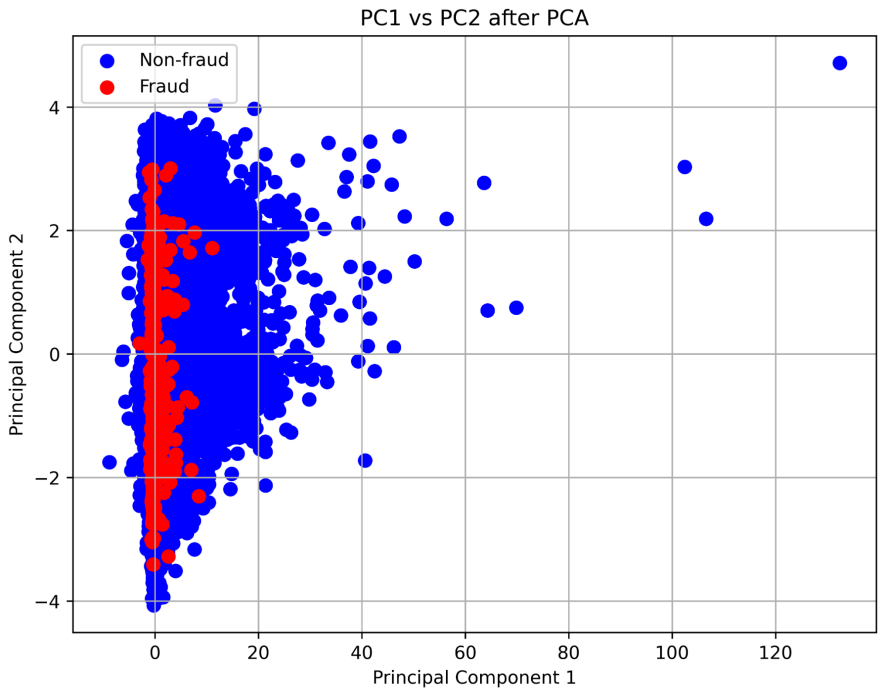


Fig. 3.9: PC1 vs PC2 comparison

Figure 3.10 shows a Principal Component Analysis (PCA) transformation, plotting the first two principal components (PC1 and PC2) on a scatter plot, where blue dots represent non-fraudulent transactions and red dots represent fraudulent ones. PCA is used to reduce the dimensionality of a dataset while retaining as much variance as possible. In this plot, each point corresponds to a transaction projected into the new feature space defined by PC1 and PC2, which are linear combinations of the original features.

The significance of this plot lies in its ability to visually represent how fraud and non-fraud transactions are distributed after dimensionality reduction. While some separation between fraudulent and non-fraudulent data is visible (with fraud cases clustering more tightly along a vertical axis), there is still a large overlap. This indicates that distinguishing between the two classes based on these two principal components alone may be challenging. Such visualizations are helpful in understanding the complexity of the problem and assessing whether PCA is an effective method for separating these classes in the context of fraud detection.

PCA reduced 30 features to 26 principal components. Therefore we select 26 principal components as PC1 to PC26. Now we will apply some machine learning algorithms to check which one gives more performance as compared to other algorithms.

3.5.3 Ensemble learning

In recent decades, multiple classifier systems, also known as ensemble learning systems, have proven to be highly effective and adaptable across a wide range of problems. Ensemble learning involves combining multiple models to enhance the predictive power and stability of a classification model. This method significantly improves prediction accuracy. The process of merging different prediction models is referred to as ensemble learning. Initially, ensemble systems were developed to boost prediction accuracy by reducing variance. They have been successfully applied to numerous machine learning challenges, including feature selection, confidence estimation, incremental learning, and various classification and prediction tasks [52].

Through the weighting of multiple independent classifiers and their combination to produce a final conclusion, ensemble learning is used to increase the confidence of a classification model [53].

In ensemble systems, the models differ from one another in aspects such as population, hypothesis, modeling technique, and initial seed [54] combining Class Labels:

1. Let us first assume that only the class labels are available from the classifier outputs, and define the decision of the ith classifier as $d_{t,c} \in \{0,1\}$, $t=1,\ldots, T$ and c=1,...., C, where T is the number of classifiers and C is the number of classes. If t^{th} classifier (or hypothesis) h_t chooses class ω_c , then $d_{t,c} = 1$, and 0, otherwise. Note that the continuous valued outputs can easily be converted to label outputs (by assigning $d_{t,c} = 1$ for the class with the highest output), but not vice versa. Therefore, the combination rules described in this section can also be used by classifiers providing specific class supports.

Ensemble learning is employed to enhance the performance of machine learning algorithms by combining various models to boost prediction accuracy [37]. The main goal of ensemble systems originally was to decrease variance and improve prediction accuracy. They have proven effective in addressing a wide range of machine learning challenges, including feature selection,

confidence estimation, incremental learning, and various classification and prediction tasks [36].

Majority Voting

In this paper, we implement ensemble majority voting, which comes in three variations depending on how the ensemble decision is determined: (1) unanimous voting, where all classifiers must agree on the class; (2) simple majority, where the class is predicted by more than half of the classifiers; and (3) plurality voting, where the class with the highest number of votes is chosen, regardless of whether those votes exceed 50%. Unless otherwise specified, majority voting typically refers to plurality voting, which can be mathematically defined as follows: choose class ω_c , if

$$\sum_{t=1}^{T} d_{t,c*} = \max_c \sum_{t=1}^{T} d_{t,c} \tag{3.6}$$

If the outputs of the classifiers are independent, majority voting can be demonstrated as the optimal combination rule. To understand this, consider an odd number of T classifiers, where each classifier has a probability ppp of making a correct classification. The ensemble makes the correct decision when at least⌊ T/2⌋ + 1 of these classifiers select the correct label. Here, the floor function ⌊■⌋ returns the largest integer less than or equal to its argument.

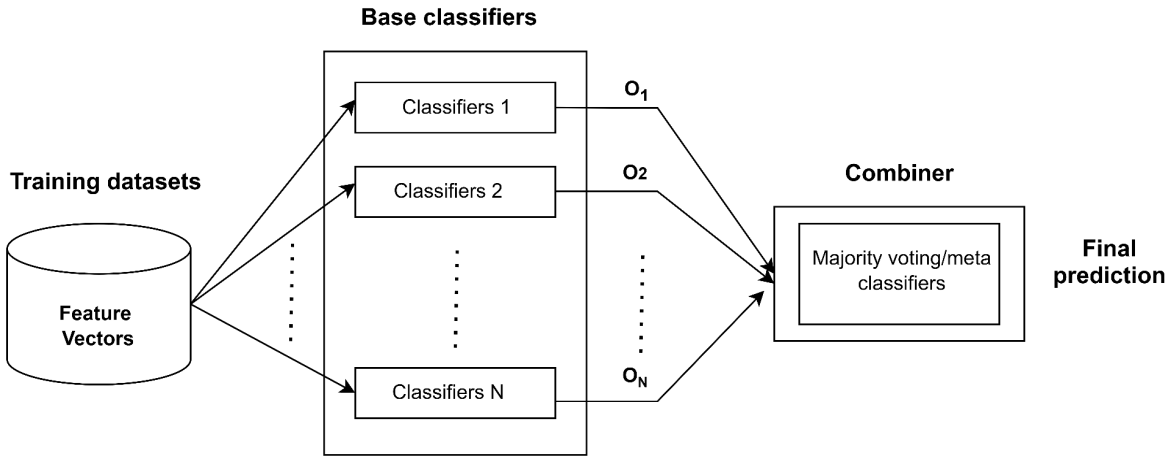


Fig. 3.10: Ensemble Voting Technique.

In this paper here training of 5 famous classifiers (LR, RFT, KNN, ANN, DT) on the training dataset are shown. Then aggregating each classifier's predictions. After that, predict the class that gets the most votes. This majority vote classifier is called a hard voting classifier.

3.5.4 Genetic Algorithm

Genetic Algorithms (GAs) are search-based algorithms rooted in the principles of natural selection and inheritance. They fall under a broader field known as evolutionary computation. In GAs, multiple diverse solutions are generated for a given problem. These solutions undergo recombination and mutation processes (similar to biological genetics), producing new offspring, and this cycle repeats over several generations [4]. Each individual solution is assigned a fitness value based on its objective function, with fitter individuals having a higher chance of reproducing to create even fitter offspring. This process gradually improves solutions over generations, continuing until a final goal is met. Genetic algorithms are largely driven by probability, but unlike local random search (which randomly generates solutions without identifying optimal ones), they perform better by incorporating historical data, making them more sophisticated and effective [38].

3.5.4.1 Operation in Genetic Algorithm

There are three type of operations in genetic algorithm which are given below with details description:

- Selection
- Crossover
- Mutation

a) Selection (Chromosome Encoding)

A chromosome will have information stored in a format pertaining to the personification of that solution. Binary string formats are a commonly used encoding format. This is how the chromosome will seem after that.

Chromosome 1 1101100100110110

Chromosome 2 1101111000011110

A binary string can be used to plot any chromosome. Every piece of information contained in the string is also accountable for holding some elements or requirements of the solution.

b) Crossover

Crossover procedure can begin as soon as the chosen coding is verified to be utilized. A portion of the parent chromosomes' genes are used in crossover, creating a new offspring [5]. The most straightforward way to do this is to randomly choose a crossover point by taking the range from the first parent point to this point. The cross-over point demonstration is displayed in the diagram below:

Chromosome 1	11011 | 00100110110
Chromosome 2	11011 | 11000011110
Offspring 1	11011 | 11000011110
Offspring 2	11011 | 00100110110

There are various methods to perform crossover, such as choosing multiple crossover points. Crossovers can also become more complex and detailed, depending largely on how the chromosome is encoded. To enhance the performance of genetic algorithms, it is important to select the appropriate crossover technique for specific problems.

c) Mutation

The following stage is called mutation after the crossing is completed. In order to prevent all solutions in the population from falling into a local optimum of the solved problem, mutation is done on purpose. Offspring is the consequence of crossing and is subject to random mutation. We can randomly choose which bits in binary encoding to change from 0 to 1 or 1 to 0.A mutation can be adorned like follows:

Original offspring 1	1101111000011110
Original offspring 2	1101100100110110
Mutated offspring 1	1100111000011110
Mutated offspring 2	1101101100110110

The encoding of chromosomes is the only factor that influences the process of crossover and mutation.

b) Crossover and Mutation Probability

Genetic algorithms consist of two basic parameters namely crossover probability and mutation probability.

c) Crossover probability

This can be linked to the frequency of crossover. If the offspring are identical to the parents, it indicates that no crossover has occurred. When crossover happens, parts of each parent combine to form the offspring. There is a 100% probability that offspring will result from a crossover, whereas if the entire new generation is merely a replica of the previous population's chromosomes (which doesn't always mean they'll be identical), the probability of crossover would be 0%. Essentially, crossover is performed to increase the likelihood that new chromosomes will inherit favorable traits from the old chromosomes, leading to better outcomes in future generations. It is beneficial to retain a portion of the old population to ensure their survival into the next generation.

d) Mutation probability

It can be described as the likelihood of changes occurring in parts of a chromosome. Without mutation, offspring are generated without any alterations prior to the crossover. The probability of mutation directly influences how much a chromosome changes—at 100%, the entire chromosome would change, while at 0%, no changes would occur. Mutation plays a crucial role in preventing genetic algorithms (GAs) from getting stuck in local optima [6]. However, the mutation rate should be kept low; otherwise, the GA could turn into a random search.

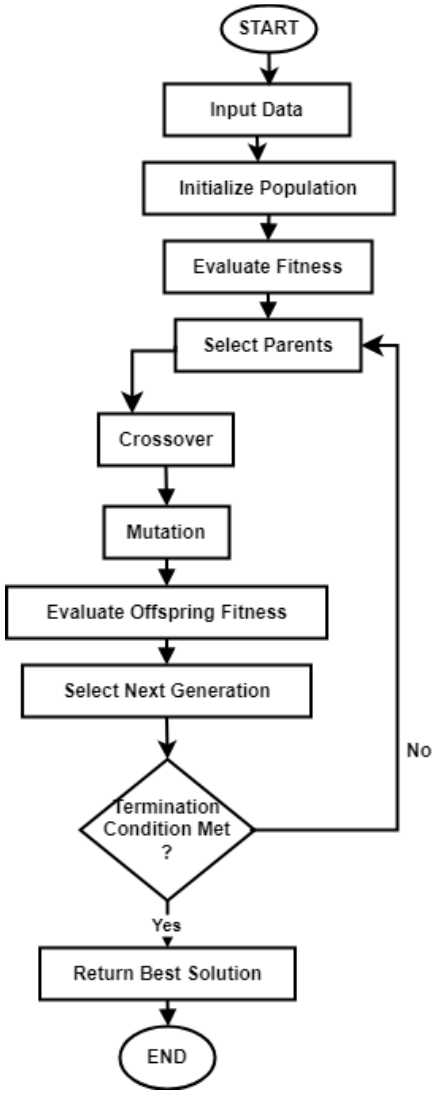


Fig. 3.11: Genetic Algorithm Flow Chart

First step aims at certain of the initial population and after that analyzing fitness factor of all the chromosomes by the application of fitness function. Seceding first step, chromosomes possessing fitness values similar to threshold will be selected as next generation parent.noe check this criteria, if it lies within the specified criteria then we need to stop, else any of two strings can be selected from initial population and after that crossover can be performed on it leading to offspring formation and again checking the criteria of star function, if yes then we need to stop, else the process need to be continued until the problem is solved satisfactorily.

Some advantages of genetic algorithm

- A number of advantages made GAs to be applicable in multiple areas -
- No need of Imitative information (which don’t exist for most of the real life problems)

- A more comprehensive and optimum in comparison to the primitive methods
- Possess a well aligned capabilities
- Enhancing both continuous as well as distinct functions along with multi-purpose problems
- Aims at delivering Best solutions instead of a single solution
- Having a satisfactory answer for the problems, which improves with time
- Works best in cases of large search space areas possessing multiple parameters into it.

It is an evolutionary algorithm. Its working methodology is inspired by the famous Charles Darwin's theory. In this theory we know how the genetic feature of a species can evolve over time for nature selection. Genetic algorithm (GA) concept is the same as Charles Darwin's theory of the natural selection process. It can be defined as a search based algorithm [42].

EA has some features [9]:

- Population: A set of possible solutions. Current population builds up the next generation by creating children.
- Gene: Each chromosome contains a set of bits. Each bit is called a gene.
- Chromosome/ individual: Consists of genes that maintain the values for the optimum variables.
- Fitness value: Each chromosome has its fitness value.

Step of Genetic Algorithm:

Step 1 : Initialization of population

Step 2 : Set N = no. of generation

Step 3: Set i=1

Step 4: While i<= N

Calculate the fitness value

Selection

Crossover

Mutation

In GA there are various solutions for a given problem. But we need to find the best possible optimal solutions.

The acquired solutions then need recombination and mutation (same as the biological concept). Then it creates a new generation and the process repeats over multiple generations. Each

individual (chromosome in biological concept) has its own unique fitness value. The more fitness values are able to create fit individuals. By this process of creating better individuals using the fitness value till the end it go to its goal of better optimal solution [38].

3.6 Explainable AI (XAI)

3.6.1 Introduction

XAI (Explainable Artificial Intelligence) refers to techniques and methods that allow humans to understand and interpret the decisions or predictions made by AI models. In traditional "black-box" AI models, especially deep learning and complex ensemble methods like Random Forests or XGBoost, it can be difficult to understand how the model arrived at a particular decision. XAI helps demystify these models by providing human-readable explanations, ensuring transparency, accountability, and trust.

The goal of the Explainable AI (XAI) program is to develop a set of machine learning methods that:

- Generate models that are more understandable without compromising their learning performance (prediction accuracy), and
- Help human users comprehend, trust, and efficiently oversee the next generation of AI systems.

3.6.2 Key aspects of XAI

Some important terms are given with details to understand this concept.

Transparency: Transparency in machine learning can be categorized into model transparency, design transparency, and algorithm transparency. A model is considered transparent if its structure and the relationship between inputs and outputs can be described mathematically, such as representing the model as a sum of kernel functions. However, using a standard Gaussian kernel based on Euclidean distances lacks transparency since the reason for selecting certain nonlinear kernels may not be clear. Greater transparency can be achieved by replacing Euclidean distance with more suitable measures. Design transparency involves choosing an appropriate kernel based on comparisons with others in kernel-based machine learning models. This design process leads to algorithm transparency, which overlaps with design transparency.

Algorithm transparency helps users better understand the model's output and enhances their interaction with explainable AI systems.

Interpretability: Interpretability refers to the extent to which a model's decision-making process can be understood in human terms. It focuses on explaining the factors a machine learning algorithm considers when making a decision. In unsupervised learning, interpretability helps gain deeper insights into the data . Additionally, interpretability is crucial in addressing problems like significant differences between future and past data due to changing consumer habits, technological advances, or socio-political factors. Interpretability also involves delivering explanations that are understandable to humans.

Explainability: Explainability refers to the collection of interpretable elements that contribute to a decision in a specific case. Unlike interpretability, explainability requires additional contextual information drawn from domain knowledge to provide meaningful explanations. Raw explanations from explainable AI methods, which may be clear to data experts, are often difficult for general users to grasp unless presented in accessible forms such as visualizations, textual summaries, or numerical data. Explainability plays a key role in justifying machine learning decisions, increasing accountability, and making the decisions more acceptable in applications like loan approvals, hiring, medical diagnoses, and insurance assessments. It also builds trust in recommendation systems and autonomous systems, as well as in critical decision-making processes. Moreover, explanations help identify biases in algorithms, which can arise from biased training data or flawed learning processes. Machine learning explanations allow users to detect bias, thus reducing algorithmic discrimination. Additionally, they give end-users insight into how their data is being used, such as in generating personalized ads or news feeds on social media.

3.6.3 Need of explanations of Models

Explanations for models are necessary in specific situations due to the time and effort they require. Basically three key cases where explanations should be provided. First, explanations are needed when a decision has a significant impact on users rather than decision-makers, as users may have a right to understand the decision. Second, explanations are necessary when decisions are contested, either to reverse or correct an error, or to establish accountability and seek compensation for any harm caused. However, if explanations can positively influence future decisions or the behavior of the decision-making system, they are also valuable. If no

corrective action can be taken for a harmful decision, the need for an explanation decreases. Third, explanations are required when there are suspicions regarding the inputs, outputs, or context of the decision-making process. For instance, certain features like race, gender, or sexual identity should not influence model inputs, and inexplicable results could indicate flawed predictions. Additionally, explanations are expected when users are concerned about the integrity of the system, even without specific suspicions about the outcome [50].

By addressing challenge problems in two areas—(1) machine learning problems to classify events of interest in heterogeneous, multimedia data, and (2) machine learning problems to construct decision policies for an autonomous system to perform a variety of simulated missions—the XAI program focuses on the development of multiple systems. These two challenging problem domains were selected to symbolize the meeting point of two significant machine learning methodologies (classification and reinforcement learning) and two crucial operational domains (autonomous systems and intelligence analysis). The following Fig 3.13 shows differences between the traditional system and the XAI.

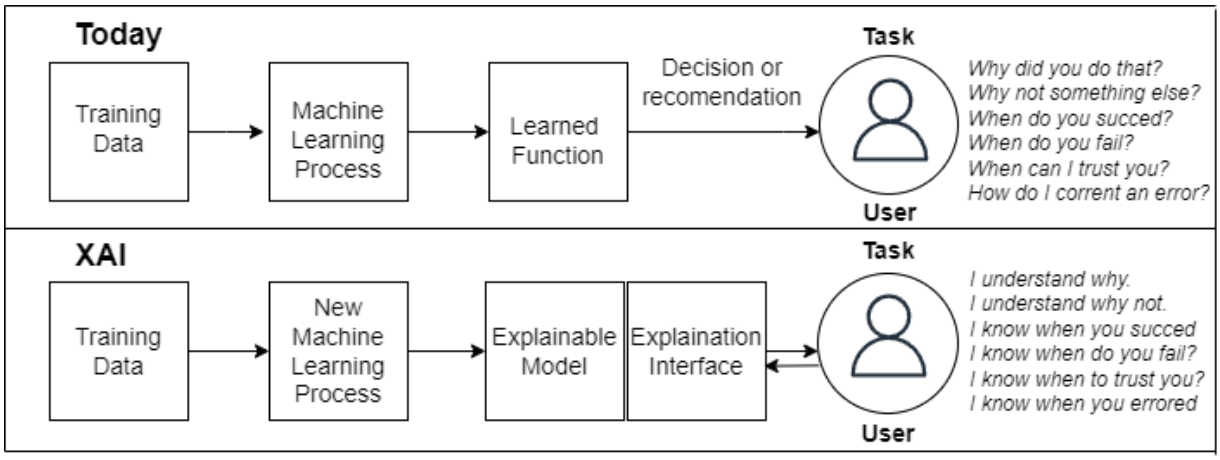


Fig. 3.12: Traditional System vs XAI

3.6.4 Working flow of XAI

XAI (Explainable Artificial Intelligence) refers to techniques and methods that allow humans to understand and interpret the decisions or predictions made by AI models. In this study our aim is to explore and assess Explainable AI (XAI) techniques for credit card fraud detection [42]. To accomplish this, the following five objectives are outlined:

Objective 1: Examine the current use of machine learning (ML and XAI techniques in credit card fraud detection to build a foundational understanding for implementing the subsequent objectives.

Objective 2: Apply four ML methods to a credit card fraud dataset and assess their performance based on accuracy, recall, precision, and F1 score.

Objective 3: Implement SHAP and LIME techniques and apply them to the results obtained from Objective 2.

Objective 4: Evaluate the explainability of the results from Objective 3 through a user study.

Objective 5: Analyze the findings from the user study and offer insights and recommendations for future research.

3.7 Real life scenario with our proposed model

The Fig. 3.15 depicts the real life of a machine learning (ML)-based fraud detection system for credit card transactions with the XAI. The process begins when a cardholder performs a transaction at an ATM or online, which is recorded and stored in a transaction history database. These transactions are then analyzed by an ML engine designed to identify fraudulent activities. This involves gathering user transaction histories, encrypting sensitive information, and storing it in a data warehouse. The bank manager and data scientist review the ML engine's decisions to either approve or flag transactions as fraudulent.

On the right side of the diagram, the specific steps for training and deploying the ML model are detailed. The collected dataset goes through feature selection, potentially using techniques such as PCA (Principal Component Analysis) or GAA (Genetic Algorithm Analysis), along with feature selection and ensemble majority voting. The dataset is divided into training and testing sets, with preprocessing steps like resampling methods (e.g., SMOTE) applied to address dataset imbalances. Various classifier models (e.g., DT, XGBoost, ANN, KNN, RF) are trained, tested, and evaluated. The best-performing model is then chosen and deployed for real-time fraud detection. This section emphasizes the tasks managed by an ML engineer to ensure an effective fraud detection system.

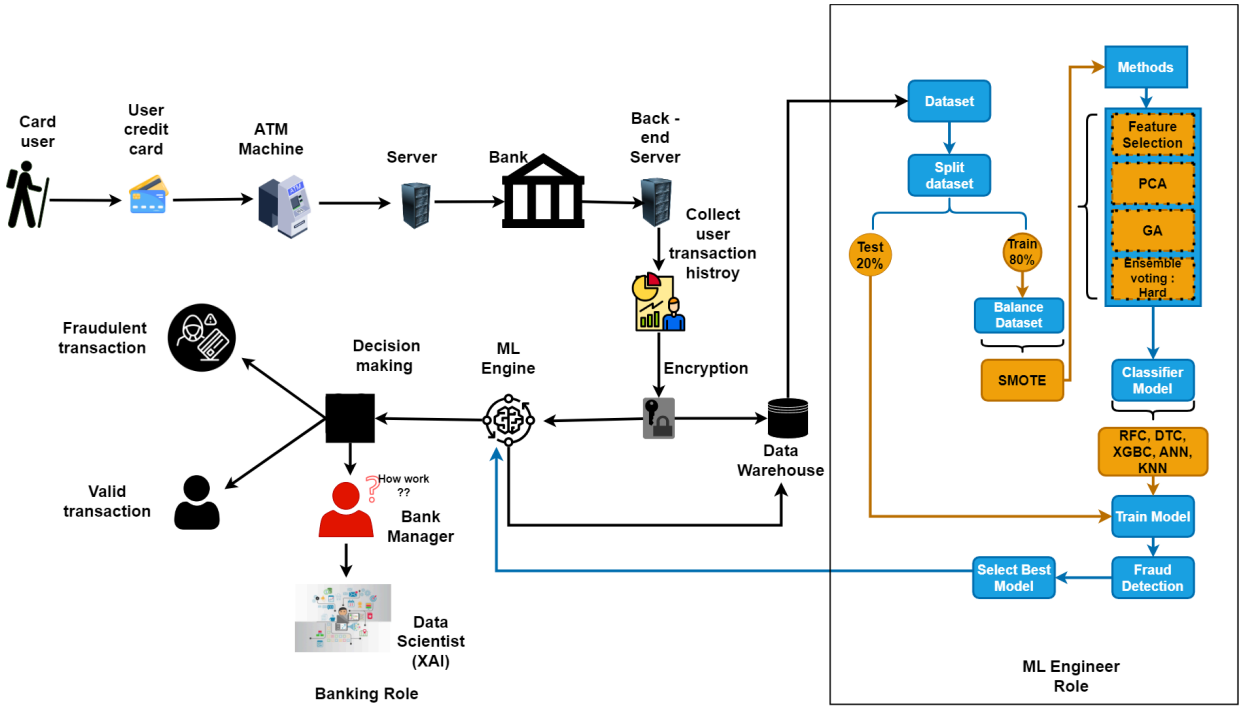


Fig. 3.13: Synergistic Roles of Banking Systems and Machine Learning Engineers in Credit Card Fraud Detection with XAI

3.8 Conclusion

This section provides an in-depth exploration of Explainable Artificial Intelligence (XAI) and its significance, particularly in making AI systems more transparent, interpretable, and trustworthy. It allows users to understand how AI models make decisions, making these systems more trustworthy and accountable. XAI is especially important in high-impact domains like healthcare, finance, and autonomous systems, where decisions must be not only accurate but also understandable. By offering human-readable explanations, XAI bridges the gap between technical complexity and user trust. Techniques like SHAP and LIME make it possible to explain even complex models like Random Forests, enabling users to understand why certain predictions, such as fraud detection, are made. Additionally, XAI helps identify and reduce bias in AI systems, ensuring fairer and more ethical decision-making. Explanations are especially needed in situations where decisions affect users directly or are contested. By conducting user studies, XAI research ensures that explanations are useful and meaningful to non-experts. Ultimately, XAI is paving the way for AI systems that are not just powerful but also

transparent, ethical, and reliable. This contributes to the broader goal of making AI more accountable and trustworthy in real-world applications.

Chapter 4

Result Analysis and Discussion

4.1 Cross validation

Cross-validation should be used either to (I) identify the appropriate level of complexity for training the model or (II) fine-tune the model's parameters. In this method, the dataset D is divided into two subsets: the training set T and the test set RRR. These subsets are disjoint (their intersection is empty), and together they cover the entire dataset (their union forms the complete dataset).

$$T \cup R = D \tag{4.1}$$

$$T \cap R = \varnothing \tag{4.2}$$

The model is trained on the training set T, while the test set R is used to assess its performance after training. Two commonly used cross-validation methods are K-fold cross-validation and Leave-One-Out cross-validation (LOOCV). In this work, K-fold cross-validation is employed to enhance model evaluation, minimize variance, make efficient use of data, and compare different models.

In K-fold cross validation, we randomly split the dataset $\mathcal{D}$ into K partitions $\{\mathcal{D}_1, \ldots, \mathcal{D}_K\}$ where,

$$|\mathcal{D}_1| \approx |\mathcal{D}_2| \approx \cdots \approx |\mathcal{D}_K|, \tag{4.3}$$

$$\bigcup_{i=1}^{K} \mathcal{D}_i = \mathcal{D}, \tag{4.4}$$

$$\mathcal{D}i \cap \mathcal{D}j = \varnothing, \ \forall i, j \in \{1, \ldots, K\}, \ i \neq j, \tag{4.5}$$

In K-fold cross-validation, one partition is used as the test set while the remaining data is used for training during each of the K iterations. The overall estimation error is calculated as the average test error across all rounds. While K = 10 is the most commonly used value, other values like K = 2, 5, and 10 are also frequently seen in the literature. In this study, K = 10 is applied. Typically, cross-validation divides the dataset into two parts: training and testing. It evaluates the model using the test set and predictions made from the training set. However, a limitation of this method is that only a specific portion of the data is used for both training and testing, which can sometimes overlook key information and result in inaccurate outcomes.

Fig. 4.1: Split Train test

To overcome this limitation we use k fold cross validation.

Steps of k-fold cross validation:

Step-1: Randomly split the dataset into k portions and each portion contains N/k (N is the total dataset) dataset.

Step-2: For j=1 to k continue step-3 to 4 for each model.

Step-3: Each j considered as test set and remaining k-1 are considered train set. So there is no data eliminated for training and testing.

Step-4: Calculate performance metric (accuracy and f1 score) of each j.

Step-5: Finally calculate the mean of each performance metric for each model.

In fig- 2 we use k=10. In fig-2 [a] fold 1 use for test data and fold 2 to 10 for train [39].

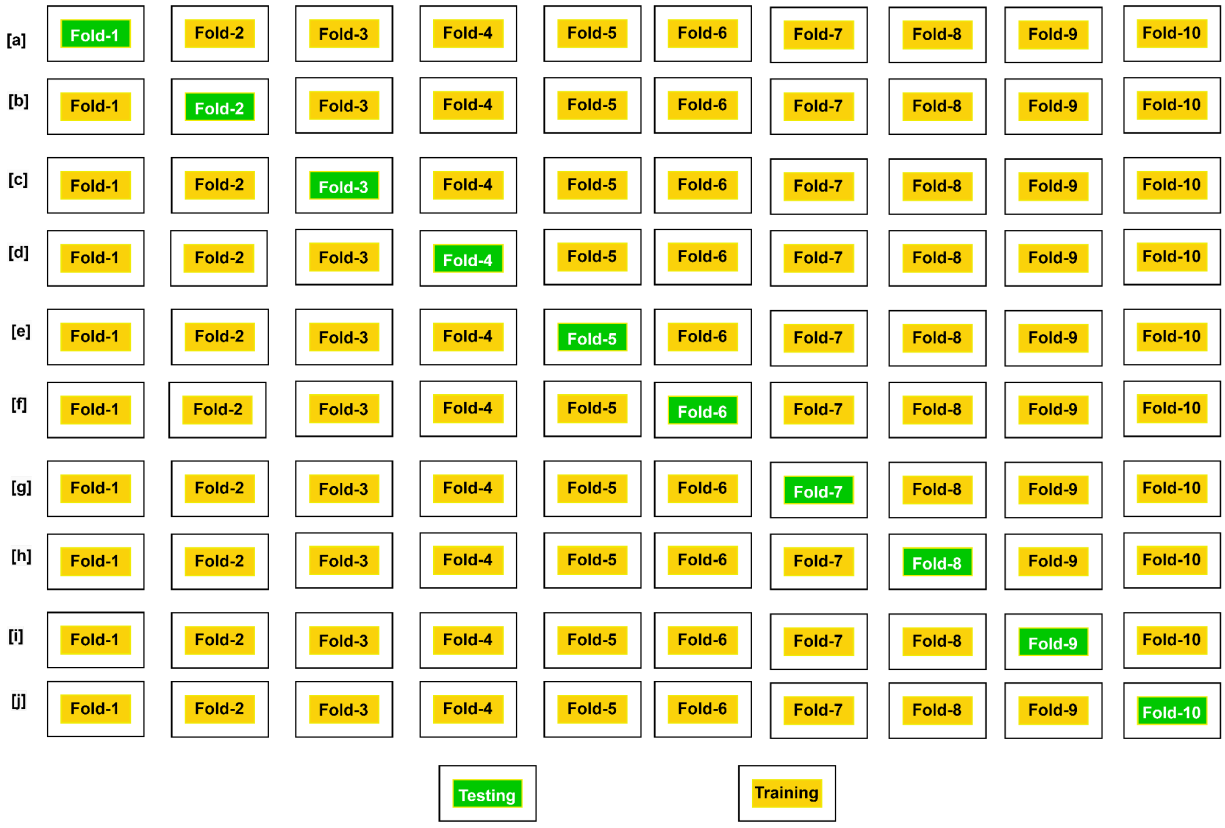


Fig. 4.2: 10 folds Cross Validation

In this paper we use k=10. After the Smote our dataset contains a total 568630 samples. Each portion contains 56863 samples. 56863 samples for test and remaining 511767 for train.

The study performed 10 fold cross validation on 5 models (XGBC,DTC,RFC,ANN,KNN) on the train dataset and calculated their performance metric (Accuracy , F1 measure).

Table-2: Performance comparison of ML models in CV

Fold	XGBC		DTC		RFC		ANN		KNN	
	ACC	F1	ACC	F1	ACC	F1	ACC	F1	ACC	F1
1	**99.97**	**99.97**	99.87	99.87	**99.97**	**99.97**	97.09	97.13	98.06	98.06
2	**99.99**	**99.99**	99.86	99.86	**99.99**	**99.99**	92.78	92.23	98.03	98.03
3	**99.99**	**99.99**	99.87	99.87	**99.99**	**99.99**	97.95	97.96	97.99	98.08
4	**99.98**	**99.98**	99.86	99.86	**99.98**	**99.98**	97.50	97.45	97.97	98.12
5	**99.99**	**99.99**	99.85	99.85	**99.99**	**99.99**	97.24	97.17	98.05	98.05
6	99.98	**99.99**	99.89	99.89	**99.99**	**99.99**	98.14	98.12	98.10	98.10
7	99.98	99.98	99.85	99.85	**99.99**	**99.99**	98.09	98.06	98.03	98.03
8	**99.98**	**99.98**	99.88	99.88	**99.98**	**99.98**	97.80	97.79	98.10	98.10
9	**99.98**	**99.98**	99.87	99.87	**99.98**	**99.98**	97.23	97.32	97.99	97.99
10	99.98	99.98	99.85	99.85	**99.99**	**99.99**	97.88	97.45	97.99	97.99
Avg.	**99.98**	99.98	99.86	99.86	**99.99**	**99.99**	97.17	97.06	98.05	98.06

In cross-validation (CV), the Random Forest Classifier (RFC) consistently performed better than the other models, though the differences in performance metrics were relatively small. RFC achieved an average accuracy of 99.99% and an F1 score of 99.99%, which is nearly identical to the performance of the XGBoost Classifier (XGBC). The XGBC had an average accuracy of 99.98%, along with an F1 score of 99.98%. These results underscore the effectiveness of ensemble-based methods in fraud detection tasks, where class imbalance is a significant challenge. The ability of both RFC and XGBC to handle imbalanced data efficiently is evident in their superior performance, both in terms of accuracy and F1 score.

The other models, while effective, did not achieve the same high performance. The Decision Tree Classifier (DTC) had an average accuracy of 99.86% and an F1 score of 98.86%, indicating that although the DTC provides strong performance in detecting fraudulent transactions, it does not match the precision and recall balance achieved by RFC and XGBC. This difference suggests that the DTC may have higher false positives or false negatives, affecting its reliability in a real-world setting.

The Artificial Neural Networks (ANN) model achieved an average accuracy of 97.17% and an F1 score of 97.17%. While ANN is often praised for its capacity to model complex relationships, it may require more tuning in terms of architecture, hyperparameters, or feature selection to match the performance of tree-based models in this specific application. ANN's

slightly lower accuracy and F1 score could also be attributed to the difficulty neural networks face with small or imbalanced datasets.

K-Nearest Neighbors (KNN) performed slightly better than ANN, with an average accuracy of 98.05% and an F1 score of 98.06%. However, KNN tends to be computationally expensive, especially when working with larger datasets, which can make it less ideal for real-time fraud detection systems where computational efficiency is crucial. Moreover, KNN's performance can degrade significantly if the feature space is not well-structured or if there is noise in the data, which is often the case in transaction datasets.

4.3 Real time Fraud Detection

Historically, fraud detection has been accomplished by applying machine learning models to large batches of previously completed transactions. Tracking down identified frauds was found to be quite difficult, and there have been numerous occasions where the fraudsters were able to perform many more fraudulent purchases before being revealed. This is because the consequences are seen after weeks or months. The application of fraud detection models in real-time, upon the completion of an online transaction, is known as fraud detection. In this manner, our system can identify scams in real time. The bank receives an alert from it that shows its fraud pattern and accuracy rate, which makes it simple for fraud monitoring teams to go to the next step without wasting time or money [51].

4.2 Performance Comparison of Methods

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:

$$Accuracy = \frac{TP+TN}{TP+FN+TN+FP} \tag{4.6}$$

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:

$$Precision = Positive\ Predicted\ Value = \frac{TP}{TP+FP} \tag{4.7}$$

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):

$$Recall = Sensitivity = TPR = \frac{True\ Positives}{Total\ Positives} = \frac{TP}{TP+FN} \tag{4.8}$$

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:

$$F1 = 2\,x\,\frac{Precision * Recall}{Precision + Recall} \tag{4.9}$$

Time is used as a parameter for performance evaluation of the various models that are used. We calculate the time for training the model and predicting the test data. The Time calculated is not the actual time, but the approximate time taken by a particular model. This parameter is used to compare the various models used based upon the time taken by them in handling the data [10]. In this paper we used only testing time to predict the test set [51]

Table-3: Comparative Performance Metrics for Fraud Detection Models

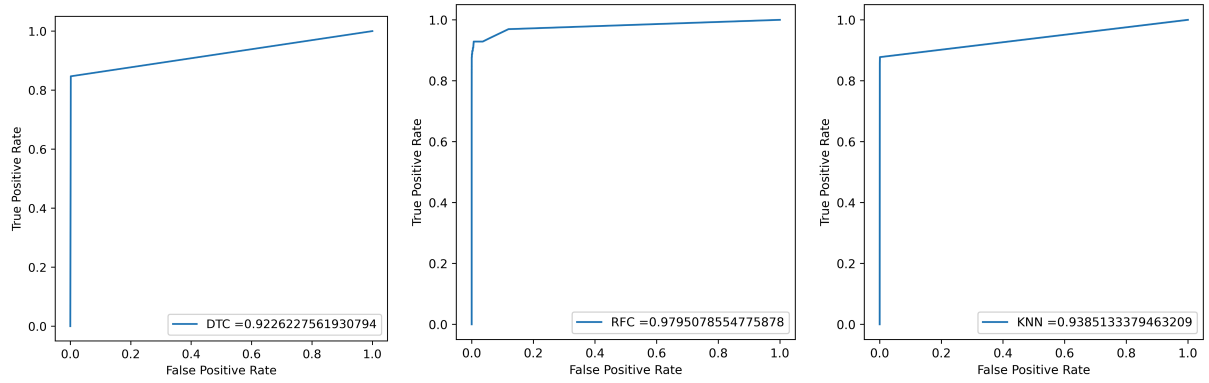
Methods	Performance Metrics	XGBC	RF	DT	ANN	KNN	Ensemble vote
Feature Selection	Accuracy	99.94	**99.95**	99.79	98.87	99.92	
	Precision	84.15	**88.65**	45.10	12.25	76.36	
	Recall	86.73	86.73	84.69	**89.79**	85.71	
	F1 score	85.42	**88.20**	58.86	21.56	80.76	
	AUC	**97.95**	**97.95**	92.25	97.40	93.85	
	PR	87.72	**88.35**	64.98	84.95	80.94	
	Time(in sec)	0.24	0.35	**0.007**	0.94	17.65	
GA	Accuracy	99.93	**99.94**	99.92	99.90	99.92	
	Precision	97.94	**98.70**	91.80	89.41	97.52	
	Recall	78.57	77.55	**80.61**	77.55	79.19	
	F1 score	87.00	86.85	85.86	83.06	**87.40**	
	AUC	95.39	94.76	92.00	**97.55**	92.34	
	PR	79.15	86.95	75.36	80.00	**88.00**	
	Time(in sec)	2.58	16.19	**0.54**	14.89	43.22	
Ensemble learning	Accuracy	99.94	**99.95**	99.85	92.13	97.03	99.94
	Precision	86.31	**87.23**	55.63	1.99	3.34	84.15
	Recall	83.67	83.67	80.61	**92.85**	58.16	86.73
	F1 score	84.97	85.41	65.83	3.90	6.31	**85.42**
	AUC	**98.93**	97.48	91.77	98.44	81.25	93.35
	PR	87.63	**88.62**	68.27	78.78	32.75	85.45
	Time(in sec)	0.14	0.17	**0.008**	0.155	142.52	314.31
PCA	Accuracy	99.92	**99.93**	99.79	**99.93**	**99.93**	
	Precision	82.72	**88.88**	49.43	83.56	82.30	
	Recall	81.25	78.57	78.57	77.67	**83.03**	
	F1 score	81.98	**83.41**	60.68	80.55	82.67	
	AUC	**96.53**	96.38	89.20	95.88	91.94	
	PR	**84.12**	83.42	63.37	80.28	81.64	
	Time(in sec)	0.57	0.25	**0.007**	0.14	64.36	

4.3 Visualization of Performance metrics

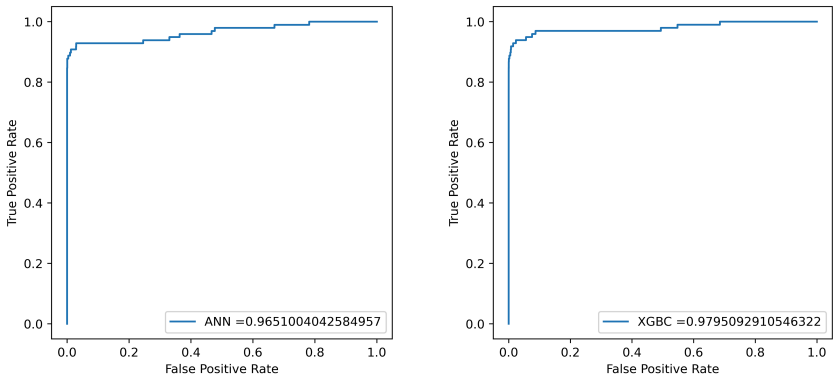
In this section, we compare various feature selection methodologies, including Feature Selection, PCA, Ensemble Voting, and the Genetic Algorithm, applied across five machine learning models: Random Forest (RF), Decision Tree (DT), XGBoost, K-Nearest Neighbors (KNN), and Artificial Neural Networks (ANN). Feature selection plays a critical role in improving model performance by selecting the most relevant features, which reduces overfitting and computation time while maintaining or enhancing predictive accuracy. Among the methods examined, Feature Selection with Random Forest emerges as the most effective across key metrics, such as accuracy, F1 score, and computation time, making it particularly well-suited for real-time applications like fraud detection. This section delves into a detailed comparison of these methodologies to highlight the balance between model performance and computational efficiency, offering insights into the best approach for handling imbalanced datasets and high-dimensional data.

4.3.1 ROC and PR curve

ROC and PR evaluate the performance of our used machine learning models while using these on different methods. As these are used particularly for binary classifiers we use these for performance evaluation. ROC curve plots the True Positive Rate (TPR) (Sensitivity) against the False Positive Rate (FPR) at various threshold settings. On the other hand PR curve plots Precision against Recall for different thresholds.

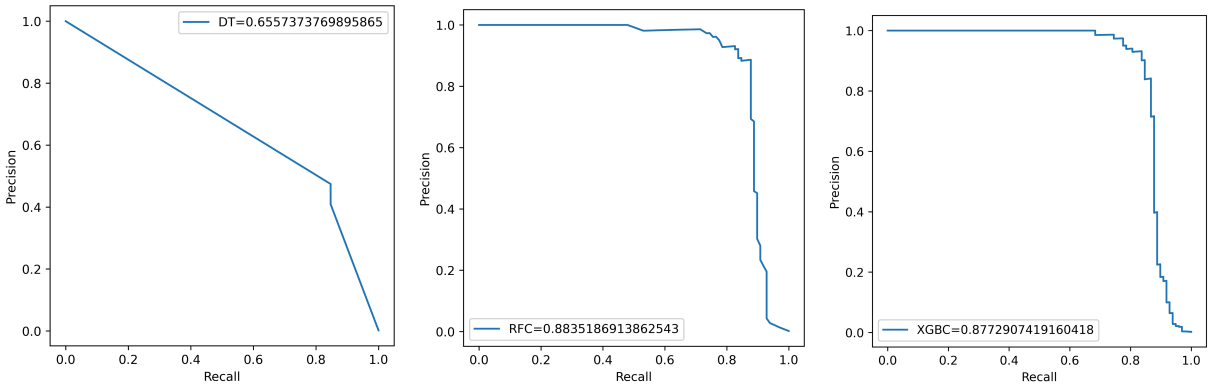


(a) **(b)** **(c)**

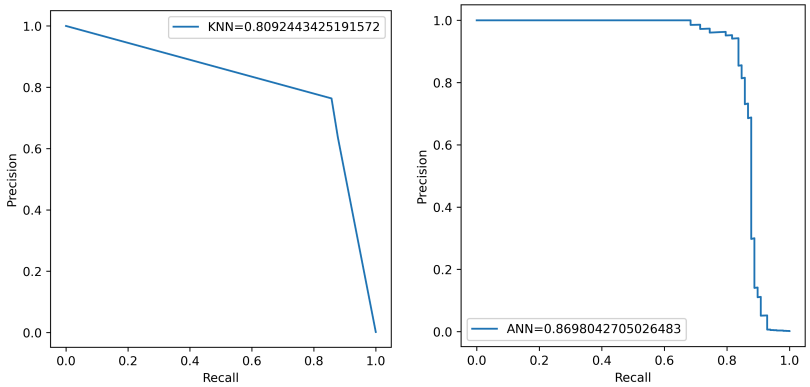


(d) **(e)**

Fig. 4.3 ROC curves on (a)DT (b) RF (c) KNN (d)ANN (e)XGBC for Feature Selection



(a) **(b)** **(c)**



(d) **(e)**

Fig. 4.4 PR Curve s on (a)DT (b) RF (c)XGBC (d)KNN (e)ANN for Feature Selection

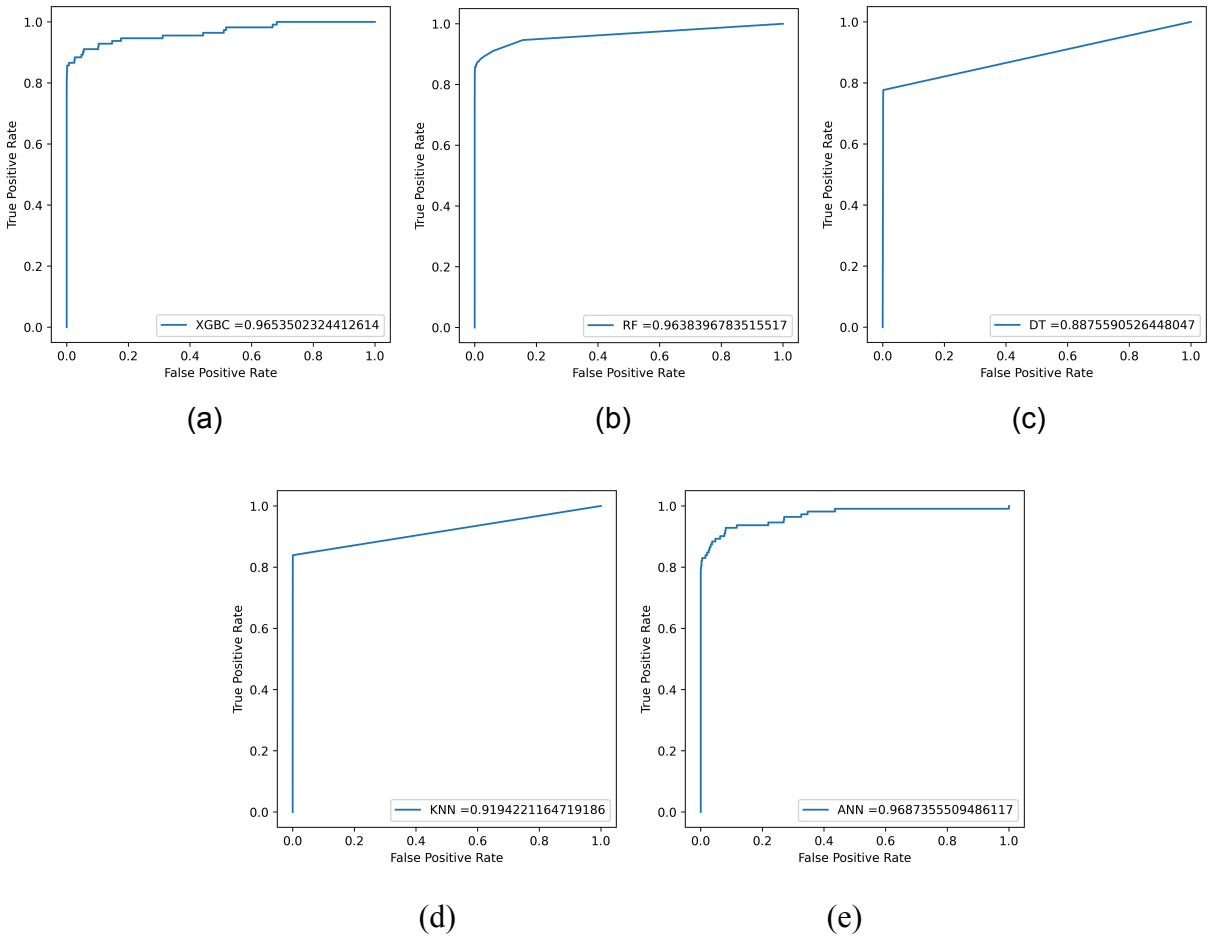
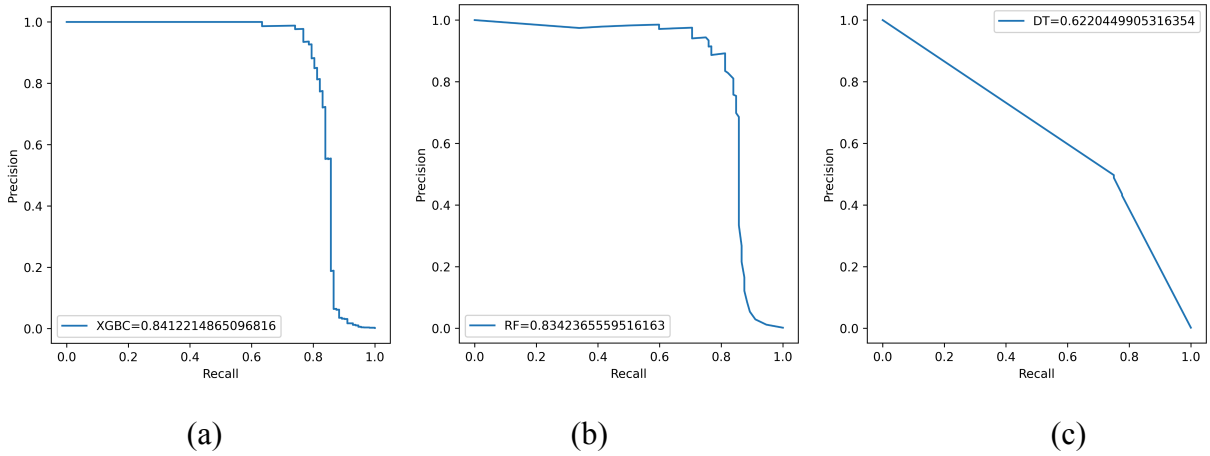
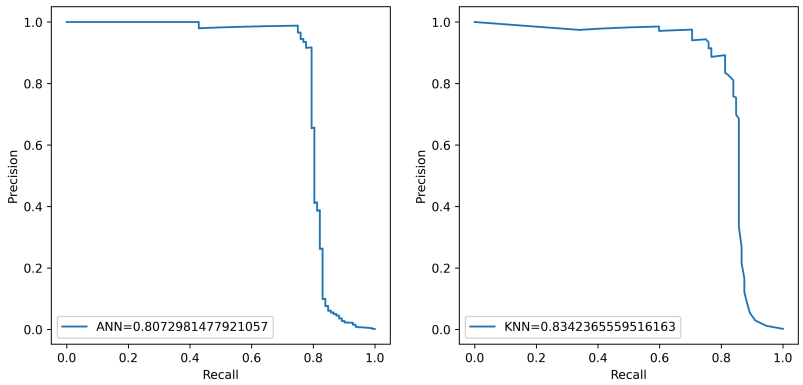


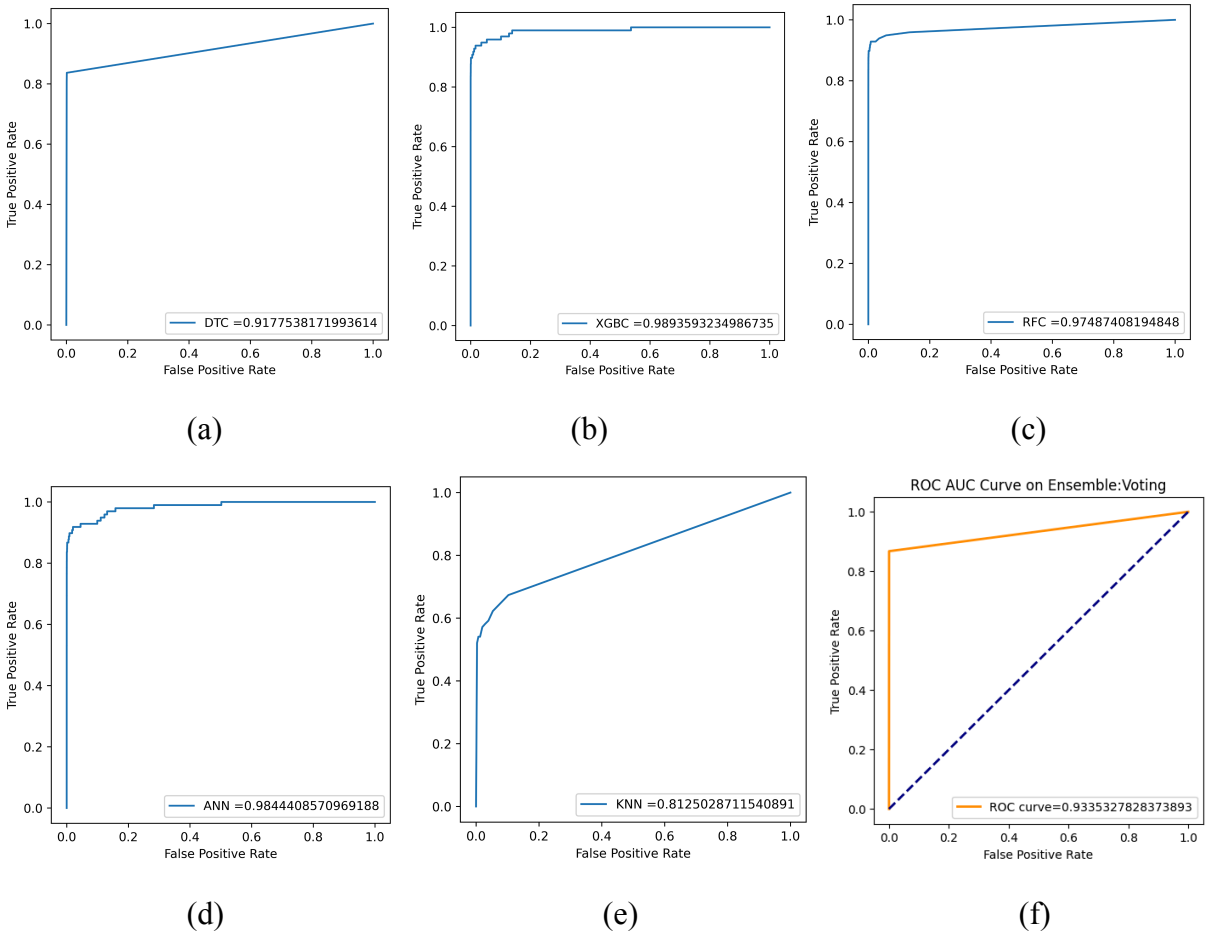
Fig. 4.5 ROC curves on (a)XGBC (b) RF (c)DT (d)KNN (e)ANN for PCA





(d) (e)

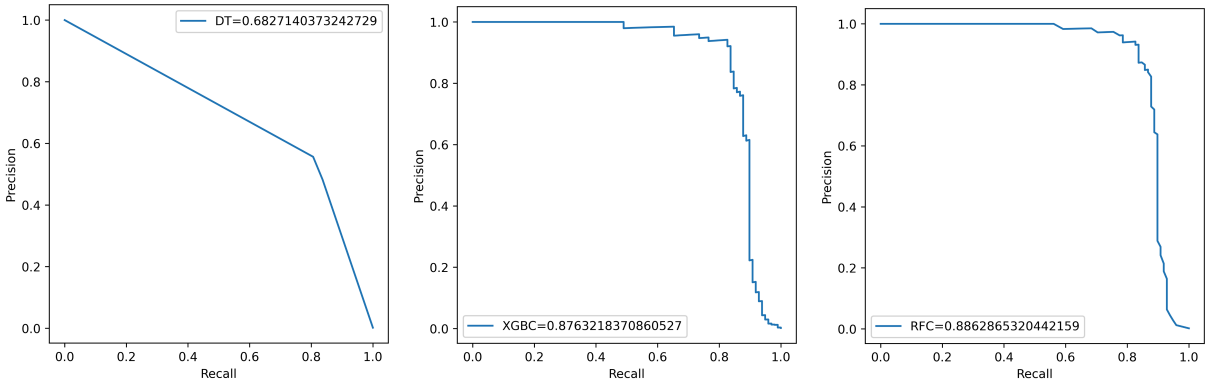
Fig. 4.6 PR curves on (a)XGBC (b) RF (c) DT (d)ANN (e)KNN for PCA



(a) (b) (c)

(d) (e) (f)

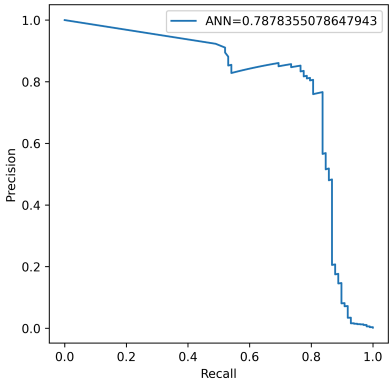
Fig. 4.7 ROC curves on (a)DT (b) XGBC(c) RF(d)ANN (e)KNN (f) ensemble for Ensemble voting



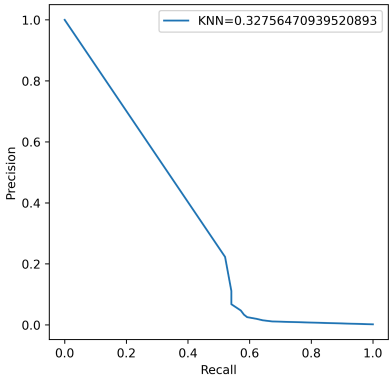
(a)

(b)

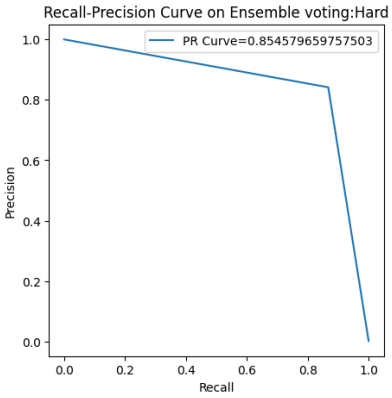
(c)



(d)

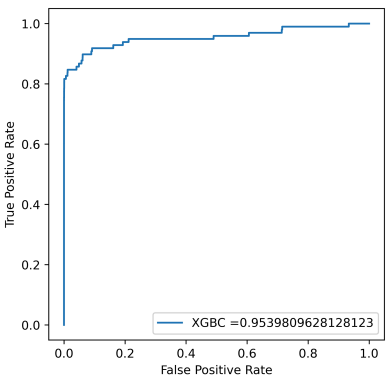


(e)

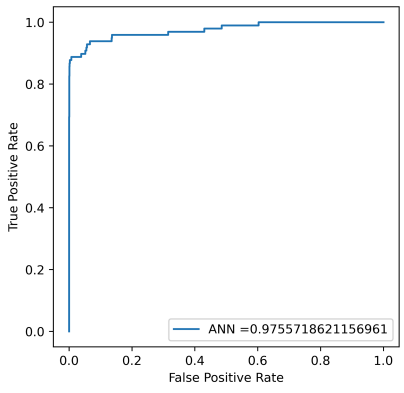


(f)

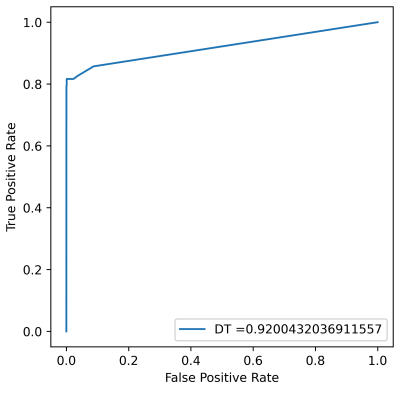
Fig. 4.8 PR curves on (a)DT (b) XGBC (c) RF (d)ANN (e)KNN (f) EL for Ensemble majority voting



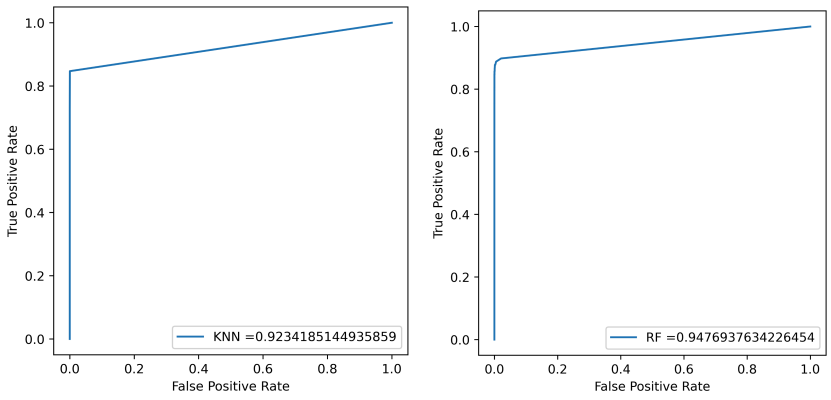
(a)



(b)

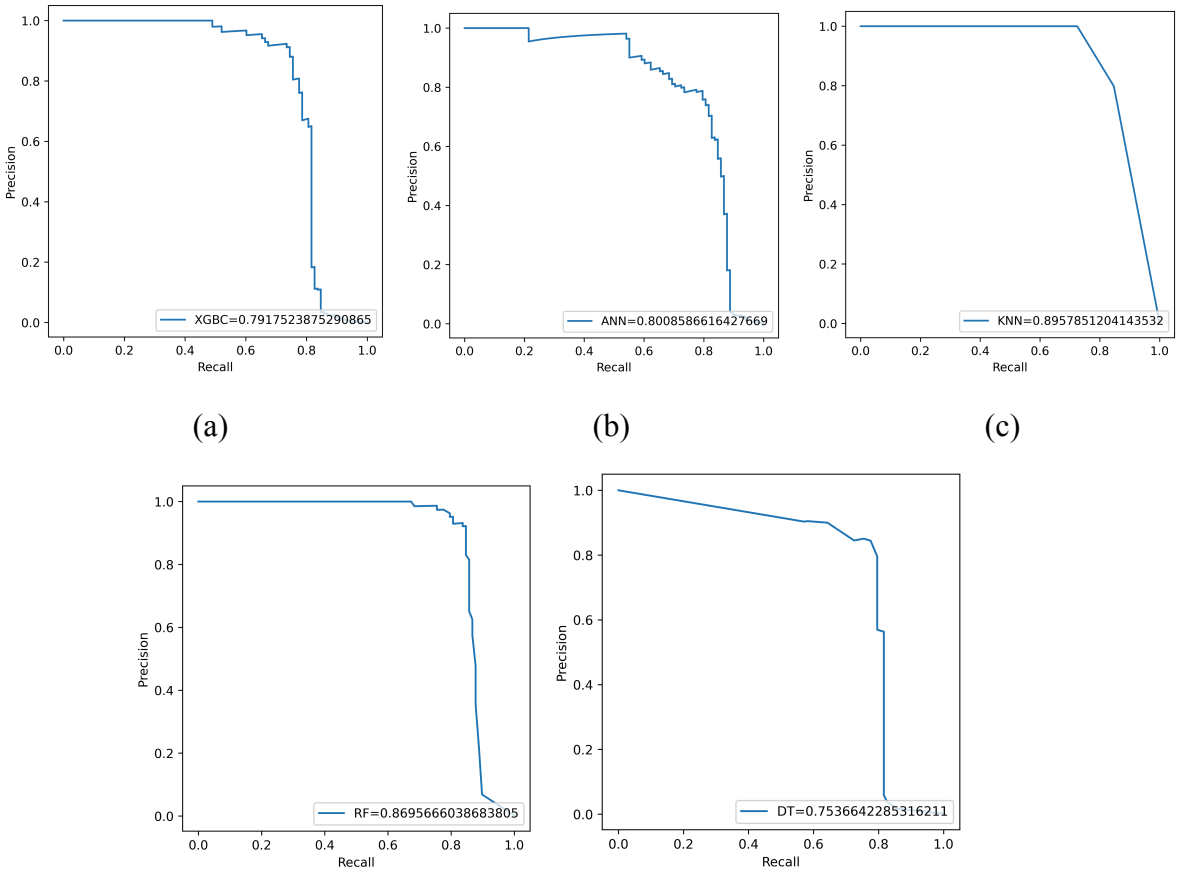


(c)



(d) (e)

Fig. 4.9 ROC curves on (a) XGBC (b) ANN (c) DT (d) KNN (e) RF for GA



(a) (b) (c)

(d) (e)

Fig. 4. 10 PR curves on (a) XGBC (b) ANN (c) KNN (d) RF (e) DT for GA

When comparing the feature selection methodologies with another three methodologies (PCA, Ensemble Voting and Genetic Algorithm) across the five machine learning models (Random Forest, Decision Tree, XGBoost, KNN, and ANN), Feature Selection with Random Forest (RF) emerges as the best choice based on several key metrics: accuracy, F1 score, and computational time shows in table 2 and Fig. 4.3 - Fig. 4.10.

In terms of accuracy, Random Forest under the Feature Selection methodology achieves an impressive 99.95%, slightly higher than XGBoost (99.94%) and Genetic Algorithm with Random Forest (99.94%), but still very competitive. Decision Tree (99.92%) and KNN (99.92%) also perform well, but Random Forest maintains a more balanced performance across multiple metrics.

For the F1 score, Random Forest under Feature Selection achieves 88.20%, outperforming Decision Tree (58.86%) and ANN, which struggles with a low F1 score. Although XGBoost achieves a higher F1 score (85.42%), Random Forest still demonstrates a balanced performance, especially when combined with its high accuracy and speed. While KNN shows an F1 score of 80.76%, it lags behind Random Forest in terms of computation time.

In terms of computation time, Random Forest under Feature Selection is by far the fastest, with a computation time of just 0.35 sec. far outperforming KNN (17.65 sec.), and the much slower Ensemble Voting (314.31 sec). This makes Random Forest ideal for real-time fraud detection applications where speed is crucial.

While AUC and PR values are important for measuring a model's ability to distinguish between classes and handle imbalanced datasets, Random Forest under Feature Selection provides a solid performance in both. Although it does not achieve the highest AUC (with XGBoost scoring 97.95% in AUC), Random Forest still delivers a competitive AUC of 97.95% and PR score of 88.35% which is the highest among all of the methodologies all models, that shows its effectiveness in handling fraud detection, especially when combined with its faster runtime and reliable accuracy.

In conclusion, while models like XGBoost may excel in certain metrics such as F1 score and AUC, Random Forest under Feature Selection provides the best balance of high accuracy, F1

score, fast computation time, AUC, and PR values. This makes it the most effective methodology for fraud detection, particularly when time and efficiency are crucial.

Table-4: Comparison of our proposed model's accuracy and F1 score with those of other novel, current models.

Reference Paper	**Model**	**No. of features**	**Performance**	
			Accuracy	**F1 score**
Ileberi, E. et al. (2022) [45]	GA-RF	9	99.93%	79.26%
Ileberi, E. et al. (2022) [45]	GA-RF	18	99.94%	82.85%
Ileberi, E. et al. (2022) [45]	GA-NB	18	98.13%	12.65%
Ileberi, E. et al. (2022) [45]	GA-LR	13	99.77%	39.84%
Ileberi, E. et al. (2022) [45]	GA-ANN	13	99.08%	21.20%
Sohony, I. et al. (2018) [46]	Ensemble voting	30	**99.95%**	86.29%*
Varmedja, D. et al. (2019) [47]	LR	27	97.46%	11.05%*
Varmedja, D. et al. (2019) [47]	ANN	27	99.93%	80.40%*
Varmedja, D. et al. (2019) [47]	NB	27	99.23%	27.04%*
Yılmaz, A. A. (2023) [48]	PSO-RF	30	99.92%	82.63%
Yılmaz, A. A. (2023) [48]	PSO-NB	30	99.55%	29.75%
Yılmaz, A. A. (2023) [48]	PSO-LR	30	99.74%	46.96%
Yılmaz, A. A. (2023) [48]	PSO-DT	30	99.81%	57.45%
Khare, N., & Sait, S. Y. (2018). [49]	RF	30	98.6%	-
Khare, N., & Sait, S. Y. (2018). [49]	DT	30	95.5%	-
Khare, N., & Sait, S. Y. (2018). [49]	LR	30	97.7%	-
Our approach	**Feature Selection, RF**	**14**	**99.95%**	**88.20%**

In Table-3, we compared the existing models with our suggested method based on the accuracy and F1 score. The F1 score gives equal weight to both precision and recall, and is measured by calculating the harmonic mean of both precision and recall. Since only using accuracy to determine a model's correctness is not optimal, we have also utilized the F1 score. Our model achieves a high accuracy of 99.95% and an F1 score of 88.20%, outperforming the models in references [45,46,48,49] using the same dataset. Specifically, against GA-RF and GA-NB with 18 features in [45], our model shows a 0.01% and 1.82% increase in accuracy, respectively, and significantly higher F1 scores by 5.35% and 75.55%. When compared to models with 9 features in [45], our method improves accuracy by 0.02% and F1 score by 8.94%. For GA-LR and GA-ANN with 13 features in the same paper, our model's accuracy is 0.18% and 0.87% higher, respectively, with a much greater F1 score difference of 48.36% and 67%. Additionally, our model's F1 score exceeds the Ensemble Voting model in [46] by 1.91%, with a similar accuracy score. Compared to all 3 models (RF, DT, and LR) that include all 30 features in the paper [49], our proposed model demonstrates an accuracy increase of up to 4.45%. Finally, our method is better than the PSO-RF, PSO-NB, PSO-LR, and PSO-DT models in [48], showing that our feature selection and model ensemble strategies work well by increasing accuracy by up to 0.4% and getting a much higher F1 score.

Our proposed model demonstrates superior performance compared to existing models across multiple reference papers, both in terms of accuracy and F1 score. The comprehensive analysis shows that our model not only achieves high accuracy (99.95%) but also excels in F1 score (88.20%), which indicates a balanced performance in precision and recall. This is particularly notable given that our model uses fewer features (14) compared to other models, which often employ a larger feature set.

4.4 Model-agnostics XAI method

In this section we explain how model-agnostic explainability methods in Explainable AI (XAI) are applied to any machine learning model to provide insights into its predictions. It helps to interpret easily black box models. We use the SHAP (SHapley Additive exPlanations) method. This method assign SHAP values to each features of the dataset that are applied to each model. The contribution values are derived from cooperative game theory and provide a unified

measure of feature importance. SHAP values of each features say how importance the feature is to contribute model prediction.

4.4.1 SHAP on different models.

The Fig. 4.11 shows a SHAP summary plot applied to a random forest algorithm. This visualization helps in understanding the contribution and impact of various features on the model's predictions. Here x-axis represents the SHAP values those are says how much a feature contributes to a pecific prediction. Larger SHAP values indicates stronger influence and lower values implying less impact on the prediction. Here V14, V12, and V4 are among the most important features in this model, showing a wide spread of SHAP value. V14 high values indicate high fraud likelihood of fraudulent transaction. V12 feature's lower values push the model towards predicting fraud, while higher values reduce the fraud likelihood. V4, V10, and V17 features also show a mix of high and low values contributing differently to the fraud prediction.

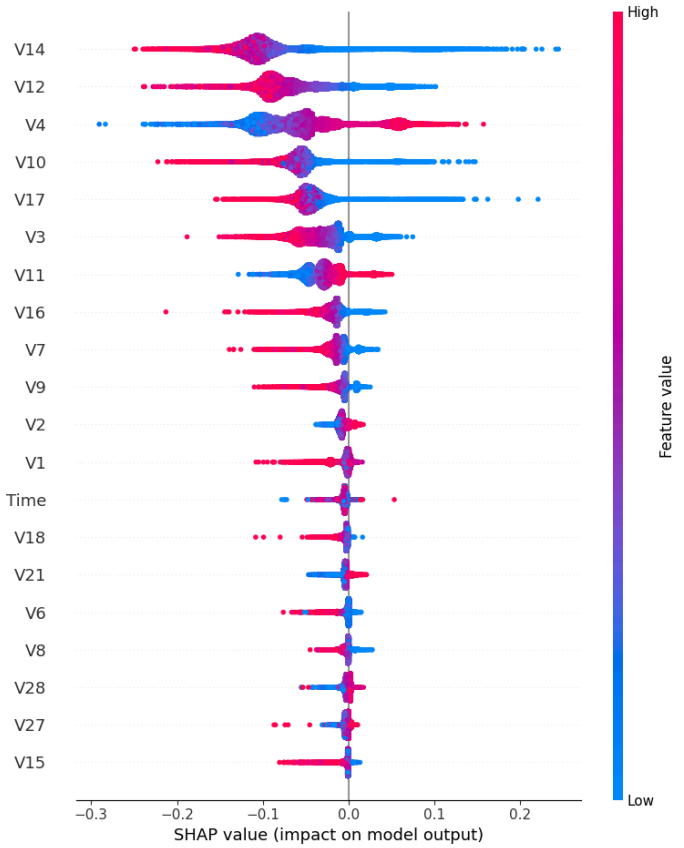


Fig. 4.11 XAI SHAP on RF

The Fig. 4.12 depicts the SHAP plot for XGBC where the figure V14 has most importance to predict fraud transaction. V4 higher values predict the model positively but lower values predict result as non fraud.

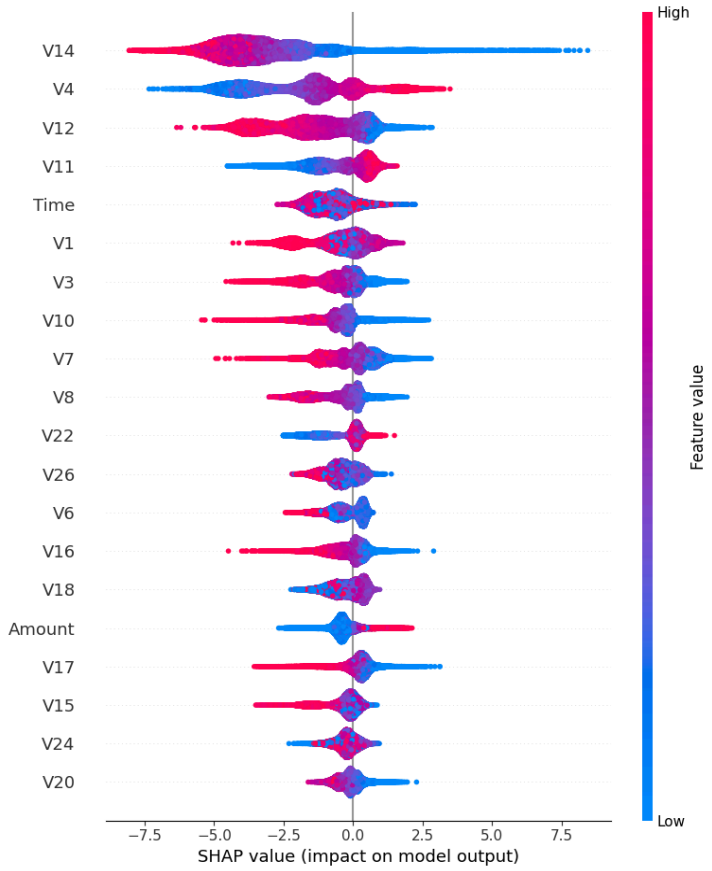


Fig. 4.12 XAI SHAP on XGBC

In the Fig. 4.13 V14, V12, and V4 are the most significant features, indicating they have the highest impact on determining whether a transaction is fraudulent or not. V14 is particularly important, with higher values reducing the likelihood of fraud, as seen by the red points on the left side. Features like Amount and V10 also influence the model, with varying effects depending on their values. The lower-ranked features, such as V19 and V1, have minimal impact on the model's predictions, contributing only slightly to the final output.

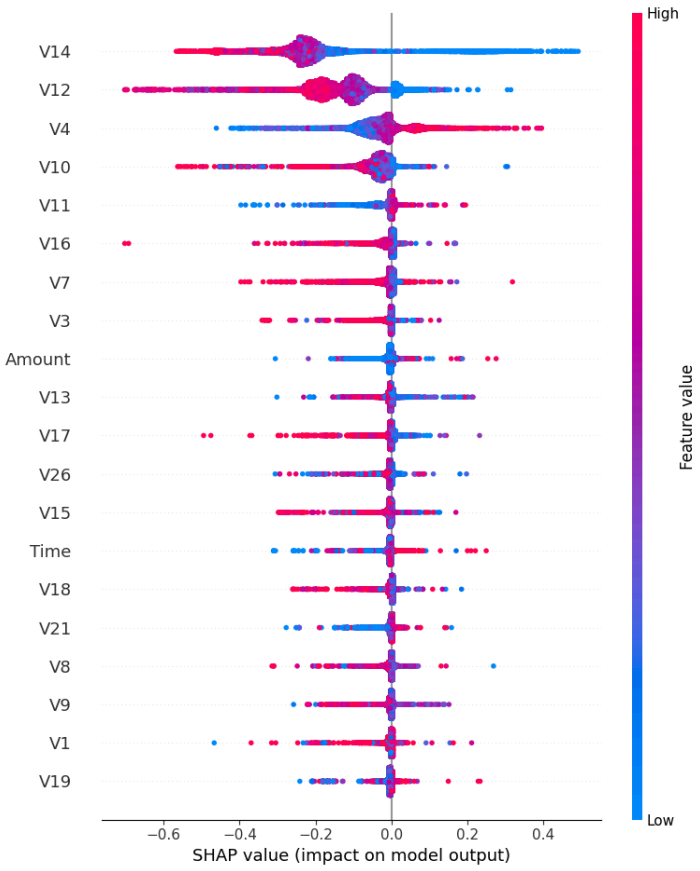


Fig. 4.13 XAI SHAP on DT

The SHAP analysis on Random Forest (RF), Decision Tree (DT), and XGBoost Classifier (XGBC) demonstrates that the most important features are positioned at the top, indicating that our selected 14 important features for hybrid feature selection are equally significant for fraud detection, as illustrated in the figure. These selected features play a crucial role in enhancing the effectiveness of fraud detection systems. By providing relevant and informative data points, they enable machine learning models to better differentiate between legitimate and fraudulent transactions. Their inclusion not only improves the model's accuracy but also reduces false positives, thereby increasing the overall reliability of fraud detection efforts. Additionally, utilizing Explainable Artificial Intelligence (XAI) techniques to identify these features ensures that the decision-making processes of the models are transparent and interpretable, fostering greater trust among users and stakeholders in the system's capability to combat fraud.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

As usage of credit cards become more and more common in every field of our daily life, credit card fraud has become much more protrusive [22]. Our research work was accomplished with the purpose of comparing some proactive developed models using five supervised machine learning algorithms with four different techniques that make the overall result as to how accurately they differentiate and classify the fraud and non-fraud transactions of the credit card dataset with SMOTE and some enhanced feature selection techniques.

These are the observations we obtained from the above experiments. Random Forest (RF) with Feature Selection emerges as the best-performing methodology across accuracy, F1 score, and computation time, with 99.95%, 88.20%, and 0.35 seconds, respectively. In this methodology, we identified 14 important features, which were validated as significant for detecting fraud by XAI (Explainable AI). For the PCA technique, we selected 26 important features. In this methodology, RF also performed well with a 99.93% accuracy score, which is the same for both KNN and ANN. However, RF outperformed in terms of F1 score, achieving 83.41.

Ensemble learning is a slower process because it uses 5 ML models simultaneously, yielding an accuracy score of 99.95% but requiring 314.31 seconds for testing. Our final methodology is the well-known Genetic Algorithm (GA), where RF, XGBC, and DT achieved a higher accuracy score of 99.95%. However, this process is also slow, and as a result, it was not selected for fraud detection.

Feature Selection significantly enhances RF's performance across all metrics, demonstrating its effectiveness in identifying the most important features. Overall, RF with Feature Selection is the most efficient and reliable model for real-time fraud detection, offering both speed and accuracy. Finally, we demonstrate how the XAI concept helps interpret the results of machine learning models in a way that is easily understandable by humans.

5.2 Limitation

The dataset is significantly imbalanced, and to address this, we apply the popular SMOTE technique to improve balance. However, this method is not without limitations, as the generated synthetic samples don't always perfectly reflect the characteristics of the original data. While

advanced models such as Random Forest, XGBoost, and Neural Networks deliver high accuracy, they often lack transparency. A high rate of false positives (incorrectly classifying legitimate transactions as fraudulent) can result in customer frustration and diminished trust in financial institutions. Additionally, the study is based on data from European cardholders in 2013, which may not encompass all fraud patterns.

5.3 Future Work

For future work, we aim to explore Federated Learning, an emerging area in Artificial Intelligence (AI) that has opened up new possibilities in Machine Learning. Federated Learning allows for the training of machine learning models across multiple decentralized devices or servers while preserving data privacy, which is particularly important in sensitive fields like financial fraud detection.

By leveraging Federated Learning, we plan to develop models using traditional machine learning algorithms like Random Forest, XGBoost, and Neural Networks, while also investigating newer algorithms specifically designed for decentralized learning environments. Our goal will be to enhance the performance of these models, evaluating them using a comprehensive set of metrics, including accuracy, precision, recall, F1 score, and computational efficiency.

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/ijarsct-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] SamanehSorournejad, 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

[36] K. Raza, *Improving the prediction accuracy of heart disease with ensemble learning and majority voting rule*. Elsevier Inc., 2018. doi: 10.1016/B978-0-12-815370-3.00008-6.

[37] I. Sohony, R. Pratap, and U. Nambiar, “Ensemble learning for credit card fraud detection,” *ACM Int. Conf. Proceeding Ser.*, pp. 289–294, 2018, doi: 10.1145/3152494.3156815.

[38] A. Lambora, K. Gupta, and K. Chopra, “Genetic Algorithm- A Literature Review,” *Proc. Int. Conf. Mach. Learn. Big Data, Cloud Parallel Comput. Trends, Prespectives Prospect. Com. 2019*, no. 1998, pp. 380–384, 2019, doi: 10.1109/COMITCon.2019.8862255.

[39] K. Pal and B. V. Patel, “Data Classification with k-fold Cross Validation and Holdout Accuracy Estimation Methods with 5 Different Machine Learning Techniques,” *Proc. 4th Int. Conf. Comput. Methodol. Commun. ICCMC 2020*, no. Iccmc, pp. 83–87, 2020, doi: 10.1109/ICCMC48092.2020.ICCMC-00016.

[40] Holland, J.H. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*; MIT Press: Cambridge, MA, USA, 1975.

[41] Alkafaween, E.O. Novel Methods for Enhancing the Performance of Genetic Algorithms. Master’s Thesis, Mu’tah University, Karak, Jordan, 2015.]

[42] Computer-Aided Gas Pipeline Operation Using Genetic Algorithms And Rule Learning, Ph.D. thesis. D. Goldberg. s.l. : University of Michigan, Ann Arbor, 1983.

[43] Raza, K. (2019). Improving the prediction accuracy of heart disease with ensemble

learning and majority voting rule. In *U-Healthcare Monitoring Systems* (pp. 179-196). Academic Press.

[44] D. Elreedy and A. F. Atiya, “A Comprehensive Analysis of Synthetic Minority Oversampling Technique (SMOTE) for handling class imbalance,” Inf. Sci. (Ny)., vol. 505, pp. 32–64, 2019, doi: 10.1016/j.ins.2019.07.070.

[45] Ileberi, E., Sun, Y., & Wang, Z. (2022). A machine learning based credit card fraud detection using the GA algorithm for feature selection. *Journal of Big Data*, *9*(1), 24.

[46] Sohony, I., Pratap, R., & Nambiar, U. (2018, January). Ensemble learning for credit card fraud detection. In *Proceedings of the ACM India joint international conference on data science and management of data* (pp. 289-294).

[47] Varmedja, D., Karanovic, M., Sladojevic, S., Arsenovic, M., & Anderla, A. (2019, March). Credit card fraud detection-machine learning methods. In *2019 18th International Symposium INFOTEH-JAHORINA (INFOTEH)* (pp. 1-5). IEEE.

[48] Yılmaz, A. A. (2023). A machine learning-based framework using the particle swarm optimization algorithm for credit card fraud detection. *Communications Faculty of Sciences University of Ankara Series A2-A3 Physical Sciences and Engineering*, *66*(1), 82-94.

[49] Khare, N., & Sait, S. Y. (2018). Credit card fraud detection using machine learning models and collating machine learning models. *International Journal of Pure and Applied Mathematics*, *118*(20), 825-838.

[50] Md Rokibul Hasan, Md Sumon Gazi, and Nisha Gurung, “Explainable AI in Credit Card Fraud Detection: Interpretable Models and Transparent Decision-making for Enhanced Trust and Compliance in the USA,” J. Comput. Sci. Technol. Stud., vol. 6, no. 2, pp. 01–12, 2024, doi: 10.32996/jcsts.2024.6.2.1.

[51] A. Thennakoon, C. Bhagyani, S. Premadasa, S. Mihiranga and N. Kuruwitaarachchi, "Real-time Credit Card Fraud Detection Using Machine Learning," *2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, Noida, India, 2019, pp. 488-493, doi: 10.1109/CONFLUENCE.2019.8776942.

[52] E. M. Learning, “Ensemble Machine Learning,” Ensemble Mach. Learn., 2012, doi: 10.1007/978-1-4419-9326-7.

[53] M. Rivki, A. M. Bachtiar, T. Informatika, F. Teknik, and U. K. Indonesia, “Ensemble

Learning," vol. 1, no. 112.

[54] Polikar, R. (2012). Ensemble Learning. In: Zhang, C., Ma, Y. (eds) Ensemble Machine Learning. Springer, New York, NY. https://doi.org/10.1007/978-1-4419-9326-7_1