International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 11 | Nov - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

A Hybrid Approach based on Ensemble Learning and SMOTE Technique for Credit Card Fraud Detection

Ravinder Kumar[1], Kulwinder Singh[2]
[1]Research Scholar, Department of CSE, Bhai Maha Singh College of Engg., Sri Muktsar Sahib
[2]Professor, Department of CSE, Bhai Maha Singh College of Engg., Sri Muktsar Sahib.

Abstract: In this study, an ensemble learning-based credit card fraud detection framework was developed to overcome the limitations of individual machine learning algorithms. Initially, a fraud detection dataset was collected and pre-processed using min-max normalization, and the Synthetic Minority Over-sampling Technique (SMOTE) was applied to balance the classes. The prepared dataset was then fed to five machine learning algorithms, and the best three were selected based on their individual performance. Hyperparameter tuning was performed for the selected algorithms using the Optuna optimization framework. Finally, the predictions of the two best-performing algorithms were combined using a weighted voting classifier. The framework was implemented in Python using Google Colab. Its performance was evaluated against existing approaches using accuracy, precision, recall, and F1-score. In addition, SHAP analysis was used to interpret the dataset's features and determine which ones contribute most to fraud detection. The experimental results showed that the proposed method achieved an accuracy of 0.99696, a precision of 0.903226, a recall of 0.879581, an F1-score of 0.879581, an ROC-AUC of 0.975524, and a PR-AUC of 0.882359, outperforming the existing approaches.

Keywords: Credit Card, Ensemble Learning, Financial Transaction, Hyper-Parameter Tuning, Optuna, SMOTE.

1. Introduction

In today's time, internet banking has been used by financial institutions to provide business facilities to the public. In today's financial society, E-payments via internet banking play a significant role [1]. It has made the experience of buying goods and services very convenient. Using technology, financial institutions provide cards to their customers that can be used anywhere for financial payments instead of cash. There are two types of cards named credit cards and debit cards. Both these cards are useful for customers for purchasing products and protect purchases that might be stolen or damaged. Before using a credit card, customers need to verify the transaction with the merchant [1]. A report by statistics showed that Mastercard [2] and VISA [2] issued 2,287 million credit cards worldwide by 2020 (Figs. 1 and 2). Globally, Mastercard and Visa issued 1156 million and 1131 million cards. These numbers show the popularity of card-based transactions among users and their ease of use. In this sector, many transactions take place, making it easy for attackers to target and manipulate the population. There are also times when it might be easy to trick people.

There is no doubt that credit cards are very useful for customers, but they are also prone to several risks like fraud and lack of security. In recent times, several financial institutions and banks have faced the issue of credit card fraud. It happens when unauthorized people use credit cards to acquire money or property. The formation of credit cards can be stolen via online web pages and platforms that are not secured.



Figure 1: Amount of Master Card Issued

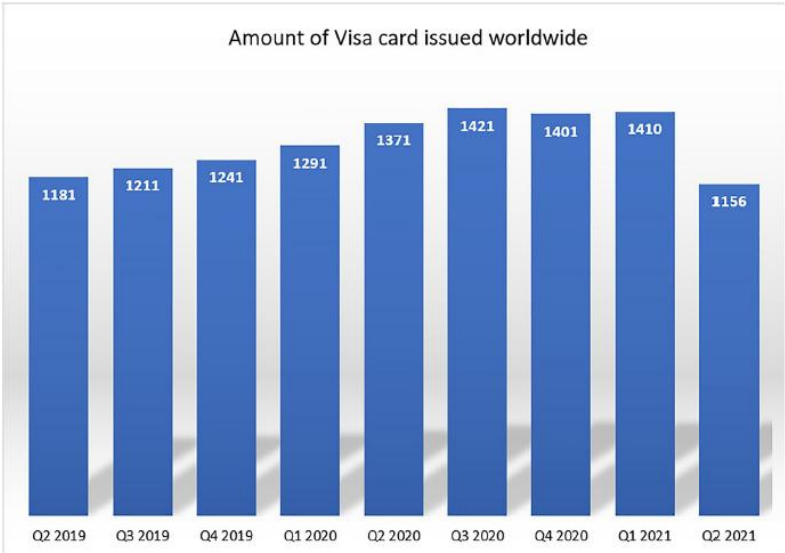


Figure 2: Amount of Visa Cards Issued

These details can be acquired via identity theft schemes as well. The fraudsters can illegally access users' information regarding credit and debit card numbers. Then, they can this information to steal money from their accounts. In such scenarios, ML models can be useful in finding which transactions are legitimate and which are fraudulent.

Alarfaj et al. [3], proposed a DL based approach for enhancing the identification performance of fraudulent credit card transactions. The research used the dataset of European credit card benchmark dataset. The dataset contained 284,807 records of transactions that include 492 (0.172%) fraudulent transactions. This showed a clearly imbalanced dataset. The study developed several ML models like RF, DT, LR, kNN, SVM, XGBoost, and DT along with DL models that were based on CNN. They utilized PCA (Principal Component Analysis) for representing features and different CNN architectures, training epochs, and the number of hidden layers. These were all studied to enhance the performance of classification. The proposed DL approach resulted in an accuracy of 99.9%, precision of 93%, AUC of 98%, and F1-score of 85.71%. They also investigated techniques for data balancing with the aim of reducing false negatives. They suggest that CNN-based models can be effective in fraud detection. However, the imbalance in the dataset emphasizes the need to consider fraud-specific metrics instead of

just accuracy. **Ileberi et al. [4],** proposed a ML based system for detecting credit card fraud. This research used real-world dataset that was produced from European credit card holders but the dataset was highly imbalanced. To resolve the problem of data imbalance, the research re-sampled the dataset using SMOTE technique. Several ML models were then utilized on the balanced dataset like LR, SVM, XGBoost, RF, DT, and ET. Furthermore, the research used AdaBoost with ML algorithms to further improve the classification performance of models. Several performance metrics were utilized in this research to measure the model's performance like recall, precision, accuracy, AUC, and MCC. The higher accuracies were achieved by proposed models with DT-AdaBoost (99.67%), RF-AdaBoost (99.95%), ET-AdaBoost (99.98%), and XGB-AdaBoost (99.98%). Based on the classification quality, two models ET-AdaBoost and XGB-AdaBoost performed superior as compared to other models with MCC of 0.99. **Khalid et al. [5],** proposed a unique ensemble model that combines SVM, KNN, RF, Bagging, and boosting classifiers. The proposed model also handled the problem of data imbalance using under-sampling and SMOTE technique for some ML models. The dataset used in this research is taken from European credit card holders that provide real-time data for the research. The proposed methodology includes pre-processing, feature engineering, model selection, and evaluation. The model development was done on Google Colab that provided high computation capabilities and effective model. The proposed model of this research achieved the accuracy of 99.9591%, demonstrating superior performance. **H. S. Alsagri [6],** presented heterogeneous ML methods for CCF (credit card fraud) detection. The proposed system uses multi-stage classification by using several classifiers. These classifiers were SVM, LR, RF, KNN, DNN, and XGBoost. The research used sampling technique to handle class imbalance and a feature selection mechanism using voting strategy. The experimental findings of the study showed that the proposed model performed superior as compare to previous DNN simple voting and stacking framework with legitimate recall of 0.995, a fraud recall of 0.901, and model cost of 0.421.**Baisholan et al. [7],** proposed an ensemble-based model FraudX AI to detect frauds and addressed the challenges of imbalanced datasets, scalability, and interpretability. The proposed FraudX AI combines XGBoost and RF as baseline models and then combined their results by taking average probabilities and then used threshold optimization technique to enhance the fraud detection performance. The research used the European credit card data to evaluate the proposed model. However, the proposed approach used the natural imbalanced data and achieved a recall of 95%, AUC-PR of 97%, and reduced the false positives. **Gupta et al. [8],** introduced a ML-based fraud detection system that included SMOTE-ENN for data imbalance, TOPSIS for feature selection, and autoencoder for dimensionality reduction. The research developed a stacking model using KNN, SVM, and Elm to improve the model's forecast accuracy. Furthermore, the research employed PSO algorithm to optimize the parameters of ELM model. This helps in improving model's generalization and convergence. The experimental results showed that the model achieved accuracy of 99.95%, precision of 99.93%, and recall of 99.97%. **Theodorakopoulos et al. [9],** implemented a PySpark-based system that processes dataset of large-scale transactions and used advanced ML models like DT, LR, XGBoost, RF, and CatBoost. From the selected models, CatBoost and XGBoost performed superiorly as compared to other models with recall, precision, and F1-score above 0.999 on test set. Their strength and generalization capability on new data and capacity for deployment in dynamic financial systems are evident. Both Random Forest and Decision Tree classifiers performed well, although their minor test-phase performance drop compared to training suggests moderate overfitting. However, they are beneficial for interpretability and quick training. Logistic Regression performed well with accuracy and recall at 0.96, confirming its use as a lean baseline model. The findings show that distributed gradient boosting models in PySpark provide a fast and scalable real-time fraud detection solution. This research shows that ML pipelines that can manage unbalanced datasets, allow low-latency inference, and adapt to the complexity of financial network operations are needed. To improve fraud detection systems' flexibility, fairness, and regulatory compliance, real-time learning, privacy-preserving methods, and behavioral analytics are necessary. **Zavvar et al. [10],** addressed the problem of detecting fraudulent transactions by using SMOTE, CNN, attention mechanism, and autoencoder in a single framework named SMOTE-AE-CNN-Att. First, the dataset is balanced using the SMOTE technique, AE-CNN-Att is used for feature extraction, and then ML models are used

for classification. The proposed approach results in higher accuracy (greater than 99.9%) in identifying fraudulent transactions with higher performance metrics. The proposed model results in precision of 90.07%, F1-score of 90.60%, and recall of 91.13%. As compared to other models, the SMOTE-AE-CNN-ATT model performed better in finding a balance between recall and accuracy that plays an important role in identifying fraud. **Karunya et al. [11],** investigated ML techniques for identifying fraudulent transactions related to credit cards. They evaluated the models kNN, SVM, and XGBoost and proposed a new probability-based kNN model using LR probabilities. The research used a dataset that contains 25,134 records related to credit card transactions and 19 features. The features were related to demographics and customer information. First, the research performed basic EDA (Exploratory Data Analysis), duplicate checks, one-hot encoding, outlier analysis, missing value analysis, and standardization. Furthermore, to handle the class imbalance in the dataset, the research used SMOTE and ADASYN. The proposed probability-based kNN model achieved an accuracy of 0.8226 and a precision of 0.836 with the SMOTE technique. However, the models XGBoost resulted in higher F1-score (0.743) and recall (0.641) with SMOTE. With ADASYN, the probability-based kNN model achieved an F1-score of 0.8144 and recall of 0.7998. Thus, the study resulted in that data balancing techniques are more effective to improve the performance of fraud detection systems. However, the results also showed that no single model performed better across all performance metrics. This shows the importance of balancing computational efficiency and fraud detection capabilities.

In the previous studies, several ML algorithms were used. However, individual models normally face the problem of sensitivity to hyperparameters, overfitting, and poor detection of minority-class fraudulent transactions. Moreover, some algorithms achieve higher accuracy but lower values for recall and F1-scores. This shows the undetected fraudulent transactions. Also, selecting the hyperparameters manually or by default configurations may limit the model's performance. To address these limitations, a framework is developed in which data class is balanced, hyper-parameter tuning using the Optuna optimizer, and ensemble learning approach is employed for enhance the performance of the credit card fraud detection method.

The rest of the structure of the paper is as follows. Section 2 presents the methodology. The results and analysis are elaborated in Section 3. Finally, this study is concluded, and future direction is defined in Section 4.

2. Methodology

Figure 3 presents the framework of the proposed credit card fraud detection framework, and a detailed description of the steps is given below.

2.1 Data Collection

The proposed method was evaluated using a simulation on the open-source credit card dataset available on Kaggle [12]. This dataset contains 284,807 transaction records, and each record contains 31 features. From this, 99.83% of records are related to non-fraud transactions, and only 0.17% is related to credit card fraud transactions. As shown in Figure 4. This indicates that the dataset is highly imbalanced. Therefore, in the pre-processing step, the data balancing was done.

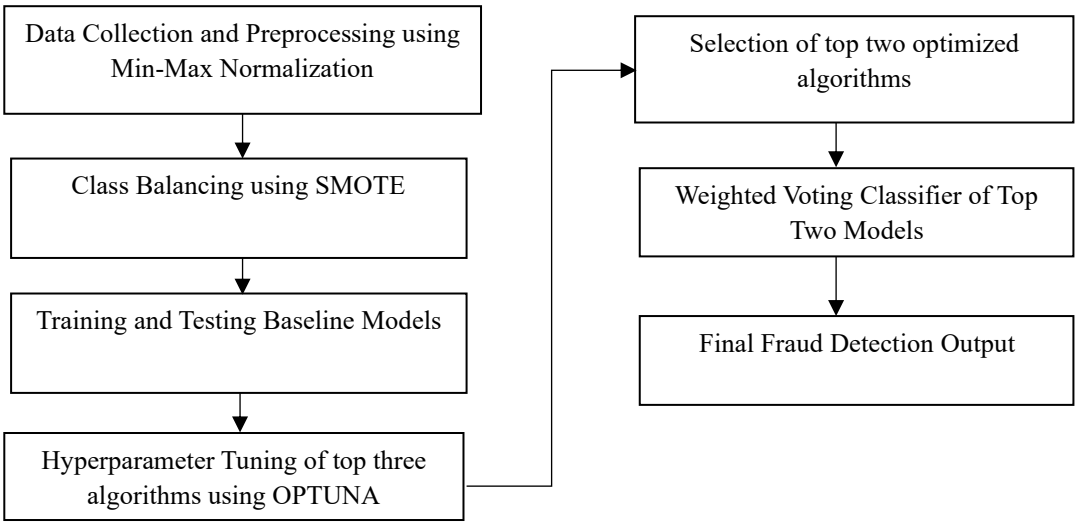


Figure 3: Framework of the Proposed Credit Card Fraud Detection

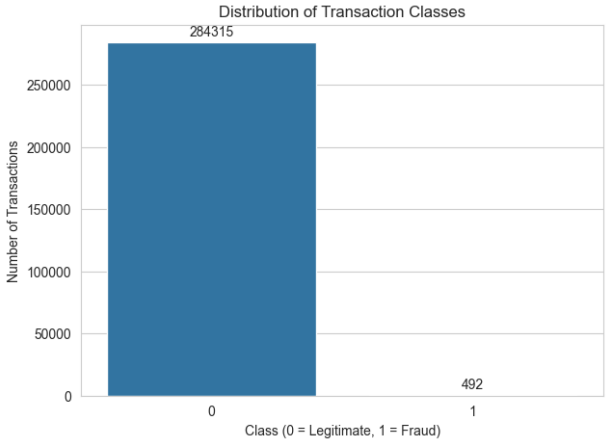


Figure 4: Distribution of Transaction Classes

2.2 Pre-Processing of the Dataset

So, first, dataset was loaded and pre-processed to know the data-types and class distribution. Then data was split into training and testing ratio of 80:20. Table 1 shows the training and testing sample sets of the data after splitting into 80:20.

Table 1: Training and Testing Sample Sets

Dataset	Total Samples	Class 0	Class 1
Training	227,845	227,451	394
Testing	56,962	56,864	98
Total	284,807	284,315	492

After that MinMax Scaler was applied on training and testing data separately. After that, to balance the dataset, BorderlineSMOTE is used that is an advanced version of the SMOTE technique. BorderlineSMOTE generates synthetic data only for the minority class that exists near the decision boundary.

2.3 Model Development

After that, five machine learning algorithms were applied to the dataset, named LR, DT, RF, KNN, and XGBoost.

Logistic Regression: This model uses the technique of linear regression in which the model predicts the numerical value for an input. This value is then converted to a probability using a sigmoid function. Then, the class is predicted based on the probability value and threshold [13].

Decision Tree: This model classifies data by splitting it into smaller sets depending on the feature values. This model prepares a tree-like structure of decision rules. In this, the traversal starts from the root, and the last-level leaves lead to the final class prediction.

Random Forest: The RF is also an ensemble learning-based model that combines several DTs that are trained on different subsets of data. The final class output is estimated using majority voting. This model improves the stability of predictions and leads to reduced overfitting in the model.

K-Nearest Neighbor: This model works by finding the nearest k training samples of the test data using a distance measure [13]. The prediction class is found by the majority of these nearest neighbors.

XGBoost: It is also called Extreme Gradient Boosting, which is a type of ensemble model that combines several weak learners (called decision trees) in a sequence. Then the model optimizes performance by correcting the errors (residuals) of previous trees before creating a new tree [14].

Ensemble Learning: After that, the top three models were selected to be further optimized using the Optuna hyperparameter optimization framework. From these three models, the top two (XGBoost and KNN) were selected for the final ensemble model. Finally, the weighted voting classifier was used to combine the predictions by assigning the 0.7 and 0.3 weight values to the selected ML algorithms.

2.4 Performance Metrics

Several performance metrics are used to evaluate the proposed model's performance, including accuracy, precision, recall, F1-score, ROC-AUC, and PR-AUC. At the end, SHAP analysis is also performed for feature importance and feature interpretability [15-18].

- **Accuracy:** Accuracy is the proportion of correctly predicted cases (both positive and negative) out of the total number of cases. A high accuracy indicates that the model correctly predicts most cases, and, therefore, it can be misleading in imbalanced datasets. For instance, in a dataset where 99% of transactions are genuine (non-fraudulent), a model predicting all transactions as genuine will have 99% accuracy despite being ineffective at detecting fraud.

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

- **Precision:** Precision is a performance metric that quantifies the proportion of correctly predicted positive cases (fraud) out of all cases classified as positive by the model. It measures the model's ability to avoid false positives (false alarms). In fraud detection, a model with low precision may misclassify many legitimate transactions as fraudulent, leading to unnecessary inconvenience for users. High precision reduces operational costs and maintains user trust by minimizing false alarms.

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

- **Recall:** Recall, also called sensitivity or true positive rate (TPR), evaluates a model's ability to correctly identify all actual positive cases in a dataset. In fraud detection, recall indicates how effectively the model identifies fraudulent transactions among all actual fraud cases. A high recall value signifies that the model successfully detects most fraudulent transactions, minimizing false negatives. Conversely, a low recall value suggests that the model fails to detect a significant number of fraud cases, which can have severe financial and reputational consequences.

$$Recall = \frac{TP}{TP+FN} \quad (3)$$

- **F1-score:** The F1-score is a performance metric that balances precision and recall. It is beneficial for evaluating models in scenarios with class imbalance.

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

- **ROC-AUC:** It is the Receiver Operating Characteristic–Area under the Curve that is used to evaluate the ability of the model in differentiating between fraudulent and legitimate transactions. A higher value of ROC-AUC generally shows better classification performance.
- **PR-AUC:** It is the Precision-Recall Area under the Curve that is used to measure the balance between the models' precision and recall for different thresholds. This measure is generally useful for imbalanced datasets that contain very few records of fraudulent transactions as compared to legitimate transactions.
- **Feature Interpretation:** SHapley Additive exPlanations (SHAP) are used in this research for measuring the contribution of each feature in the final prediction of the model. SHAP is used for both local and global explanations that help to identify the most influential features and their effect on the final prediction.

3. Results and Analysis

This section presents the simulation evaluation of the proposed method and comparative analysis with the state-of-the-art techniques. All of the tests were done on a laptop with an Intel i7 processor, a 64-bit Windows operating system, 16 GB of RAM, and a 1 TB hard drive. This configuration provided sufficient computational power for data loading, preprocessing, model training, and evaluation. The implementation was carried out in the Python language and simulated on Google Colab software. The standard ML libraries and functions, including pandas, NumPy, TensorFlow/Keras, Matplotlib, and scikit-learn, were used to develop the proposed framework. The section begins with the simulation setup configuration of the proposed method followed by the simulation results and comparative analysis with the existing methods in Sections 3.2-3.3.

3.1 Simulation Setup Configuration

Table 2 shows the simulation setup configuration that was used to evaluate the proposed method. This table defines the hyper-parameter configuration of the ML algorithms. Finally, the ensemble learning and Optuna optimizer algorithm parameters were defined that were used to combine the prediction of ML algorithms and hyper-tune the parameters.

Table 2: Simulation Parameter Configuration of the Fraud Detection Framework

Model	Key Hyper-parameters
Logistic Regression	Max-Iteration=1000, solver=lbfgs, random-state=42
Decision Tree	Max-depth=12, min-samples-split=5, min-samples-leaf=2, random-state=42
Random Forest	n-estimators=200, max-depth=15, min-samples-split=5, min-samples-leaf=2, max-features=sqrt, random-state=42, n-jobs=-1
KNN	n-neighbours=5, weights=distance, n-jobs=-1
XGBoost	n-estimators=300, max-depth=6, learning-rate=0.05, min-child-weight=3, subsample=0.8, col-sample-bytree=0.8, objective=binary: logistic, eval-metric=logloss, random-state=42, n-jobs=-1
Ensemble Learning	xgb_weight = 0.7 knn_weight = 0.3

3.2 Simulation Results

Figure 5 shows the data samples before and after SMOTE technique for the training set. The graph shows that SMOTE successfully raised the fraud class from 394 to 68,235 samples, moving the training set from extreme imbalance (about 0.17% fraud) to a manageable one (about 23% fraud) without discarding any legitimate data. The setup is methodologically sound because it was applied only to the training set.

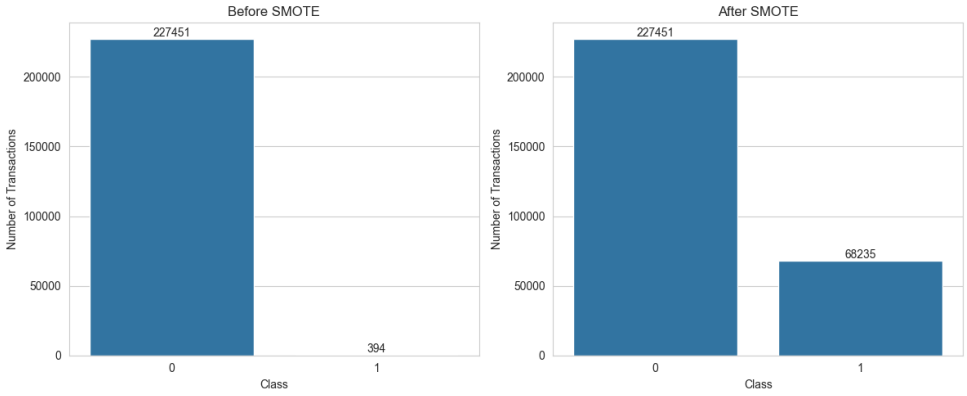


Figure 5: Data Samples Before and After SMOTE Technique

Table 3 shows the evaluation of the individual ML algorithms. The result indicates that the **XGBoost is the best model** (PR-AUC 0.8692, F1 0.8259, ROC-AUC 0.9739), with Random Forest a near-equal and more stable alternative. Both offer a strong precision/recall balance of roughly 0.81 to 0.85. Logistic Regression is a high-recall, low-precision baseline, and Decision Tree is the weakest overall, as shown in Figure 6.

Table 3: Evaluation of the ML Algorithms

Model	Accuracy	Precision	Recall	F1-Score	ROC-AUC	PR-AUC
Logistic Regression	0.9950	0.2374	0.8673	0.3728	0.9682	0.6561
Decision Tree	0.9985	0.5423	0.7857	0.6417	0.8608	0.5076
Random Forest	0.9993	0.8061	0.8061	0.8061	0.9691	0.8579
KNN	0.9994	0.8039	0.8367	0.8200	0.9335	0.7801
XGBoost	0.9994	0.8058	0.8469	0.8259	0.9739	0.8692

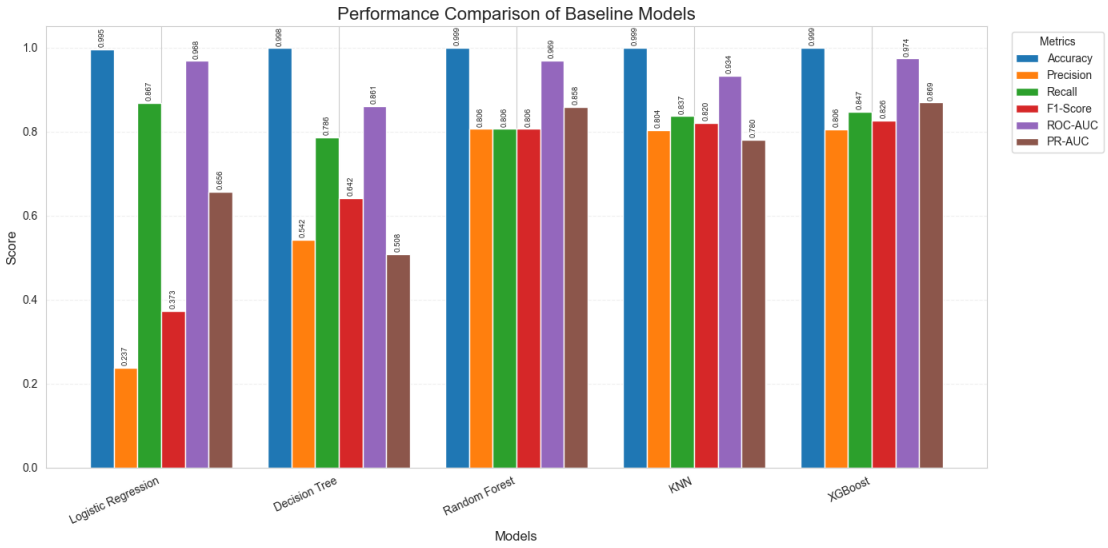


Figure 6: Graphical Evaluation of the Individual ML Algorithms

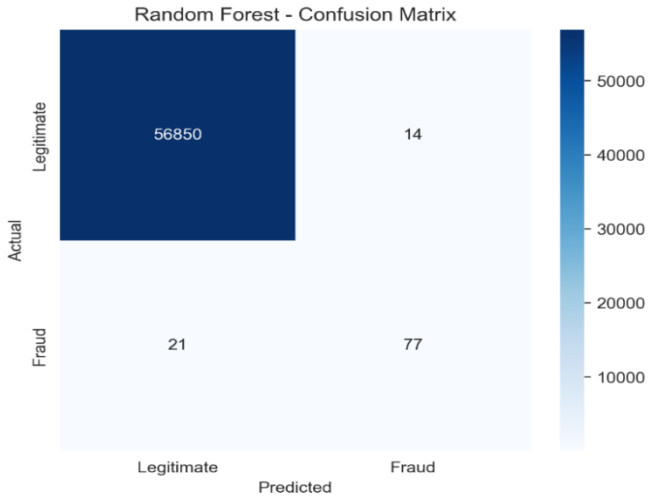
Based on the individual performance of the ML algorithms, the top three best-performing ML algorithms were chosen for optimal hyperparameter tuning using the Optuna optimizer. For all three ML models, an Optuna objective function was defined using the same optimization strategy. In each trial, Optuna sampled a set of model-specific hyperparameters from the predefined search space. The model was then trained using the resampled training data, and predictions were generated on the validation set. The binary F1-score was calculated from the validation predictions and returned as the objective value. Optuna iteratively explored different hyperparameter combinations with the objective of maximizing the validation F1-score. Thus, the overall objective-function structure remained consistent across Random Forest, KNN, and XGBoost, while the hyperparameter search space varied according to the characteristics of each model. The optimized hyper-parameters value for the selected ML algorithm is presented in Table 4.

Table 4: Optimized Hyper-Parameters Value for the Selected ML Models

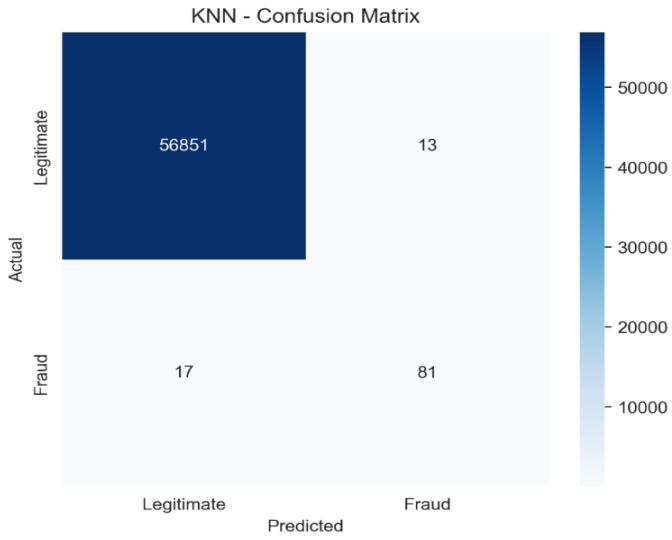
Model	**Optimized Hyperparameters**
Random Forest	n-estimators=300, max-depth=19, min-samples-split=4, min-samples-leaf=3, max-features=log2, class-weight=None
KNN	n-neighbors=3, weights=distance, p=2, leaf-size=30
XGBoost	n-estimators=400, max-depth=8, learning-rate=0.1970, min-child-weight=1, subsample=0.8422, col-sample-bytree=0.9987, gamma=0.0128

Figure 7 (a-c) shows the confusion matrix for the three ML algorithms: Random Forest, K-Nearest Neighbours, and XGBoost. These matrices demonstrate the effectiveness of the models in differentiating legitimate and fraudulent transactions. The RF model accurately classified 56,850 legitimate transactions and 77 fraudulent transactions. But it misclassified 14 legitimate transactions as fraud and failed to classify 21 fraudulent transactions. Overall, the model performed very well as compared to baseline models.

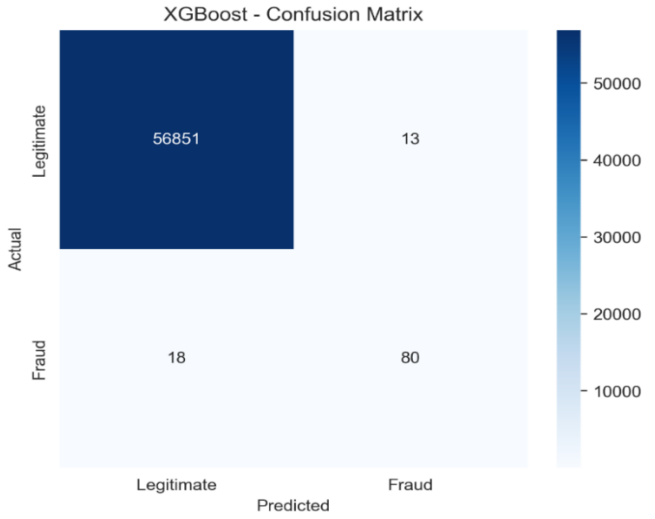
Similarly, the KNN model accurately classified 56,851 legitimate transactions and identified 81 fraudulent transactions. It generated 13 false positives and 17 false negatives, demonstrating effective fraud detection with relatively few classification errors.



(a) Confusion Matrix for Optimized Random Forest



(b) Confusion Matrix for Optimized KNN



(c) Confusion Matrix for Optimized XGBoost

Figure 7: Confusion Matrix (a) Random Forest (b) KNN (c) XGBoost

From the confusion matrix of the XGBoost model, the model accurately classified 56,851 legitimate transactions and 80 fraudulent transactions. However, the model incorrectly predicted 13 legitimate transactions as fraudulent transactions and missed 18 fraudulent transactions. The model demonstrated strong classification performance, achieving a balance between fraud detection and minimizing false alarms.

Table 5 shows the evaluation of the ML algorithms after hyper-parameter tuning using the Optuna optimizer. Tuning improved the overall balance of performance for all three models, mainly by increasing precision. For XGBoost, precision rose from 0.806 to 0.884 and F1 from 0.826 to 0.871, with PR-AUC increasing from 0.869 to 0.885. Random Forest improved in F1 (0.806 to 0.821), ROC-AUC (0.969 to 0.979), and PR-AUC (0.858 to 0.866), while KNN's F1 rose from 0.820 to 0.844. Recall remained largely unchanged (within one transaction) for Random Forest and KNN and increased slightly for XGBoost (0.847 to 0.857), indicating that tuning reduced false alarms without sacrificing fraud detection. KNN was the exception in ranking metrics, with ROC-AUC (0.934 to 0.923) and PR-AUC (0.780 to 0.767) declining slightly. After tuning, XGBoost achieved the best performance across precision, recall, F1, and PR-AUC, followed by Random Forest and KNN. Finally, based on their individual performance, the best two ML algorithms were chosen to design the ensemble learning and their predictions were combined using the weighted voting classifier. 0.7 and 0.3 weight values were taken. The 0.7 weight value was assigned to the XGBoost whereas 0.3 weight value was assigned to the KNN. After this process, the performance evaluation of the proposed method based on ensemble learning is shown in Table 6.

Table 5: Evaluation of the ML Algorithms after Hyper-Parameter Tuning using the Optuna Optimizer

Model	Accuracy	Precision	Recall	F1-Score	ROC-AUC	PR-AUC
Random Forest	0.999403	0.847826	0.795918	0.821053	0.978507	0.865778
KNN	0.999473	0.861702	0.826531	0.843750	0.923343	0.766590
XGBoost	0.999561	0.884211	0.857143	0.870466	0.975540	0.885035

Table 6: Evaluation of the Ensemble Learning Approach

Model	Accuracy	Precision	Recall	F1-Score	ROC-AUC	PR-AUC
Weighted XGBoost + KNN	**0.999596**	0.903226	0.857143	**0.879581**	0.975524	0.882359

Figure 8 compares the optimized individual models with the final weighted voting ensemble combining XGBoost and KNN (weights of 0.7 and 0.3, respectively). The ensemble achieved the highest precision (0.903) and F1-score (0.880) of all models, improving on the best individual model, XGBoost (precision = 0.884, F1 = 0.871). Its recall (0.857) was identical to that of XGBoost and higher than those of KNN (0.827) and Random Forest (0.796), which corresponds to 84 of 98 fraudulent transactions detected. The gain in precision reflects a reduction of approximately two false positives (from 11 to 9), meaning fewer legitimate transactions were wrongly flagged without any loss of fraud detection. ROC-AUC (0.976) was identical to that of XGBoost, and PR-AUC (0.882) was marginally lower than XGBoost's (0.885), indicating that the ensemble improved the final classification decisions rather than the underlying probability ranking. Accuracy was approximately 0.9996 for all models and is not informative given the severe class imbalance. Overall, the ensemble offered the best precision-recall balance, although the margin over tuned XGBoost is small and, given only 98 fraudulent test cases, should be interpreted cautiously.

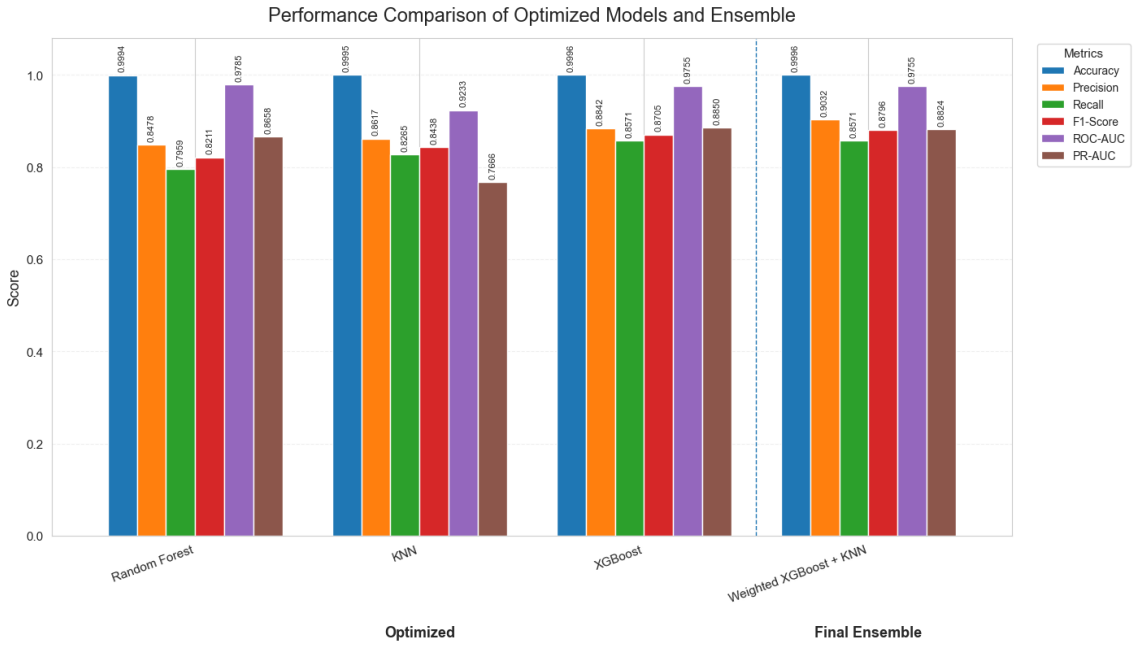


Figure 8: Evaluation of the Ensemble Learning Approach

To interpret the optimized XGBoost model, SHAP values were computed and summarized in Figures 9 and 10.

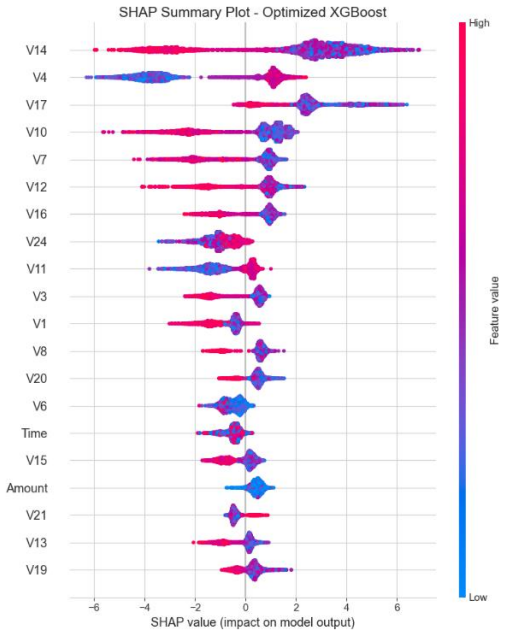


Figure 9: SHAP Summary Plot

The global importance plot shows that predictions are dominated by a small set of features: V14 (mean |SHAP| ≈ 3.2), V4 (≈ 2.7), and V17 (≈ 2.5), followed by V10, V7, V12, and V16. The remaining features, including Time and Amount, contributed substantially less (mean |SHAP| below ≈ 0.6). The beeswarm plot shows that these leading features exert strong and directional effects: extreme feature values produce SHAP contributions of several log-odds units toward either the fraud or the legitimate class, with a clear separation between low and high feature values for V14, V4, and V17. Wide, overlapping bands for some features suggest that their effect depends on interactions with other variables. Because V1 to V28 are anonymized principal components, the findings identify which components drive detection but do not allow direct business interpretation. These results indicate that the

model's fraud decisions rely mainly on a handful of components, consistent with the strong classification performance reported earlier.

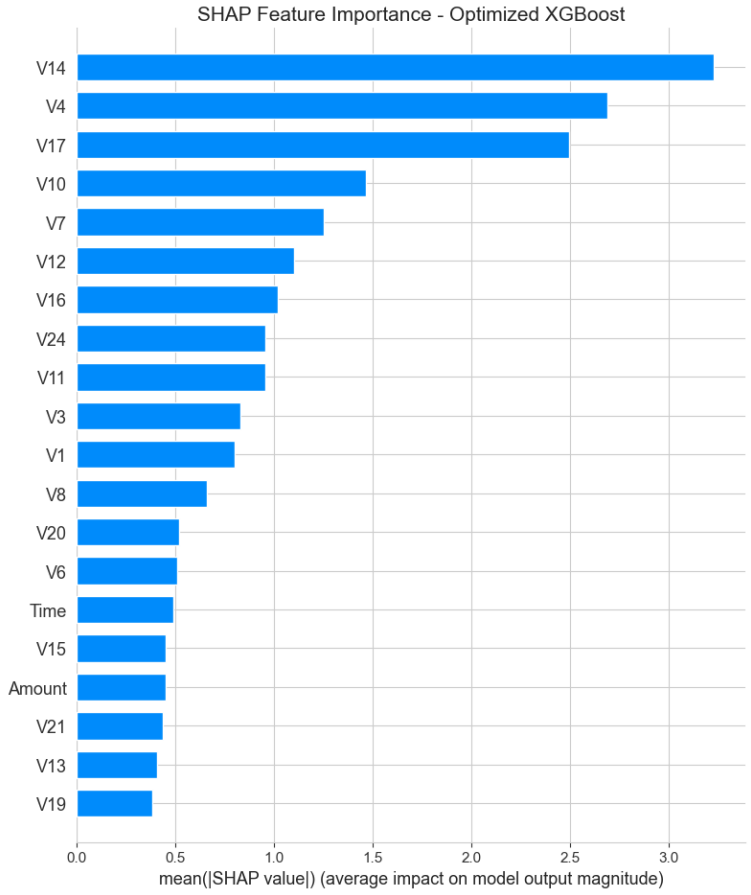


Figure 10: Feature Importance Analysis using SHAP

3.3 Comparative Analysis

Table 7: Comparative Analysis

Model / Method	Accuracy	Precision	Recall	F1-Score	ROC-AUC	PR-AUC
SMOTE – Probability-based kNN [3]	0.8226	0.8360	—	—	—	—
SMOTE – XGBoost [3]	—	—	0.6410	0.7430	—	—
ADASYN – Probability-based kNN [3]	0.9670	0.8297	0.7998	0.8144	—	—
ADASYN – XGBoost [3]	—	0.8830	—	—	—	—
RF Algorithm [11]	0.9992	—	—	0.7727	—	—
XGBoost [11]	0.9994	—	—	0.8449	—	—
Proposed Optimized – Random Forest	0.999403	0.847826	0.795918	0.821053	0.978507	0.865778
Proposed Optimized –	0.999473	0.861702	0.826531	0.843750	0.923343	0.766590

KNN						
Proposed Optimized – XGBoost	0.999561	0.884211	0.857143	0.870466	0.975540	0.885035
Proposed Weighted XGBoost + KNN	**0.999596**	**0.903226**	**0.857143**	**0.879581**	**0.975524**	**0.882359**

Table 7 compares the proposed models with results reported in earlier work on the same credit card fraud problem. The proposed optimized models matched or exceeded the reported baselines on every metric available for comparison. The optimized XGBoost achieved an F1-score of 0.871, compared with 0.845 for XGBoost in [11] and 0.743 for SMOTE-XGBoost in [3], and its recall (0.857) was substantially higher than the 0.641 reported for SMOTE-XGBoost in [3]. Similarly, the optimized Random Forest (F1 = 0.821) outperformed the Random Forest of [11] (F1 = 0.773), and the optimized KNN (precision = 0.862, recall = 0.827, F1 = 0.844) exceeded both the SMOTE- and ADASYN-based probability kNN models of [3] (F1 = 0.814 for the latter). The precision of the optimized XGBoost (0.884) was essentially equal to that of ADASYN-XGBoost in [3] (0.883). The proposed weighted XGBoost-KNN ensemble achieved the best overall results (precision = 0.903, recall = 0.857, F1 = 0.880), surpassing the best previously reported precision and F1 in the table. Accuracy was uniformly high (0.999 or above) across the studies and, given the severe class imbalance, is not discriminative. Moreover, the proposed work additionally reports ROC-AUC (up to 0.976) and PR-AUC (up to 0.885), which were not available for the earlier studies. These comparisons should be interpreted with caution because the cited studies differ in data partitioning, preprocessing, and resampling strategies, and the test set contains only 98 fraudulent transactions, so small differences may not be statistically meaningful.

4. Conclusion and Future Scope of the Work

In this study, a model for detecting credit card fraud is developed using the ensemble learning approach to identify legitimate and fraudulent transactions. The dataset was collected from the open-source Kaggle repository. The min-max and SMOTE techniques were applied to the dataset to prepare it for ML algorithms. The pre-processed dataset was given to five ML algorithms, including LR, DT, RF, KNN, and XGBoost, and evaluated using various performance metrics. The key finding from this evaluation shows that RF, KNN, and XGBoost outperform LR and DT algorithms. The outperforming ML algorithms were chosen for hyper-parameter tuning using the Optuna optimizer. After hyper-tuning, the best two ML algorithms were selected in the ensemble learning, and their predictions were combined using the weighted voting classifier. We have assigned the 0.7 weight value to the best ML algorithm, whereas the second-best ML algorithm was given a 0.3 weight value. The simulation results show that the proposed method achieves an accuracy value of 0.99696, a precision value of 0.903226, a recall value of 0.879581, an F1-score value of 0.879581, an ROC-AUC value of 0.975524, and a PR-AUC value of 0.882359. The comparative analysis shows that the proposed method outperforms the existing approaches developed by Alarfaj et al. [3] and Karunya et al. [11]. In the future, we will explore other data balancing and optimization approaches for balance the data classes and hyper-tune the parameters of the ML algorithms.

References

[1] N. Baisholan, J. E. Dietz, S. Gnatyuk, M. Turdalyuly, E. T. Matson, and K. Baisholanova, “A Systematic Review of Machine Learning in Credit card fraud detection under original class imbalance,” *Computers*, vol. 14, no. 10, p. 437, Oct. 2025, doi: 10.3390/computers14100437. Available: https://doi.org/10.3390/computers14100437

[2] R. B. Sulaiman, V. Schetinin, and P. Sant, “Review of Machine Learning Approach on Credit Card Fraud Detection,” *Human-Centric Intelligent Systems*, vol. 2, no. 1–2, pp. 55–68, May 2022, doi: 10.1007/s44230-022-00004-0. Available: https://doi.org/10.1007/s44230-022-00004-0

[3] F. K. Alarfaj, I. Malik, H. U. Khan, N. Almusallam, M. Ramzan, and M. Ahmed, "Credit card fraud detection using state-of-the-art machine learning and deep learning algorithms," *IEEE Access*, vol. 10, pp. 45498–45510, 2022, doi: 10.1109/ACCESS.2022.3166891.

[4] E. Ileberi, Y. Sun, and Z. Wang, "Performance evaluation of machine learning methods for credit card fraud detection using SMOTE and AdaBoost," *IEEE Access*, vol. 9, pp. 165286–165294, Jan. 2021, doi: 10.1109/access.2021.3134330. Available: https://doi.org/10.1109/access.2021.3134330

[5] A. R. Khalid, N. Owoh, O. Uthmani, M. Ashawa, J. Osamor, and J. Adejoh, "Enhancing Credit Card Fraud Detection: An ensemble Machine learning approach," *Big Data and Cognitive Computing*, vol. 8, no. 1, p. 6, Jan. 2024, doi: 10.3390/bdcc8010006. Available: https://doi.org/10.3390/bdcc8010006

[6] H. S. Alsagri, "Hybrid Machine Learning-Based Multi-Stage framework for detection of credit card anomalies and fraud," *IEEE Access*, vol. 13, pp. 77039–77048, Jan. 2025, doi: 10.1109/access.2025.3565612. Available: https://doi.org/10.1109/access.2025.3565612

[7] N. Baisholan, J. E. Dietz, S. Gnatyuk, M. Turdalyuly, E. T. Matson, and K. Baisholanova, "FraUdX AI: An interpretable machine learning framework for credit card fraud detection on imbalanced datasets," *Computers*, vol. 14, no. 4, p. 120, Mar. 2025, doi: 10.3390/computers14040120. Available: https://doi.org/10.3390/computers14040120

[8] R. K. Gupta *et al.*, "Enhanced framework for credit card fraud detection using robust feature selection and a stacking ensemble model approach," *Results in Engineering*, vol. 26, p. 105084, Apr. 2025, doi: 10.1016/j.rineng.2025.105084. Available: https://doi.org/10.1016/j.rineng.2025.105084

[9] L. Theodorakopoulos, A. Theodoropoulou, A. Tsimakis, and C. Halkiopoulos, "Big Data-Driven distributed machine learning for scalable credit card fraud detection using PySpark, XGBoost, and CatBoost," *Electronics*, vol. 14, no. 9, p. 1754, Apr. 2025, doi: 10.3390/electronics14091754. Available: https://doi.org/10.3390/electronics14091754

[10] M. Zavvar *et al.*, "A hybrid deep learning framework using synthetic oversampling, autoencoder, convolutional neural networks, and an attention mechanism for credit card fraud detection," *Journal of Big Data*, vol. 13, no. 1, Dec. 2025, doi: 10.1186/s40537-025-01331-2. Available: https://doi.org/10.1186/s40537-025-01331-2

[11] K. R. V. Karunya, S. Ganesh, D. Muralidharan, G. R. Brindha, and M. Thiruvengadam, "Credit card fraud data analysis and prediction using machine learning algorithms," *Security and Privacy*, vol. 8, no. 3, Art. no. e70043, 2025, doi: 10.1002/spy2.70043

[12] Kaggle, "Credit Card Fraud Detection," *www.kaggle.com*, 2018. https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud (accessed Oct. 25, 2026).

[13] J. Chung and K. Lee, "Credit Card Fraud Detection: An Improved Strategy for high Recall using KNN, LDA, and linear regression," *Sensors*, vol. 23, no. 18, p. 7788, Sep. 2023, doi: 10.3390/s23187788. Available: https://doi.org/10.3390/s23187788

[14] S. K. Jain and S. Asha, "Credit Card Fraud Detection System using SMOTEENN and Adaptive XGBoost and comparing the result with state-of-art-technique," *IEEE International Conference for Convergence in Technology (I2CT)*, pp. 1–7, Apr. 2024, doi: 10.1109/i2ct61223.2024.10543887. Available: https://doi.org/10.1109/i2ct61223.2024.10543887

[15] N. Baisholan, J. E. Dietz, S. Gnatyuk, M. Turdalyuly, E. T. Matson, and K. Baisholanova, "FraUdX AI: An interpretable machine learning framework for credit card fraud detection on imbalanced datasets," *Computers*, vol. 14, no. 4, p. 120, Mar. 2025, doi: 10.3390/computers14040120. Available: https://doi.org/10.3390/computers14040120

[16] H. Rathore and R. Ratnawat, "A robust and efficient machine learning approach for identifying fraud in credit card transaction," *International Conference on Smart Electronics and Communication (ICOSEC)*, pp. 1486–1491, Sep. 2024, doi: 10.1109/icosec61587.2024.10722387. Available: https://doi.org/10.1109/icosec61587.2024.10722387

[17] H. Du, L. Lv, H. Wang, and A. Guo, "A novel method for detecting credit card fraud problems," *PLoS ONE*, vol. 19, no. 3, p. e0294537, Mar. 2024, doi: 10.1371/journal.pone.0294537. Available: https://doi.org/10.1371/journal.pone.0294537

[18] K. Kandi and A. García-Dopico, "Enhancing performance of credit card model by utilizing LSTM networks and XGBOOST algorithms," *Machine Learning and Knowledge Extraction*, vol. 7, no. 1, p. 20, Feb. 2025, doi: 10.3390/make7010020. Available: https://doi.org/10.3390/make7010020