International Journal of Statistical Sciences *ISSN 1683-5603*
Vol. 26(1), March, 2026, pp 95-106
DOI: https://doi.org/10.3329/ijss.v26i1.88842
© 2026 Dept. of Statistics, Univ. of Rajshahi, Bangladesh

Defending Digital Wallets: A Multi-Method Approach to Detect Fraudulent Card Transactions

Md. Sojib Hossain[1], Ripon Mahmud Akash[1], Md. Mostafizur Rahman[1,*] and M. Sayedur Rahman[2]

[1]Data Mining and Environmental Research Group, Department of Statistics, University of Rajshahi, Rajshahi-6205, Bangladesh
Email: sojibhossainstat.ru@gmail.com
[2]North Bengal International University, Rajshahi, Bangladesh
[*]Correspondence should be addressed to Md. Mostafizur Rahman
(Email: mostafiz_bd21@yahoo.com)

[Received August 3, 2025; Accepted September 9, 2025]

Abstract

Credit card fraud poses a significant threat to the financial industry, causing substantial losses for both individuals and institutions. This paper presents a comprehensive investigation into the application of machine learning techniques to enhance credit card fraud detection. We explore a diverse range of algorithms, including K-Nearest Neighbors (KNN), Naive Bayes, Logistic Regression, and Random Forest, to accurately identify fraudulent transactions. To address the inherent class imbalance in fraud detection datasets, we employ the Synthetic Minority Over-sampling Technique (SMOTE). Additionally, we delve into Exploratory Data Analysis (EDA) to gain valuable insights into the data distribution and potential patterns that may aid in fraud detection. To optimize model performance, we meticulously tune hyperparameters, fine-tuning the algorithms to achieve optimal results. The result confirmed that the Random Forest model is more advantageous than the other models considered (K-Nearest Neighbor, Naïve Bayes, and Logistic Regression) because it provides high accuracy and balance in the ability to predict classes. Our empirical evaluation demonstrates the effectiveness of the proposed approach, highlighting its potential to significantly improve the accuracy and precision of credit card fraud detection systems.

Keywords: Credit Card Fraud Detection, K-Nearest Neighbors (KNN), Naive Bayes, Logistic Regression and Synthetic Minority Over-sampling Technique.

AMS Classification: Primary 62H30; Secondary 62P20, 62J12, 68T10, 68T05.

1. Introduction

Credit card fraud has increased in conjunction with the proliferation of credit cards, whether used online or offline, posing problems for both the consumer and the bank that issued the card. Also at risk is the difficulty of detecting fraud as e-business grows and regulations are still broken. To

identify credit card fraud, the article will demonstrate how machine learning algorithms use a variety of models, including KNN, NB, LR, RF, SMOTE, and Hyperparameter Optimization. at fact, several studies found that the algorithms were successful at identifying fraud. Its performance has received appreciation. This method has been linked to high accuracy and precision in identifying fraudulent transactions (O. et al., 2024; Kumar & Goswami, 2024; Manorom et al., 2024) [1] [2] [3]. Additionally, Logistic Regression and Naïve Bayes have been employed in predicting fraudulent activities. Logistic Regression is frequently recognized for its efficacy in addressing binary classification challenges (Itoo et al., 2020; Kumar et al., 2023) [4] [5]. K-Nearest Neighbors provides a straightforward comparison of new transactions with established data points (Itoo et al., 2020; Manorom et al., 2024) [4] [3]. SMOTE (Synthetic Minority Over-sampling strategy) is a technique used to balance the dataset and improve the effectiveness of the model in fraud detection when faced with imbalanced datasets, which are common in fraud detection (Itoo et al., 2020) [4]. Furthermore, by changing these algorithms' parameters to achieve the highest possible efficiency, Hyperparameter Optimization enhances the algorithm's efficiency (Karthikeyan et al., 2019) [6]. This research adds to the existing body of knowledge in the aspect of credit card fraud detection through: (1) systematic comparative analysis of a dozen machine learning algorithms under standard experimental conditions (Khan & Palaniswamy, 2024); (2) empirical validation of the effectiveness of SMOTE in improving fraud detection performances in the face of extremely imbalanced classes; (3) quantifying the effect of hyperparameter optimization on the observed performance of the models; and (4) proposed evidence-based recommendations for implementing machine learning-based fraud detection systems in the operational financial environment. Findings presented here have serious implications for the financial institution in their attempt to improve fraud detection while shrinking false positives and thereby maintaining a high-quality customer experience (Bolton & Hand, 2002) [8]. Therefore, by integrating the advantages and disadvantages of several machine learning models and approaches, this research develops a model for detecting credit card fraud. The banking industry will subsequently have more defense against the ongoing risk of credit card fraud as a result.

2. Literature Review

Credit card fraud detection remains one of the increasingly critical applications in the financial industry due to the rapidly growing use of e-commerce and online transactions. Many machine learning algorithms have been applied to this problem with different strengths and weaknesses. KNN is a very simple yet powerful algorithm for credit card fraud detection. It classifies new transactions based on their similarity to existing ones, however, it requires choosing 'k' and distance metric which can be naive [1][9]. This is a type of naive classifier based on independence assumption among predictors so that simplicity becomes the reason for its quick implementation. In real data sets, such an assumption does not hold so accuracy may be affected [10][9]. Logistic Regression is one of the most popular statistical methods used for binary classification problems such as detecting fraud. It predicts the probability that a transaction is fraudulent using input features. Interpretability is a major asset, and high accuracy in many cases, but it may not learn as well as other more complex algorithms [1] [9]. Another reason K-Nearest Neighbors and Logistic Regression are compared is their effectiveness in detecting fraud. In general, KNN performs better than Logistic Regression in terms of precision, recall, and overall accuracy as well as Naïve Bayes especially when data set imbalances taken into consideration [15] [16]. Even though it has very high accuracy, Logistic Regression sometimes with imbalanced data misleadingly inflates apparent levels of fraud [15]. In fraud detection tasks, the performance and reliability of Random Forest

always meets or exceeds that of the other models in this task area (e.g., Logistic Regression, Naïve Bayes and KNN). Some studies found accuracy rates exceeding 99%, which encourages its use in industry among financial companies for evaluating fraudulent activity [1] [17] [16]. In fraud detection datasets, SMOTE (Synthetic Minority Over-sampling Technique) helps alleviate the effects of class imbalance by producing synthetic examples for the minority class. Using SMOTE allows for classifier performance improvement from the balancing of the datasets [13] [14]. XGBoost, an open-source gradient boosting framework, can introduce hyperparameter tuning to improve model performance. The main benefits of XGBoost are the speed and performance of this model when deployed in the same tasks as the other models mentioned here, especially with respect to large scale data sets involving fraud detection [13] [14]. The implementation of the machine learning algorithms has exhibited diverse success rates in credit card fraud detection. While Random Forest has always provided the most effective results, with studies reporting accuracy scores as high as 99.95% [9] [12]. SMOTE and hyperparameter tuning further improved detection performance from class imbalance and hyperparameter optimization [13] [14]. XGBoost is consistently acknowledged as one of the superior models for fraud detection due to its effectiveness, with reported results suggesting almost perfect accuracy - once hyperparameters are optimized - specifically for credit card fraud detection [18] [19]. We also found that XGBoost combined with hyperparameter tuning and ensemble methods improved the model's capabilities in detecting fraudulent transactions as well [20] [19].

3. Data and Methodology

3.1 Data Description

This dataset of simulated credit card transactions includes both authentic and fraudulent transactions from January 1, 2019, to December 31, 2020. It covers 1000 clients' credit cards when they make purchases from a pool of 800 businesses [24]. A wealth of features in the dataset utilized in this study can be used to spot fraudulent transactions. The following is a synopsis of the main columns is given in Table 1.

Table 1: Characteristics of dataset

Sl	Column Name	Description	Sl	Column Name	Description
1	index	Unique identifier for each row	13	Zip	ZIP code of cardholder
2	trans_date_trans_time	Transaction date and time	14	Lat	Latitude of cardholder's location
3	cc_num	Credit card number	15	Long	Longitude of cardholder's location
4	merchant	Merchant name	16	city_pop	Population of cardholder's city
5	category	Merchant category	17	Job	Occupation of cardholder
6	amt	Transaction amount	18	Dob	Date of birth of cardholder
7	first	First name of cardholder	19	trans_num	Transaction number
8	last	Last name of cardholder	20	unix_time	UNIX timestamp of transaction
9	gender	Gender of cardholder	21	merch_lat	Latitude of merchant's location
10	Street	Street address of cardholder	22	merch_long	Longitude of merchant's location
11	City	City of cardholder	23	is_fraud	Fraud flag (target variable)
12	State	State of cardholder			

3.2 Machine learning Algorithms

3.2.1 K-Nearest Neighbor algorithms

The classifier K-nearest neighbor algorithm is an instance-based learning technique that classifies data using a similarity metric, such as the Minkowski, Manhattan, or Euclidean distance functions. Whereas the final measurement of distance does better with categorical data, the first two do well with continuous variables. This investigation will employ the Euclidean distance metric in the KNN classifier [21]. The following formula is used to get the Euclidean distance (Dij) between two input vectors (Xi, Xj).

$$D_{ij} = \sqrt{\sum_{k=1}^{n} \left(X_{ik} - X_{jk}\right)^2} \quad (1)$$

The Euclidean distance between an input data point and a current data point is computed for each data point in the dataset. Items with the lowest k-distance from the input data point were chosen after the estimated distances were sorted in increasing order. These components are used to determine the majority class, which is then returned for the input data point. Performance was best for k = 3, and parameter adjustment for k was done for k = 1, 3, 5, 7, 9, and 11. As a result, the classifier chose k = 3.

3.2.2 Naive Bayes Classifier

The Bayes theorem, which categorizes decisions according to their highest probability, is the foundation of the Naïve Bayes statistical model. Using known values, the Bayesian probability assesses the unknown probabilities. Additionally, it allows for the application of logic and past knowledge to ambiguous assertions. The dataset's characteristics are assumed to be conditionally independent in this technique. Conditional probabilities and the binary classification classes (fraud and non-fraud) form the foundation of the Naive Bayes classifier [23].

$$A = \frac{P(f_k|c_i)*P(c_i)}{P(f_k)} \qquad \text{k=1,2,3…… n \& i=1, 2} \quad (2)$$

$$P(f_k|c_i) = \prod_{i=1}^{n} P(f_k|c_i) \quad (3)$$

where n is the maximum number of features (up to 30), P (ci | fk) is the probability that feature value fk belongs to class ci, P (fk | ci) is the probability that feature value fk is generated under class ci, P (ci) is the probability that class ci will occur, and P (fk) is the probability that feature value fk will occur. Using the Bayesian classification rule, the classifier classifies data into binary categories.

If P (c1 | fk) > P (c2 | fk) then Classification c1
If P (c1 | fk) < P (c2 | fk) then Classification c2

Ci is the target class for classification where c1 is the negative class (non-fraud cases) and c2 is the positive class (fraud cases).

3.2.3 Logistic Regression Classifier

Logistic Regression estimates the probability of a binary response using a modeling approach based on features, or variables, either single or multiple variables. It gives the most suitable value for fitting the the sigmoid a nonlinear function. In (7) and (8) of the manuscript [21], the sigmoid function (σ) and its input (x) are shown.

$$\sigma(x) = \frac{1}{(1+l^{-x})} \tag{4}$$

$$x = w_0 z_0 + w_1 z_1 + w_2 z_2 + \cdots w_n z_n \tag{5}$$

The input data, vector z, is multiplied by the best coefficients, w, in each element. The outcome is a single integer that classifies the target class. The sigmoid value is regarded as a 1 if it is bigger than 0.5, and as a 0 else. The classifier is trained using an optimization technique to determine the optimal fit of the parameters. The classifier's performance was assessed using the method of gradient ascending (6) and improved randomly generated gradient ascent methods for optimization.

$$w = w + \alpha \nabla_w \, f(w) \tag{6}$$

The gradient ascent's magnitude of movement is the parameter ∇. Until a stopping requirement is satisfied, or it is established that the parameter values are constant, the steps are repeated. For iterations 50 through 1000, the methods for optimization are examined in order to determine which parameters are convergent. That is, if the parameters were constantly fluctuating or stabilized. It was accepted that stable parameter values were attained after 100 repetitions. Instead of updating the classifier all at once, stochastic gradient ascent updates it more gradually as new data is received. All weights are set to 1 at the beginning of the algorithm. The gradient ascent is then computed for each feature value in the dataset. The gradient is calculated and the weights vector is updated by the product of alpha. After that, the weights are given back. Because it adjusts weights by looking at only one occurrence at a time, stochastic gradient ascent reduces calculating cost and is suitable for this study's massive data set.

3.2.4 Random Forest Algorithm

One popular kind of supervised learning method is Random Forest. Although it is mostly used for classification issues, it may be applied to regression and classification applications. The Random Forest approach builds decision trees from the sample data and produces predictions from all of the sample data, much as forests are really made up of trees. An ensemble approach is the Random Forest. The Random Forest method has an advantage over a single decision tree since it prevents overfitting inside the trees by averaging the forecasts of all the trees. The Random Forest Algorithm [22] steps are as follows:

1. Make use of the credit card fraud dataset you examined on Kaggle and choose a few samples at random.
2. Utilize the sample data to create the Decision Trees, which will utilize the data to categorize the cases as either fraud or non-fraud.
3. Split the nodes to create the decision trees; the split that produced the most information gain was designated as the root node, which was then utilized to consistently identify the case as either fraud or non-fraud.
4. Proceed with a majority vote; for instance, the Decision Trees may vote to generate 0 as the output, classifying the situation as non-fraudulent
5. Finally, we find the accuracy, precision, recall, and F1 -score for both fraud and non-fraud cases.

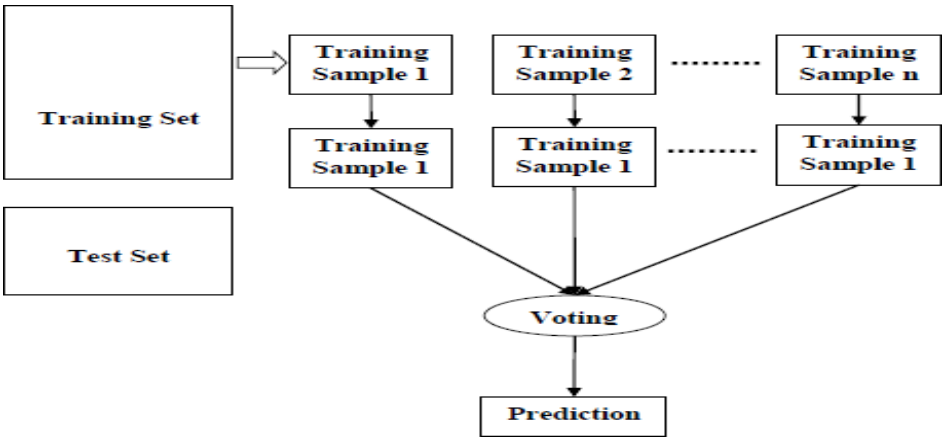

Figure 1: Random Forest Algorithm Flowchart

Algorithm Random Forest:

To generate c classifiers:

For i=1 to c do

Randomly select the training data D with replacement to produce Di

Create a root node N containing Di and cell

Build Tree(N)

End for

Majority Vote

Build Tree(N)

Randomly select x% of all the possible splitting features in N

Select the features F that has the highest Information

A gain for further splitting

Gain (T,X)=Entropy (T)-Entropy(T,X)

Now to calculate the entropy we use,

$$E(S) = \sum_{i=1}^{c} -P_i logP$$

Create f child nodes

For i=1 to f do

Set contents f N to Di

Call Build Tree(Ni)

End for

End

3.2.5 Addressing Class Imbalance with SMOTE

Dealing with Class Imbalance through SMOTE (Synthetic Minority Oversampling Technique) is a technique that improves your machine learning model performance by dealing with class imbalance. SMOTE creates synthetic samples for the minority classes, enhances performance, and mitigates bias created in imbalanced datasets. SMOTE generated synthetic samples by interpolating between an instance of the minority class (X) and its k-nearest neighbors (Xneighbors) according to the following formula:

$$\text{New Sample}=X+\delta\times(Xneighbors-X) \tag{7}$$

where δ is a random scalar in the interval [0,1] so that diverse synthetic data is generated.

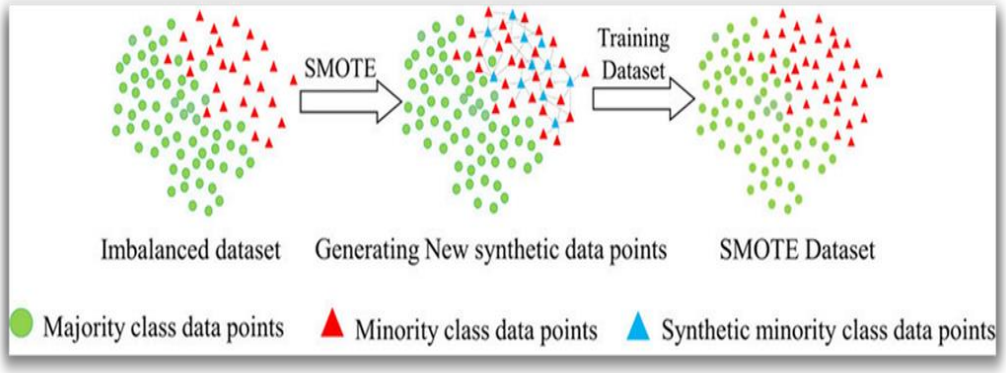

Figure 2: SMOTE Algorithm Flowchart

3.2.6 XGBoost Algorithm

The XGBoost algorithm uses an additive training process to maximize the objective function. In other words, at each step of the optimization model, the result of that step is used for the next step. A mathematical expression of the t-th objective function for the XGBoost model is shown below [23]:

$$F_0^i = \sum_{k=1}^{n} l(y_{ki}\hat{y}^{i-1} + f_i(x_k) + R(f_i) + C \tag{8}$$

where the t-th iteration's loss term is represented with l, C is a constant, and R is the model's regularization parameters expressed as:

$$R(f_i) = \gamma T_i + \frac{\lambda}{2} \tag{9}$$

As a rule of thumb, the larger the values of customization parameters g and l are, the simpler the tree structure/making the tree structure simpler is. The first g and second h derivatives of the model can be represented as:

$$g_i = \delta\widehat{y_k^{i-1}}\left(y_j, \widehat{y_k^{i-1}}\right) \tag{10}$$

$$h_j = \delta^2\widehat{y_k^{i-1}}\left(y_j, \widehat{y_k^{i-1}}\right) \tag{11}$$

From the following, you can get the solution:

$$w_j^* = -\frac{\sum g_t}{\sum h_t + \lambda} \tag{12}$$

$$F_0^* = -\frac{1}{2}\sum_{j=1}^{T}\frac{(\sum g)^2}{\sum h+\lambda}\gamma T \tag{13}$$

where $Fo*$ represents the score of loss function, and $w*j$ denotes the solution of weights.

4. Results and Discussions

The performance of different traditional classification methods and also advanced techniques was examined by overall accuracy and F-score statistics. This performance also checked by ROC curve. These are given below.

Table 2: Table of Traditional ML Algorithm Results

Serial No	Machine Learning Model	AUC	F1_Score
1	K-Nearest Neighbors	0.98	0.73
2	Naive Bayes	0.87	0.63
3	Logistic Regression	0.85	0.70
4	Random Forest	0.9822	0.65
5	SMOTE	0.97	0.78
6	XGBoost	0.9963	0.98

From table 2 we found that XGBoost is the highest performing algorithm with excellent performance (AUC = 0.9963, F1 = 0.98), providing excellent instance ranking and optimal precision-recall trade-off critical in minority class discovery. Performance comparison indicates broad variations: Random Forest performed well in AUC (0.9822) but poorly in F1 score (0.65), indicating weakness in detecting minority classes and pointing towards the performance trap wherein high ability in discrimination is not enough to guarantee effective minority class administration. K-Nearest Neighbors reported consistent performance (AUC = 0.98, F1 = 0.73), while baseline classifiers like Naive Bayes (AUC = 0.87, F1 = 0.63) and Logistic Regression (AUC = 0.85, F1 = 0.70) reported moderate performance at the expense of imbalance sensitivity.

SMOTE stands at the center in class imbalance primarily enhancing model performance (AUC = 0.97, F1 = 0.78) with synthetic oversampling and enabling better minority instance learning and balanced prediction outcomes. The results confirm that F1 score and ROC AUC have complementary roles: AUC enables threshold-free ranking quality estimation and F1 score useful information on class skew classification accuracy. For high-stakes problems like medical diagnosis or fraud detection, F1 score will prove more helpful than AUC in combination. The findings highlight the need for utilizing advanced algorithms and preprocessing techniques for handling class imbalance effectively. Future research must focus on context-aware testing with greater emphasis on metrics like F1 score that offer better predictive performance under real-world conditions.

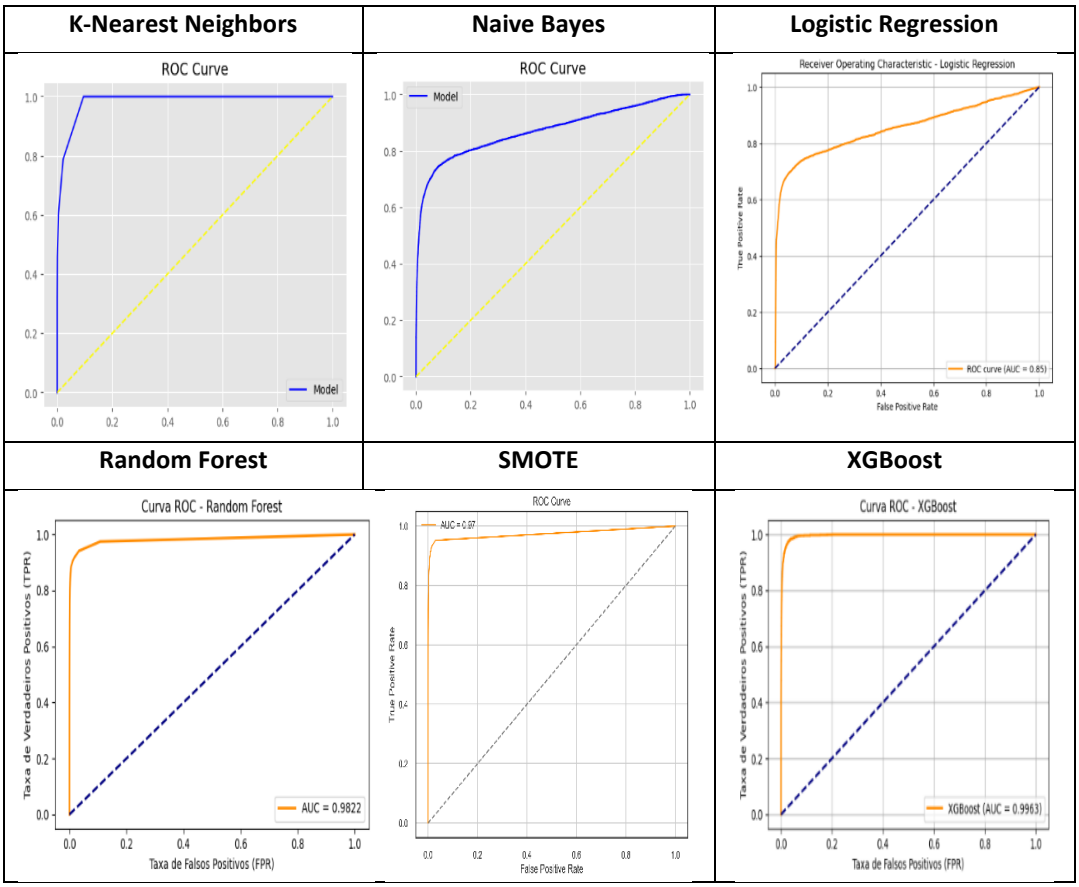

Figure 3: ROC Curves for Various Machine Learning Algorithms.

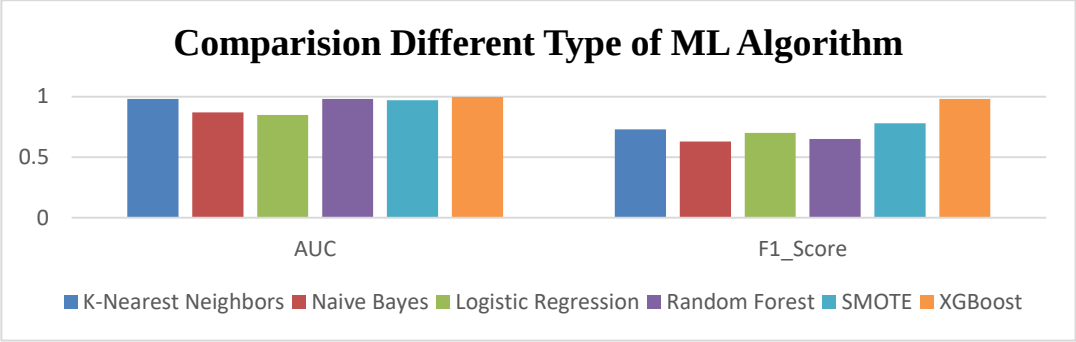

Figure 4: Comparison of AUC-ROC and F1-Score Across Different Machine Learning Algorithms

5. Conclusions

AUC-ROC in conjunction with F1 score helps us evaluate the performance of traditional machine learning algorithms. The XGBoost algorithm had the best outcomes with highest AUC (0.9963) and F1 score (0.98). This indication of level of class separation as well as balanced precision-recall functionality is exceptional.K-Nearest Neighbors (KNN) did perform well for AUC (0.98), but low F1 score of 0.73 indicates an inability to solve class imbalance or a misclassification problem; essentially a form of poor predictive performance. SMOTE, which is an algorithm for balancing data for the purpose of improvement, produced the highest F1 score gain (0.78) and high AUC (0.97). All of which indicates a substantial potential for improvement with regard to model sensitivity. Random Forest had a good AUC (0.9822) but very low F1 score 0.65, which indicates a problem with precision or recall, not its discrimination ability. Logistic Regression and Naive Bayes had lower than average AUC's (0.85 and 0.87) and average F1 scores (0.70 and 0.63); which means they didn't do well in this data set. In conclusion XGBoost did the best job of display, level of discrimination ability, and balance in predictions. SMOTE is another good option to see potential performance improvement with uneven probability distribution. Using only the AUC-ROC machine learning problem, makes performance look better than in fact it is, while the F1 score provides insight to real problems in overall machine learning model.

Although it has been established that traditional machine learning algorithms may be effective in detecting credit card fraud, it is clear from the results of this analysis that further improvement can be done in using these algorithms to detect fraud with fragile transactions and highly imbalanced datasets. The next phase of analysis can test different deep learning methods to compare fraud in transactions with better performance capabilities.

Acknowledgement: The authors would like to thank the reviewer for the time and effort dedicated to reviewing this paper and offering useful suggestions.

References

[1] Ayodele, K. J., Oluwajuwon, B. O., and Auodele, A. A. (2024). Credit Card Fraud Detection Using Machine Learning Algorithms. British Journal of Computer, Networking and Information Technology. https://doi.org/10.52589/bjcnit-ydijnxg2

[2] Kumar, J., and Goswami, P. (2024). Credit Card Fraud Detection Using Machine Learning. International Journal for Multidisciplinary Research. https://doi.org/10.36948/ijfmr.2024.v06i02.19237

[3] Manorom, P., Detthamrong, U., and Chansanam, W. (2014). Comparative Assessment of Fraudulent Financial Transactions using the Machine Learning Algorithms Decision Tree, Logistic Regression, Naïve Bayes, K-Nearest Neighbor, and Random Forest. Engineering, Technology & Applied Science Research. https://doi.org/10.48084/etasr.7774

[4] Itoo, F., M., and Singh, S. (2020). Comparison and analysis of logistic regression, Naïve Bayes and KNN machine learning algorithms for credit card fraud detection. International Journal of Information Technology. https://doi.org/10.1007/s41870-020-00430-y.

[5] Kumar, P., Kiran, P., Kouashik, S., Koushik, V., Kumar, K., Chaitanya, A., and Kalyani, P. (2023). Credit Card Fraud Detection Using Machine Learning Algorithms. International Journal for Research in Applied Science and Engineering Technology. https://doi.org/10.22214/ijraset.2023.57429.

[6] Karthikeyan, K., Raj, K., Ramaganesh, S., Parthasarathi, P., and Suguna, N. (2019). Credit Card Fraud Detection using Machine Learning. International Journal of Engineering and Advanced Technology. https://doi.org/10.32628/IJSRST196271

[7] Aslam, F. (2024). Advancing Credit Card Fraud Detection: A Review of Machine Learning Algorithms and the Power of Light Gradient Boosting. American Journal of Computer Science and Technology. https://doi.org/10.11648/ajcst.20240701.12

[8] Bolton, R. J., and Hand, D. J. (2002). Statistical Fraud Detection: A Review. Statistical Science, https://doi.org/10.1214/ss/1042727940

[9] Khan, M., Shaikh, S., Shaikh, M., Khatri, K., Rauf, M., Kalhoro, A., and Adnan, M. (2023). The Performance Analysis of Machine Learning Algorithms for Credit Card Fraud Detection. Int. J. Online Biomed. Eng. https://doi.org/10.3991/ijoe.v19i03.35331

[10] Credit Card Fraud Detection using Machine Learning Algorithms (2022). International Journal of Engineering Research in Computer Science and Engineering. https://doi.org/10.36647/ijercse/09.06.art003

[11] Zahid, S., Hafeez, H., Iqbal, M., Asif, A., Yaqoob, S., and Mehboob, F. (2024). Credit Card Fraud Detection using Deep Learning and Machine Learning Algorithms. Journal of Innovative Computing and Emerging Technologies. https://doi.org/10.56536/jicet.v4i1.106

[12] Kumar, J., and Goswami, P. (2024). Credit Card Fraud Detection Using Machine Learning. International Journal For Multidisciplinary Research. https://doi.org/10.36948/ijfmr.2024.v06i02.19237

[13] Kowsalya, K., Vasumathi, M. and Selvakani, D. S. S (2024). Credit Card Fraud Detection Using Machine Learning Algorithms. EPRA International Journal of Multidisciplinary Research (IJMR). 2024 https://doi.org/10.36713/epra16045

[14] Bhati, B. (2024). Credit Card Fraud Detection Using Machine Learning. Interantional Journal of Scientific Research in Engineering and Management. https://doi.org/10.55041/ijsrem29580

[15] Arora, K., Pathak, S., and Linh, N. (2023). Comparative Analysis ofK-Nn, Naïve Bayes, and logistic regression for credit card fraud detection. Ingenieria Solidaria. https://doi.org/10.16925/2357-6014.2023.03.05

[16] Aghwar, F., Ojugo, A., Adigwe, W., Odiakaose, C., Ojei, E., Ashioba, N., Okpor, M., and Geteloma, V. (2024). Enhancing the Random Forest Model via Synthetic Minority Oversampling Technique for Credit-Card Fraud Detection. Journal of Computing Theories and Applications. https://doi.org/10.62411/jcta.10323

[17] Manorom, P., Detthamrong, U., and Chansanam, W. (2024). Comparative Assessment of Fraudulent Financial Transactions using the Machine Learning Algorithms Decision Tree, Logistic Regression, Naïve Bayes, K-Nearest Neighbor, and Random Forest. Engineering, Technology & Applied Science Research. https://doi.org/10.48084/etasr.7774

[18] Airlangga, G. (2024). Evaluating the Efficacy of Machine Learning Models in Credit Card Fraud Detection. Journal of Computer Networks, Architecture and High-Performance Computing. https://doi.org/10.47709/cnahpc.v6i2.3814

[19] Tripathy, N., Balabantaray, S., Parida, S., and Nayak, S. (2024). Cryptocurrency fraud detection through classification techniques. International Journal of Electrical and Computer Engineering (IJECE). https://doi.org/10.11591/ijece.v14i3.pp2918-2926

[20] Aziz, R., Baluch, M., Patel, S., and Kumar, P. (2022). A Machine Learning based Approach to Detect the Ethereum Fraud Transactions with Limited Attributes. Karbala International Journal of Modern Science. https://doi.org/10.33640/2405-609x.3229.

[21] Awoyemi, J. O., Adetunmbi, A. O. and Oluwadare, S. A., (2017). Credit card fraud detection using machine learning techniques: A comparative analysis. In 2017 international conference on computing networking and informatics (ICCNI) (pp. 1-9). IEEE.

[22] Sailusha, R., Gnaneswar, V., Ramesh, R. and Rao, G. R. (2020). Credit card fraud detection using machine learning. In 2020 4th international conference on intelligent computing and control systems (ICICCS) (pp. 1264-1270). IEEE.

[23] Jovanovic, D., Antonijevic, M., Stankovic, M., Zivkovic, M., Tanaskovic, M. and Bacanin, N., (2022). Tuning machine learning models using a group search firefly algorithm for credit card fraud detection. Mathematics, 10(13), p.2272.

[24] Kartik. (2020). Fraud Detection. Kaggle. Retrieved from https://www.kaggle.com/datasets/kartik2112/fraud-detection