INDIAN JOURNAL OF SCIENCE AND TECHNOLOGY

ORIGINAL ARTICLE

# Contrastive Study of Machine Learning Techniques for Credit Card Fraud Detection

**Dhwanir Shah**[1]*, **Lokesh Kumar Sharma**[2]

**1** Department of Computer Science, Gujarat University, Navrangpura, Ahmedabad–9, Gujarat, India
**2** National Institute of Occupational Health, Ahmedabad, Gujarat, India

OPEN ACCESS

**Received:** 27/03/2025
**Accepted:** 14/04/2025
**Published:** 06/05/2025

**Citation:** Shah D, Sharma LK (2025) Contrastive Study of Machine Learning Techniques for Credit Card Fraud Detection. Indian Journal of Science and Technology 18(16): 1248-1259. https://doi.org/10.17485/IJST/v18i16.572

* **Corresponding author**.

dhwanir12may1982@gmail.com

**Funding:** None

**Competing Interests:** None



Published By Indian Society for Education and Environment (iSee)

**ISSN**
Print: 0974-6846
Electronic: 0974-5645

## Abstract

Objective: To assess the efficacy of five machine learning algorithms—Logistic Regression, Support Vector Machine (SVM), Decision Tree, Random Forest, and XGBoost—in detecting credit card fraud, utilizing a simulated Kaggle dataset created through Sparkov for credit card transactions. Methods: The dataset was partitioned into three training-test ratios: 60%:40%, 70%:30%, and 80%:20%. To address class imbalance, the Synthetic Minority Over-sampling Technique (SMOTE) was employed. The performance of the models was measured using Accuracy, Precision, Recall, F1-score, and ROC-AUC, with validation conducted through 10-fold cross-validation. Findings: XGBoost consistently surpassed the other models, achieving precision and recall rates of up to 99.05% and 99.81%, respectively. The Decision Tree and Random Forest models produced precision scores of 96.86% and 94.65%, with recall values of 99.08% and 99.49%. Logistic Regression and SVM exhibited comparatively lower performance across all training-test splits. Novelty: This study introduces a unique methodology by evaluating machine learning algorithms across different training-test splits and validating the outcomes through 10-fold cross-validation to ensure the robustness of the models.

**Keywords:** Credit card Fraud; Logistic Regression; Support Vector Machine; Decision Tree; Random Forest; XGBoost; K-Fold cross validation; OneHotEncoding

## 1 Introduction

Technological innovation and cyberspace use have made online shopping Smooth, with credit cards enabling millions of fast, convenient transactions daily worldwide. Analysing and categorizing transaction patterns enables the detection of fraudulent activities.

In [(1)], outlined the essential aspects of credit cards and the various types of fraud that can occur. They have also reviewed several machine learning algorithms such as Naive Bayes, Logistic Regression, SVM, Decision Trees, Random Forest, AdaBoost, XGBoost, etc. which can be employed to identify credit card fraud. Detecting credit card fraud continues to pose significant challenges due to the rare occurrence of

fraudulent activities and the constantly changing tactics employed by fraudsters. Many current studies do not effectively address these challenges; they often place excessive emphasis on accuracy as the primary evaluation metric, neglect class imbalance, utilize limited or no cross-validation, and fail to consider essential features such as geographic distance or contextual factors. These shortcomings diminish the effectiveness of fraud detection models in practical scenarios. To overcome these issues, this research employs SMOTE for class balancing, HalvingRandomSearchCV for efficient hyperparameter optimization, and 10-fold cross-validation for a more dependable performance assessment. Additionally, it incorporates distance-based features to enhance the understanding of transaction behaviour. In contrast to earlier methods, this study emphasizes recall, precision, F1-score, and ROC-AUC, thereby creating a more balanced and resilient fraud detection model.

Table 1 shows Methodology summary of previous research.

**Table 1.** Methodology summary

| Reference | Dataset Used | Approach | Key Contributions | Limitations |
|---|---|---|---|---|
| [2] | Kaggle (European cardholder dataset) | Local Outlier FactorIsolation Forest | • The algorithm is achieving 99.6% accuracy and 28% precision for a tenth of the dataset and on full dataset 33% precision occurs. | • Low Precision Rate<br>• The focus is mainly on accuracy and precision metrics<br>• Data distribution information and data balancing technique are missing.<br>• Cross validation technique and Hyper parameter tuning are missing |
| [3] | Kaggle (European cardholder dataset) | Naïve Bayes, Logistic Regression, Random Forest, AdaBoost | • Shown the increased accuracy of detecting fraudulent transactions and minimizing the number of false alerts<br>• Random Sampling method has been used<br>• Data is distributed in 70%:30% ratio only | • Normalization technique is missing<br>• Cross validation technique is missing<br>• No specification for parameter tuning approach<br>• Work is done only on the samples of size to 560 transactions only. |
| [4] | Kaggle (European cardholder dataset) | Logistic Regression,Naive Bayes,Decision Tree,Random Forest,Artificial Neural Network | • User interface is made with Python Tkinter module<br>• Data is normalized and standardized. | • The algorithm performance is evaluated using accuracy, precision and recall only.<br>• No specification for which oversampling technique has been used.<br>• Hyper parameter tuning is missing<br>• Cross validation Technique is missing<br>• Data distribution information is missing. |
| [5] | Kaggle (European cardholder dataset) | Logistic Regression, Random Forest, Naive Bayes, Multilayer Perceptron | • Feature selector tool by Will Koehrsen was used.<br>• SMOTE and scaling techniques have been used.<br>• Train-test split:80%:20% only | • The algorithm performance is evaluated using accuracy, precision and recall only<br>• Hyper parameter tuning is missing<br>• Cross validation technique is missing |

*Continued on next page*

*Table 1 continued*

| | | | | |
|---|---|---|---|---|
| [6] | Kaggle (European cardholder dataset) | • Naïve Bayes, Support vector machine, Logistic regression, Random Forest, Decision tree, and XGBoost | • Hyper parameter tuning is done with a few parameters<br>• Precision, recall, accuracy, AUC, and F-measure are used to evaluate the model performance<br>• Data is normalized and standardized. | • Data is distributed in 70%:30% ratio only<br>• No specification for data balancing concept<br>• Cross validation technique is missing. |
| [7] | Kaggle (Predicting Churn for Bank Customers) | K-Neighbours, Random Forest, XGboost, Adaboost classifiers and Ensemble Model (Stacking Technique) | • Different oversampling techniques have been used.<br>• Cross validation K-fold where k=5 is used.<br>• Label encoding and normalization have been done.<br>• Metrics used: Accuracy, precision, recall and f1-score | • Not clear specification regarding in which ratio data is divided into train and test<br>• Hyper parameter tuning is missing. |
| [8] | Kaggle (European cardholder dataset) | SVM, Random Forest, Bagging, XGBoost, and DT | • Precision, F1-score, accuracy, and recall are used for performance evaluation.<br>• Train-test split:80%:20% only<br>• SMOTE has been used for imbalance data | • No specification for Normalization and standardization of data<br>• Parameter tuning is also missing<br>• cross validation technique is missing |
| [9] | Kaggle (Credit Card Fraud Dataset, includes Western United states) | Naïve Bayes, Support vector machine, Logistic regression, Random Forest, Decision tree, KNN and XGBoost | • Accuracy, precision, recall and f1-score are used for performance evaluation.<br>• Data is distributed in 70%:30% ratio only | • Not shown exact scaling process<br>• Hyper parameter tuning is missing and no parameter description is given<br>• cross validation technique implementation is missing |
| [10] | Although Not Specified but from feature it seems taken from Kaggle (European dataset) | Random Forest | • achieved Accuracy of 99.93%, precision of 99.78% and recall of 99.85%.<br>• Data is distributed in 70%:30% ratio only<br>• Normalization is applied | • Cross validation technique is missing.<br>• No specification for data balancing technique.<br>• Hyper parameter tuning is missing. |

In [11] addressed the limitations of rule-based systems and underscored the significant importance of machine learning in the realm of fraud detection. The research involved a comparison of various models, scrutinizing their strengths and weaknesses through metrics such as accuracy, precision, recall, F1-score, and ROC curves. The analysis concluded that XGBoost was the most effective model among those evaluated.

In [12] identified that the SVM algorithm utilized in the current system exhibited lower accuracy and slower performance. They assessed the model's effectiveness through metrics such as accuracy, precision, recall, classification report, and confusion matrix. Their findings indicated that conventional security methods are inadequate for fraud detection, and they discovered that neural networks yielded superior accuracy.

In [13] implemented the SMOTE technique to achieve balance within the dataset and employed feature selection to pinpoint essential features. Additionally, they utilized undersampling, oversampling, and hybrid approaches to address class imbalance. The performance of the model was assessed through metrics such as accuracy, recall, precision, F1-score, and specificity. The MLP model demonstrated remarkable performance, attaining an F1-score of 0.9998 and an accuracy rate of 99.95%, indicating its exceptional capability in identifying credit card fraud. In [14] carried out MinMax normalization and analyzed the data using R. The model was refined through either k-fold cross-validation or basic cross-validation. The Area Under the Precision-Recall Curve (AUPRC) served as the key performance metric. To tackle data imbalance, a combination of undersampling, oversampling, and hybrid techniques was utilized. The study concluded that effectively tuned oversampling and the creation

of synthetic data led to improved predictive performance, with Random Forest proving to be a strong and reliable method in managing class imbalance.

In[15] used the SMOTE technique to handle class imbalance and found XGBoost to be the most effective model, achieving 99.88% accuracy in fraud detection. The authors suggested combining machine learning with explainable AI (XAI). Model performance was evaluated using recall, accuracy, precision, F1-score, and ROC-AUC, and results were validated using 10-fold cross-validation. According to Samidha Khatri, Aishwarya Arora, and Arun Prakash Agrawal[16], accuracy is not a suitable measure for datasets with imbalances. They analyzed different models by focusing on sensitivity, precision, and processing time. At a threshold of 0.4, the Logistic Regression (LR), Naive Bayes (NB), and Random Forest (RF) models showed improvements in both sensitivity and precision. Although k-Nearest Neighbors (kNN) achieved higher sensitivity than the Decision Tree, it required more time for testing. As a result, the Decision Tree was identified as the most effective model, balancing performance with efficiency.

As noted by Xinman Wang, essential components in detecting fraud encompass the ratio of the purchase price to the median price and the distance from the cardholder's home address. To facilitate this analysis, standardization was implemented to adjust the scale of features, and oversampling was applied to achieve a balanced dataset. Feature importance was determined using the Random Forest algorithm, and a correlation heatmap was utilized to analyze the interrelationships between features. The Random Forest and Decision Tree models demonstrated superior performance among the various models tested. Model evaluation was conducted through the use of a confusion matrix, along with metrics such as accuracy, F1-score, recall, and precision.[17]

In[18] presented Compact Data Learning (CDL) as a method to minimize both the size and complexity of datasets without compromising accuracy. They utilized a dataset of European cardholders and implemented five different models: Random Forest with AdaBoost, Gradient Boosting, K-Nearest Neighbors (KNN), Convolutional Neural Networks (CNN), and Support Vector Machines (SVM). Through correlation analysis, CDL eliminated redundant features, ensuring that only the most pertinent ones were retained. The effectiveness of the models was assessed based on accuracy, precision, recall, and F1-score. In[19] assessed their model through various metrics, including accuracy, specificity, error rate, precision, recall, F1-score, ROC, and precision-recall curves. The findings indicated that the Random Forest model significantly surpassed the performance of Logistic Regression, K-NN, and VC Classifier.

In[20], introduced an ensemble model that integrates Random Forest, Logistic Regression, and AdaBoost through a soft voting approach to enhance fraud detection capabilities. By combining predictions from various models rather than depending on a single one, the method aims to improve accuracy and minimize false positives. Utilizing the PaySim and Kaggle credit card datasets, the model attained an impressive accuracy of 99.96%, precision of 99.53%, recall of 100%, an F1-score of 0.99, and an AUC of 1.0. The evaluation of performance was conducted using several metrics, and a comparison of training durations for different algorithms was also performed.

In[21], made a systematic review following the PRISMA framework to analyse credit card datasets, machine learning algorithms, and evaluation metrics. It emphasizes the issue of overlap, where fraudulent transactions can closely mimic legitimate ones. The review centers on the five leading studies that utilize datasets from Europe and Taiwan, employing models such as Deep Learning, Neural Networks, Ensemble Learning, and various Sampling techniques. The evaluation criteria mainly include recall, accuracy, AUC, precision, and F1-score.

Utilizing CNN, LSTM, and CNN-LSTM models on a dataset of transactions from Indonesia in March 2021, the study[22] applied SMOTE to tackle issues of data imbalance. Fraud detection was further enhanced through the use of outlier detection methods. The performance of the models was measured using a variety of metrics, including confusion matrix, accuracy, precision, recall, F1-score, and ROC-AUC. Among the models tested, CNN achieved the highest performance, with an accuracy of 99.88%, precision of 87.07%, recall of 100%, F1-score of 93.09%, and an AUC of 99.94%.

In[23], examined the effectiveness of Random Forest and Neural Network models in detecting fraud. The evaluation of performance was conducted using metrics such as accuracy, precision, recall, F1-score, confusion matrix, and classification report. Additionally, for Neural Networks, both loss and validation loss were taken into account. The Random Forest model achieved an impressive accuracy of 96%, demonstrating robust performance across classes: for class 0, it recorded 93% precision, 99% recall, and 96% F1-score; for class 1, it achieved 99% precision, 93% recall, and 96% F1-score. Both models proved to be effective in identifying fraudulent transactions.

In[24], presented a hybrid approach that merges Artificial Neural Networks (ANN) with Support Vector Machines (SVM) to improve the detection of fraudulent activities. The ANN is tasked with feature extraction and pattern recognition, while the SVM focuses on classification through the use of linear, RBF, and polynomial kernels. The model's performance is measured by evaluating accuracy, precision, recall, F1-score, and visualizing the confusion matrix.

### 1.1 Research Gap

The main deficiencies identified during research include:

**Excessive dependence on accuracy**: Accuracy can be a misleading metric in highly imbalanced datasets, as a model may record high accuracy by classifying the majority class it means non-fraudulent transactions while ignoring actual fraud transactions. This study minimizes this disadvantage by giving priority to Precision, Recall, F1-score, and ROC-AUC, which score offer more valuable insights into fraud detection.

**Inadequate Cross-Validation**: Certain algorithms are evaluated through a singular train-test partition, like a 70:30 or 80:20 split, which constrains their generalization potential. Adopting K-fold cross-validation, with k set to 10, yields a more accurate measure of performance.

**Lack of Hyperparameter Optimization**: A number of research neglect to fine-tune model parameters, which can make the performance weak. This research adopts HalvingRandomSearchCV for efficient parameter tuning.

**Inadequate Addressing of Class Imbalance**: Fraud detection datasets tend to be significantly unbalanced. Different studies are using different techniques for balancing imbalanced dataset like like Random Oversampling, Random Undersampling etc. This study applies SMOTE to achieve a more balanced dataset.

**Limited Feature Engineering**: Geographical elements are frequently neglected. This study improves fraud detection by creating distance-based features from both merchant and user location data.

This research utilizes multiple train-test ratios (60:40, 70:30, 80:20) to ensure robust evaluation. By using k-fold cross validation technique where k=10, we divided the dataset into ten segments and training the model ten times with various combinations to enhance its reliability. This study uses SMOTE in order to balance the dataset by generating synthetic samples for the minority class it means fraud cases. In this research we have used HalvingRandomSearchCV for hyperparameter Tuning to find the best model parameters in order to enhance the performance. As part of feature engineering, new meaningful features have been added to improve model predictions. By focusing more on precision, recall and F1-score it will ensure that the model will perform better, specifically in fraud detection, where accuracy alone is not sufficient.

## 2 Methodology

### 2.1 Data Analysis and Implementation of algorithms

The dataset has been collected from the Kaggle website[25]. In this paper, we have made an effort to present a variety of graphs that, will offer deeper insight into the data. The dataset, which spans the dates of January 1, 2019, through December 31, 2020, includes both legitimate and fraudulent credit card transactions. The Figure 1 illustrates how severely unbalanced the dataset is.

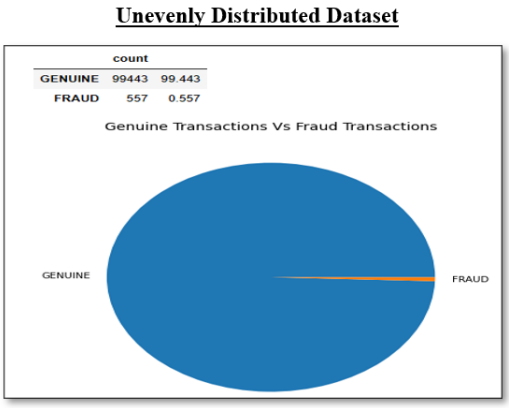


**Fig 1.** Imbalance Dataset

As part of data preprocessing, we have checked the dataset for duplicate records and null values. Whenever type casting was required, it has been used because it was necessary. In order to scale the data MinMaxScaler() has been used. For various categorical features OneHotEncoding technique is used. During the same stage label encoding technique has been used for city, state and merchant features.

The gender distribution of male and female cardholders for both fraudulent and legitimate transactions is depicted in the Figure 2. Additionally, both male and female cardholders experience an equal number of fraud transactions. The Figure 3 shows that the majority of fraud transactions occur between the hours of 21 and 4 in the morning, when legitimate card holders are sleeping.

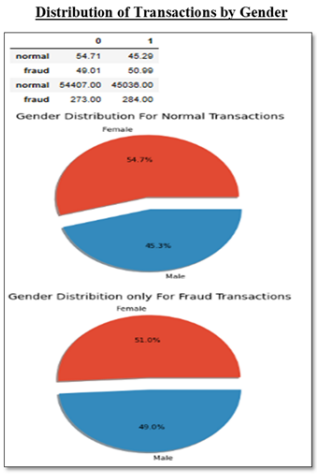


**Fig 2.** Gender-specific Transactions distribution

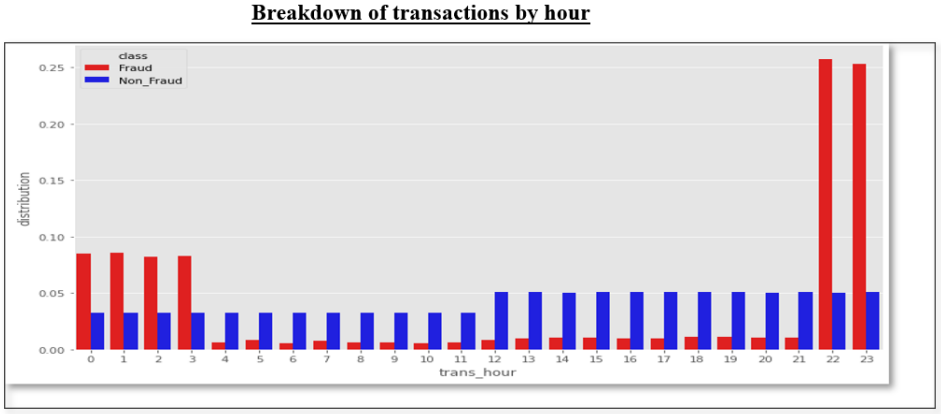


**Fig 3.** Distribution of transactions by hour

The Figure 4 (a) and (b) both show a few categories which are experiencing an increase in fraud transactions. The variable "is_fraud" has two possible values: 1 for transactions that are fraudulent and 0 for transactions that are not fraudulent.

The dataset is unbalanced, as we saw in the section previously. It has been noted that learning becomes challenging for ML algorithms when classification category data are not evenly distributed.[11] The amount of data makes parameter tuning more time consuming. SMOTE method is used to make the dataset balanced one. The Figure 5 shows balanced dataset after applying SMOTE technique to balance the imbalanced dataset. As you can see from Figure 1 before applying SMOTE genuine transactions were 99443 and fraud transactions were 557. After applying SMOTE genuine and fraud transactions count are 99443. It can be seen that now our dataset is balanced. Because HalvingRandomSearchCV is quicker than GridSearchCV and RandomizedSearchCV, we employed it for parameter tuning.

Various ML techniques have been used to test the data in this case. The algorithms are executed on the machine with configuration like, 8 GB RAM, Intel I-3 CPU 3.40GHZ, 64bit Windows10 pro–operating system. We have also used Google Colab to execute our ML algorithms. A few packages are necessary to utilise for any data science project. For the purpose of calculation NumPy has been used. To read data or to store it at specific location in memory Pandas is used. In order to visualise

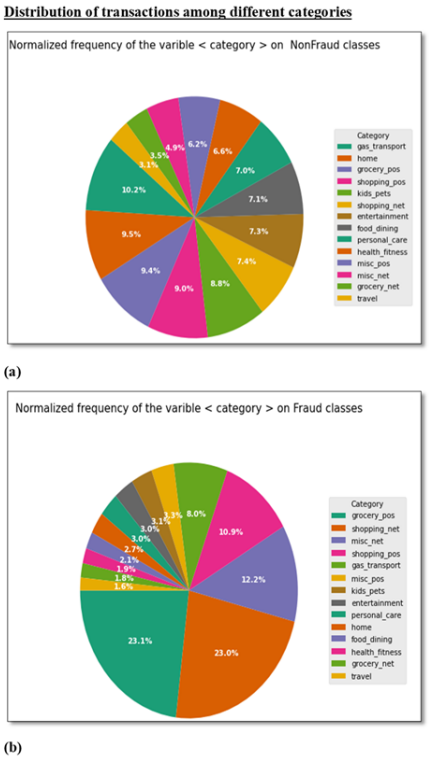


**Fig 4.** Distribution of Transactions by Category — (a) Non-Fraud Classes, (b) Fraud Classes

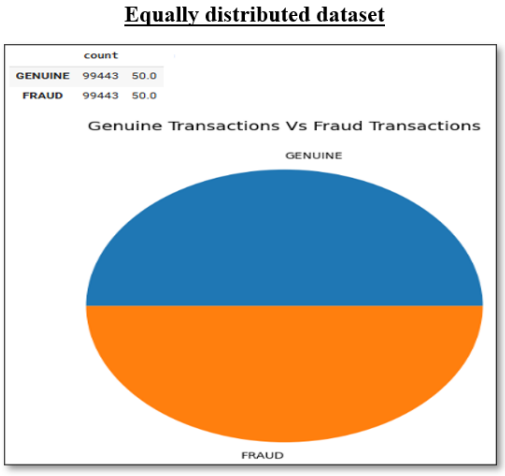


**Fig 5.** Balanced Dataset after applying SMOTE

the data matplotlib is used. Seaborn is used for various tasks like changing the colour of elements, etc. Machine learning techniques are implemented using the Anaconda navigator. The written code is processed using Jupyter Notebook. Logistic Regression, SVM, Decision Tree, Random Forest and XGBoost algorithms are employed in the implementation process.

Logistic regression is a statistical method utilized to forecast the likelihood of a binary result. This technique falls under supervised machine learning. For instance, it can be employed to anticipate if an email is spam or not, if a transaction is fraudulent or legitimate, or if a customer will make a purchase or not. Within logistic regression, the model's output is a sigmoid or logistic function, which changes any real-numbered input into a value ranging from 0 to 1.

Support Vector Machine (SVM) is a supervised machine learning algorithm utilized for classification and regression purposes. The hyperplane is strategically placed to optimize the separation between itself and the nearest points from each class, known as "support vectors." The margin represents the gap between the hyperplane and the closest points from each class. Essentially, SVM's primary objective is to identify the best possible straight line that divides the data into two distinct groups, with the biggest gap possible between them.

A decision tree is a supervised learning method used in machine learning to predict outcomes based on input features. The tree is constructed by repeatedly dividing the data into subsets based on feature values, with each split chosen to maximize separation using metrics like information gain or Gini impurity. The process stops when certain conditions are satisfied, like having too few data points in a group to split further or reaching a predefined tree depth.

Random Forest can deal with both classification and regression. In random forest multiple models are combined to tackle difficult problems and increase prediction accuracy. This algorithm builds several decision trees, each trained on random subsets of the dataset. Instead of relying on the output of a single tree, it combines predictions from all trees to determine the final result. In case of classification problem majority voting of trees will be taken into consideration.

XGBoost (Extreme Gradient Boosting) is a widely used and powerful machine learning algorithm, primarily designed for structured or tabular data. It is based on the concept of gradient boosting, where multiple weak decision trees are built sequentially to create a strong predictive model. In XGBoost, each new decision tree is constructed to correct the mistakes of the previous one, focusing on areas where the model performs poorly. It is also capable of handling missing data and is optimized for high performance, using parallel processing to speed up computation, particularly on large datasets.

## 3 Results and Discussion

The fundamental concept of k-fold cross validation involves partitioning the dataset into "k" subsets or folds. The model is subsequently trained and assessed "k" times, with a distinct fold serving as the test set in each iteration, while the remaining folds are utilized for training purposes. Any classifier will be 100% accurate because of the dataset's nature. Considering accuracy is not the appropriate indicator, alternative metrics are employed to assess the model's performance.

We have used K-fold cross validation technique to verify the result after parameter tuning. Now, this time we have assumed the value of K = 10. We have used various metrics like accuracy, precision, recall, F1-score, ROC-AUC to evaluate the performance of the model.

The dataset has been divided into three different ratios of data: 60%:40%, 70%:30% and 80%:20. The following Tables 2(a)(b) to 4(a)(b) show the performance evaluation of all the models on various metrics on this distribution of data. XGBoost consistently delivers the highest performance across both Train:Test splits and K-fold cross-validation, demonstrating its robustness and adaptability. While K-fold validation helps stabilize performance across all algorithms by reducing bias and variance, XGBoost still outperforms the rest.

As train data increases from 60% to 80%, performance improves slightly for most algorithms due to more data for training. However, computation times also increase. K-Fold Cross-Validation evaluates the model on multiple splits to ensure that the performance is stable and reliable, regardless of which part of the data is used for testing.

In case of Logistic regression, the performance is moderate across all metrics ( 93% accuracy, precision, recall, and ROC-AUC). It is not ideal for maximizing fraud detection due to lower scores. In case of SVM, it gives better precision ( 93%), recall ( 98%), and F1-score ( 95%) than Logistic Regression. But it seems from time perspective, it is time-consuming and specifically on massive datasets, as it takes lots of time for parameter tuning, training, etc. Decision tree gives high recall ( 98–99%) and moderate F1-score ( 97%), with accuracy up to 98.39%. It is much faster than SVM but slower than Logistic Regression. It tries to balance both speed and performance in better way. Random forest gives excellent recall ( 99%) and F1-score ( 98%), with accuracy up to 98.01%. But it is computationally heavy, especially for larger train-test splits. XGBoost gives highest results across all metrics (accuracy 99%, precision 99%, recall 99%, ROC-AUC 99%). It is faster than SVM and Random Forest but slower than Logistic Regression and Decision Tree. It gives consistent results across all train-test splits and K-fold cross-validation. XGBoost is best overall performer for fraud detection in this dataset. In fraud detection, precision and recall matter more than speed. Missing frauds can be very costly.

Logistic Regression is faster, but its lower recall ( 95.8%) means it could miss about 4% of fraud cases. In critical situations like fraud detection, this is not acceptable. Although the Decision Tree has slightly better recall ( 98%), XGBoost outperforms it in all other metrics, especially precision and F1-score.

**Table 2.** Models performance on 60%:40% Distribution. (a) after Parameter Tuning, (b) K-fold technique

| **(a): Train:Test Split => 60%:40%** | | | | | |
|---|---|---|---|---|---|
| **Algorithm** | **Accuracy** | **Precision** | **Recall** | **F1-Score** | **ROCAUC** |
| **Logistic Regression** | 93.27% | 91.23% | 95.81% | 93.46% | 93.26% |
| **SVM** | 95.58% | 93.02% | 98.60% | 95.73% | 95.56% |
| **Decision Tree** | 97.65% | 96.61% | 98.78% | 97.68% | 97.64% |
| **Random Forest** | 96.84% | 94.67% | 99.30% | 96.93% | 96.83% |
| **XGBoost** | 99.31% | 98.84% | 99.80% | 99.32% | 99.31% |
| **(b): K-fold = 10 (60%:40%)** | | | | | |
| **Algorithm** | **Accuracy** | **Precision** | **Recall** | **F1-Score** | **ROCAUC** |
| **Logistic Regression** | 93.30% | 91.20% | 95.79% | 93.44% | 93.30% |
| **SVM** | 95.67% | 93.15% | 98.56% | 95.78% | 95.68% |
| **Decision Tree** | 97.62% | 96.61% | 98.69% | 97.63% | 97.62% |
| **Random Forest** | 97.50% | 95.79% | 99.39% | 97.55% | 97.51% |
| **XGBoost** | 99.23% | 98.68% | 99.78% | 99.23% | 99.23% |

**Table 3.** Models performance on 70%:30% Distribution. (a) after Parameter Tuning, (b) K-fold technique

| **(a): Train:Test Split => 70%:30%** | | | | | |
|---|---|---|---|---|---|
| **Algorithm** | **Accuracy** | **Precision** | **Recall** | **F1-Score** | **ROCAUC** |
| **Logistic Regression** | 93.23% | 91.20% | 95.76% | 93.42% | 93.23% |
| **SVM** | 95.59% | 93.04% | 98.58% | 95.73% | 95.58% |
| **Decision Tree** | 97.92% | 96.87% | 99.04% | 97.95% | 97.91% |
| **Random Forest** | 96.82% | 94.58% | 99.36% | 96.91% | 96.81% |
| **XGBoost** | 99.34% | 98.91% | 99.78% | 99.34% | 99.34% |
| **(b): K-fold = 10 (70%:30%)** | | | | | |
| **Algorithm** | **Accuracy** | **Precision** | **Recall** | **F1-Score** | **ROCAUC** |
| **Logistic Regression** | 93.28% | 91.20% | 95.79% | 93.44% | 93.29% |
| **SVM** | 95.74% | 93.25% | 98.60% | 95.85% | 95.74% |
| **Decision Tree** | 98.05% | 97.14% | 99.02% | 98.07% | 98.05% |
| **Random Forest** | 97.96% | 96.54% | 99.51% | 97.99% | 97.96% |
| **XGBoost** | 98.67% | 97.68% | 99.72% | 98.69% | 98.67% |

Table 5 shows the best hyperparameters which have been received after parameter tuning process on various algorithms. As shown in Table 6, the proposed model achieved outstanding performance across all essential metrics, with XGBoost recording a precision of 99.05% and a recall of 99.81% on the 80:20 train-test split. Table 7 shows Research technique comparison of Previous and Current research.

To ensure the model's robustness, we also implemented 10-fold cross-validation, during which XGBoost maintained a strong performance with a precision of 97.77% and a recall of 99.75%. In comparison to related research, some studies, such as [9] and [10], reported slightly higher accuracy or precision; however, many of these studies lacked consistent validation methods or did not evaluate multiple algorithms at once. Our study not only examined five key algorithms but also validated the results using different data splits and K-fold cross-validation, thereby enhancing the reliability of our conclusions. While a few papers ([2],[7]) reported perfect ROC-AUC scores, such results may indicate overfitting or limited validation. Overall, the proposed approach demonstrates strong, balanced, and reliable performance, with XGBoost recognized as the best-performing model.

**Table 4.** Models performance on 80%:20% Distribution. (a) after Parameter Tuning, (b) K-fold technique

| **(a): Train:Test Split => 80%:20%** | | | | | |
|---|---|---|---|---|---|
| **Algorithm** | **Accuracy** | **Precision** | **Recall** | **F1-Score** | **ROCAUC** |
| **Logistic Regression** | 93.23% | 91.23% | 95.72% | 93.42% | 93.22% |
| **SVM** | 95.71% | 93.22% | 98.62% | 95.84% | 95.69% |
| **Decision Tree** | 97.93% | 96.86% | 99.08% | 97.96% | 97.92% |
| **Random Forest** | 96.93% | 94.65% | 99.49% | 97.01% | 96.92% |
| **XGBoost** | 99.42% | 99.05% | 99.81% | 99.43% | 99.42% |
| **(b): K-fold = 10 (80%:20%)** | | | | | |
| **Algorithm** | **Accuracy** | **Precision** | **Recall** | **F1-Score** | **ROCAUC** |
| **Logistic Regression** | 93.27% | 91.18% | 95.80% | 93.43% | 93.27% |
| **SVM** | 95.75% | 93.25% | 98.64% | 95.87% | 95.75% |
| **Decision Tree** | 98.39% | 97.55% | 99.28% | 98.40% | 98.39% |
| **Random Forest** | 98.01% | 96.60% | 99.56% | 98.05% | 98.01% |
| **XGBoost** | 98.73% | 97.77% | 99.75% | 98.75% | 98.73% |

**Table 5.** Best Hyper Parameters after Parameter Tuning

| **Algorithm** | **Hyper Parameters** |
|---|---|
| Logistic Regression | {'penalty': 'l1', 'C': 0.1, 'solver': 'saga', 'class_weight': 'balanced'} |
| SVM | {'kernel': 'rbf', 'C': 10, 'gamma': 'scale'} |
| Decision Tree | {'splitter': 'best', 'min_samples_split': 10, 'min_samples_leaf': 10, 'max_depth': 50, 'criterion': 'entropy'} |
| Random Forest | {'n_estimators': 500, 'max_depth': 20, 'min_samples_split': 10, 'min_samples_leaf': 10, 'max_features': 'sqrt', 'criterion': 'gini'} |
| XGBoost | {'n_estimators': 500, 'max_depth': 10, 'learning_rate': 0.3, 'subsample': 0.8, 'colsample_bytree': 1.0, 'gamma': 0.2, 'reg_alpha': 0.5, 'reg_lambda': 0.5} |

**Table 6.** Metrics Evaluation Comparison

| **Result Comparison Table** | | | | | | |
|---|---|---|---|---|---|---|
| **Reference** | **Algorithms Used** | **Accuracy** | **Precision** | **Recall** | **F1-Score** | **ROCAUC** |
| (2) | Local Outlier Factor, Isolation Forest | 99.60% | 33% | 33% | 33% | – |
| (3) | Naïve Bayes, Logistic Regression, Random Forest, AdaBoost | 100% | 100% | 100% | 100% | Area=1(100%) |
| (4) | Logistic Regression, Naive Bayes, Decision Tree, Random Forest, ANN | 98.69% | 98.41% | 98.98% | – | – |
| (5) | Logistic Regression, Random Forest, Naive Bayes, Multilayer Perceptron | 99.96% | 96.38% | 81.63% | – | – |
| (6) | Naïve Bayes, Support vector machine, Logistic regression, Random Forest, Decision tree, and XGBoost | 96.44% | 96% | 97% | 96% | 96% |
| (7) | K-Neighbors, Random Forest, XGboost, Adaboost classifiers and Ensemble Model (Stacking Technique) | 97.31% | 98% | 97% | 97% | Area=1(100%) |
| (8) | SVM, Random Forest, Bagging, XGBoost, and DT | 99% | 89% | 79% | 84% | 98% |
| (9) | Naïve Bayes, Support vector machine, Logistic regression, Random Forest, Decision tree, KNN and XGBoost | 99.82% | 99.83% | 99.88% | 99.90% | 99% |
| (10) | Random Forest | 99.93% | 99.78% | 99.85% | – | – |
| Present Work | Logistic Regression, SVM, Decision Tree, Random Forest, XGBoost | 99.42% | 99.05% | 99.81% | 99.43% | 99.42% |

**Table 7.** Comparative Study Table

| COMPARATIVE STUDY TABLE | | | | | | |
|---|---|---|---|---|---|---|
| **Reference** | **Train-Test Ratio** | **Cross Validation** | **Balancing Technique** | **Hyperparameter Tuning** | **Feature Engineering** | **Evaluation Metrics** |
| [2] | N/A | N/A | N/A | N/A | No geographical feature and Minimal feature engineering | The primary focus is on accuracy and precision, while also providing records on recall and F1-score. |
| [3] | 70%:30% | N/A | Random Sampling | N/A | N/A | Accuracy, Precision, Recall, F1-Score, Support |
| [4] | N/A | N/A | Oversampling | N/A | N/A | Accuracy, Precision, Recall |
| [5] | 80%:20% | N/A | SMOTE | N/A | Minimal feature engineering | Accuracy, Precision, Recall |
| [6] | 70%:30% | N/A | N/A | Minimal Parameter tuning | N/A | Precision, Recall, Accuracy, AUC, and f-measure |
| [7] | N/A | 5-fold CV | Random over sampling and SMOTE-ENN | N/A | Minimal feature engineering | Accuracy, precision, recall and f1-score |
| [8] | 80%:20% | N/A | SMOTE | N/A | Minimal feature engineering | Precision, F1-score, accuracy, and recall |
| [9] | 70%:30% | N/A | N/A | N/A | Minimal feature engineering | Accuracy, Precision, Recall and F1-score, ROC |
| [10] | 70%:30% | N/A | N/A | N/A | N/A | Accuracy, Precision, Recall |
| This Research | 60%:40%, 70%:30%, 80%:20% | 10-fold CV | SMOTE | HalvingRandomSearchCV | Added new meaningful geographic distance feature along with other feature transformation | Accuracy, Precision, Recall, F1-score, ROCAUC Main focus on Precision, Recall |

## 4 Conclusion

This research conducted a comparative analysis of five machine learning algorithms—Logistic Regression, Support Vector Machine (SVM), Decision Tree, Random Forest, and XGBoost—for the detection of credit card fraud, utilizing a simulated dataset created through Sparkov and obtained from Kaggle. The study involved thorough testing across three dataset divisions (60:40, 70:30, and 80:20) and employed SMOTE to mitigate class imbalance. XGBoost was identified as the top-performing algorithm, achieving impressive precision and recall rates of 99.05% and 99.81%, respectively. The Decision Tree and Random Forest algorithms followed, with recall rates of 99.08% and 99.49%, although their precision was lower than that of XGBoost. A key contribution of this research is the assessment of model performance across various train-test splits, complemented by 10-fold cross-validation, which enhances the robustness and reliability of the findings. In contrast to many prior studies that utilize a fixed data split, this analysis offers valuable insights into the impact of different data splits on model performance. The study's strength lies in its thorough performance evaluation using diverse metrics and validation techniques. However, a notable limitation is the dependence on a synthetic dataset, which may not accurately reflect the complexities and transaction patterns of real-world scenarios. Furthermore, while SMOTE aids in balancing the dataset, it could introduce artificial patterns absent in genuine fraud cases. Future research could focus on applying these algorithms to real-world, large-scale transactional datasets, integrating deep learning models, or employing ensemble methods that combine multiple classifiers for enhanced performance. Additionally, incorporating real-time detection capabilities and cost-sensitive learning could increase the system's relevance in practical applications. It is advisable for practitioners to prioritize models such as XGBoost for fraud detection while also exploring hybrid or adaptive strategies that can adapt to evolving fraud patterns.

## 5 Acknowledgement

The authors express their gratitude to Brandon Harris for creating a user-friendly simulation tool to generate datasets of fraudulent transactions which has been instrumental in enabling effective experimentation and analysis of this project.

## References

1) Dhwanir S, Kumar SDL. A Survey on Credit Card Fraud Detection Using Machine Learning. In: National Conference on Contemporary Practices in Management & Information Technology KSCON2021 (Virtual mode), November 2021. . Available from: https://drive.google.com/file/d/1OK9s6hiRd_Dv6akz6K1Buy1Wbsp8LcN1/view.
2) Maniraj SP, Saini A, Sarkar SD, Ahmed S. Credit Card Fraud Detection Using Machine Learning and Data Science. *International Journal of Engineering Research and Technology (IJERT)*. 2019;8(9). http://dx.doi.org/10.17577/IJERTV8IS090031.
3) Bhanusri A, Valli KRS, Jyothi P, Sai GV, Subash RRS. Credit card fraud detection Using Machine learning algorithms. *Quest Journals, Journal of Research in Humanities and Social Science*. 2020;8(2):4–11. Available from: https://www.questjournals.org/jrhss/papers/vol8-issue2/B08020411.pdf.
4) S MVKK, G MVKV, A MVS, K MP. Credit Card Fraud Detection using Machine Learning Algorithms. *International Journal of Engineering Research & Technology (IJERT)*. https://doi.org/10.17577/IJERTV9IS070649.
5) Varmedja D, Karanovic M, Sladojevic S, Arsenovic M, Anderla A. Credit Card Fraud Detection - Machine Learning methods. . https://doi.org/10.1109/INFOTEH.2019.8717766.
6) Mohbey KK, Khan MZ, Indian A. Credit-Card-Fraud-Prediction-Using-XGBoost-An-Ensemble-Learning-Approach. *International Journal of Information Retrieval Research*. 2022;12(2). Available from: https://www.researchgate.net/publication/361982480_Credit-Card-Fraud-Prediction-Using-XGBoost_-An-Ensemble-Learning-Approach. https://doi.org/10.4018/IJIRR.299940.
7) Faruq O, Mily AS, Ahammed F, Islam A. Machine Learning for Enhanced Churn Prediction in Banking: Leveraging Oversampling and Stacking Techniques. *International Journal of Scientific Research and Management (IJSRM)*. 2024 September. https://doi.org/10.18535/ijsrm/v12i09.ec03.
8) Sinha H. An examination of machine learning-based credit card fraud detection systems. *International Journal of Science and Research Archive*. https://doi.org/10.30574/ijsra.2024.12.2.1456.
9) Khadhori SSMA, Mukhaini AJKA, Sherimon V. MACHINE LEARNING APPROACHES FOR CREDIT CARD FRAUD DETECTION: A PREDICTIVE ANALYSIS. *International Journal of Artificial Intelligence & Machine Learning (IJAIML)*. https://iaeme.com/Home/article_id/IJAIML_03_01_005.
10) Raghavendra MN, Chiranjeevi KGD. Credit Card Fraud Detection Using Machine Learning. https://doi.org/10.71097/IJSAT.v16.i1.2684.
11) Gopalakrishnan K. Fraud Detection in Banking: A Machine Learning Approach using Credit Card Transaction Data. *Journal of Artificial Intelligence & Cloud Computing*. Available from: https://www.onlinescientificresearch.com/articles/fraud-detection-in-banking-a-machine-learning-approach-using-credit-card-transaction-data.pdf.
12) S PM, Swetha BN, Patil M. CREDIT CARD FRAUD DETECTION USING MACHINE-LEARNING. *International Journal of Advanced Research (IJAR)*. https://dx.doi.org/10.21474/IJAR01/16824.
13) Unogwu OJ, Filali Y. Fraud Detection and Identification in Credit Card Based on Machine Learning Techniques. *Wasit Journal of Computer and Mathematics Science*. https://doi.org/10.31185/wjcms.185.
14) Rosenbam A. Detecting Credit Card Fraud with Machine Learning. Available from: https://cs229.stanford.edu/proj2019spr/report/32.pdf.
15) Nobel SMN, Sultana S, Singha SP, Chaki S, Mahi MJN, Jan T, et al. Unmasking Banking Fraud: Unleashing the Power of Machine Learning and Explainable AI (XAI) on Imbalanced Data. *Information*. https://doi.org/10.3390/info15060298.
16) Khatri S, Arora A, Agrawal AP. Supervised Machine Learning Algorithms for Credit Card Fraud Detection: A Comparison. Available from: https://ieeexplore.ieee.org/document/9057851. https://doi.org/10.1109/Confluence47617.2020.9057851.
17) Wang X. Credit Card Fraud Prediction Based on Machine Learning Algorithms. . https://doi.org/10.54254/2754-1169/47/20230366.
18) Feng X, Kim SK. Novel Machine Learning Based Credit Card Fraud Detection Systems. https://doi.org/10.3390/math12121869.
19) A TM, Varmann SSM, Sharmila L, Das SR. Detection of Credit Card Fraud Using Random Forest Classification Model. *FMDB Transactions on Sustainable Technoprise Letters*. Available from: https://www.fmdbpub.com/user/journals/article_details/FTSTPL/137.
20) Al-Maari AA, Abdulnabi M, Nathan Y, Ali A, Ali U, Khan M. Optimized Credit Card Fraud Detection Leveraging Ensemble Machine Learning Methods. https://doi.org/10.48084/etasr.10287.
21) Kalid SN, Khor KC, Ng KH, Tong GK. Detecting Frauds and Payment Defaults on Credit Card Data Inherited with Imbalanced Class Distribution and Overlapping Class Problems: A Systematic Review. https://doi.org/10.1109/ACCESS.2024.3362831.
22) Hasugian LS, Suharjito. Fraud Detection for Online Interbank Transaction Using Deep Learning. https://doi.org/10.36418/syntax-literate.v8i6.12627.
23) Steven W, Wilfredo W, Kristina G, Abdi D. Analysis of Credit Card Fraud Detection Performance Using Random Forest Classifier & Neural Networks Model. https://doi.org/10.47191/etj/v9i02.11.
24) Ndama O, Bensassi I, En-Naimi EM. Innovative credit card fraud detection: a hybrid model combining artificial neural networks and support vector machines. https://doi.org/10.11591/ijai.v13.i3.pp2674-2682.
25) Kaggle com. Credit Card Fraud Detection. Available from: https://www.kaggle.com/code/rahulrajml/fraud-detection-systematic-approach/data.