Jurnal Ilmu Komputer dan Informatika (JIKI) Vol. 5, No. 2, December 2025, Page. 115-126
P-ISSN: 2807-6664 https://jiki.jurnal-id.com
E-ISSN: 2807-6591 DOI: https://doi.org/10.54082/jiki.306

Performance Comparison of Logistic Regression and XGBoost for Credit Card Fraud Detection using Random Undersampling and Hyperparameter Tuning

Hasri Akbar Awal Rozaq*[1], Deni Sutaji[2]

[1,2]Graduate School of Informatics, Department of Computer Science, Gazi University, Türkiye

Email: [1]hakbar.rozaq@gazi.edu.tr

Submitted: 05 November 2025; Revised: 14 December 2025; Accepted: 26 December 2025;
Published: 31 December 2025

Abstract

Credit card fraud is a growing problem due to the rise of card transactions. This study investigates the effectiveness of Logistic Regression (LogReg) and Extreme Gradient Boosting (XGBoost) in identifying fraudulent transactions in a highly imbalanced dataset, where only 8% of the data represents fraudulent activity. To address the class imbalance, random undersampling was applied, reducing the number of legitimate transactions. This technique significantly improved LogReg's ability to detect fraud, with the AUC-ROC increasing from 0.7994 to 0.9089. XGBoost performed well even without hyperparameter tuning or random undersampling, indicating its robustness as a baseline model. The study highlights the critical importance of addressing class imbalance in fraud detection. Both LogReg and XGBoost demonstrated potential, particularly when combined with techniques like undersampling or hyperparameter tuning. These findings underscore the need for effective data preprocessing methods to enhance the performance of machine learning models in detecting credit card fraud.

Keywords: *Fraud Detection, Hyperparameter Tuning, Imbalanced data, Logistic Regression, Random Undersampling, XGBoost*

1. INTRODUCTION

Over the past few decades, the fintech industry has experienced rapid growth, making credit cards a common choice for everyday purchases. However, this rapid rise in credit card usage has also led to an increase in credit card fraud, creating significant financial risks to both consumers and financial institutions. Effective fraud detection has become essential to mitigate these risks and maintain the integrity of financial systems.

To address this vulnerability, it is crucial to implement effective methods that can accurately identify fraudulent transactions. Data mining and machine learning approaches have proven to be particularly effective in this context[1]. Logistic Regression is a proven method known for its effectiveness when dealing with straightforward datasets that have linear relationships[2]. However, it faces challenges when handling complex data with many dimensions, known as the "curse of dimensionality." As the number of features grows, Logistic Regression struggles to maintain accuracy, requiring more data points[3], [4].

In contrast, Extreme Gradient Boosting (XGBoost) is a powerful machine learning algorithm that enhances logistic regression's binary classification concept through gradient boosting[2]. It leverages gradient tree boosting to create a more flexible ensemble model, making it highly effective in handling complex relationships and non-linear patterns within high-dimensional data[3], [5]. Both algorithms

share the foundation of predicting binary outcomes using a logistic function, but XGBoost offers a more robust approach for intricate datasets.[2].

To maximize the performance of XGBoost, hyperparameter tuning is applied. This method optimizes the algorithm's performance by adjusting its parameters. In this study, Randomized Search CV is employed for hyperparameter tuning due to its simplicity and minimal computational resource requirements[6]. It often helps to improve model performance in many cases.

Real-world applications often involve imbalanced data, where classes are not equally represented. This imbalance can significantly impact model performance, biased to the majority class [7], [8]. With this matter, It is crucial to find the best way to address the imbalance data problem[7]. However, the optimal approach depends on the characteristics of the dataset and the specific classification task[7]. For that reason, Random undersampling could be a great approach regarding this matter[9], [10].

Random undersampling is a simple sampling technique that can be used to addresses data imbalance by randomly selecting samples, in this study, from the majority class[11]. It is suitable for this study as it effectively reduces the size of the majority class. Given the large size of the dataset, substantial computational resources are required. However, the simplicity of random undersampling makes it an efficient choice for this purpose.

In summary, this paper aims to achieve two objectives: (I) to compare the performance of Logistic Regression and XGBoost algorithms using Random Undersampling in classifying fraudulent transactions, and (II) to optimize the XGBoost accuracy Hyperparameter Tuning using Randomized Search CV.

2. METHOD

This chapter explain the methodology employed in this research to compare the performance of Logistic Regression and XGBoost for credit card fraud detection with Random Undersampling technique. The chapter will be divided into the following sections in Figure 1:

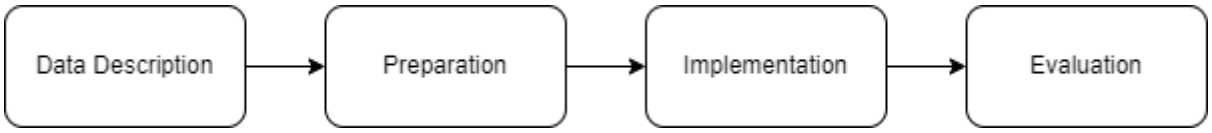

Figure 1. Research method stage

2.1. Data Description

The dataset is obtained from Kaggle[12] which is available publicly. It has 1,000,000 samples with 87,403 true values in the target class and 912,597 false values in the target class. It is implied that this dataset has an extremely imbalanced class with only 8.74% of the minority class. Besides, It also has 7 features with 3 numeric data and 4 subtype data, with a target class column at the end. The dataset is described in Table 1 and Table 2.

Table 1. Dataset Description

No	Column	Description
1	distance_from_home	Distance between the attempted transaction location and the cardholder's registered address (in Kms).
2	distance_from_last_transaction	Distance between the attempted transaction location and the location of the most recent transaction using the same card (in Kms).
3	ratio_to_median_purchase_price	Ratio of the current transaction amount to the median transaction amount for this card (e.g., transaction amount of 85 with a median of 50 would be 0.59).

4	repeat_retailer	Indicates whether the attempted transaction occurs at a retailer where the card is frequently used (Yes/No).
5	used_chip	Indicates whether the transaction attempt involved using an RFID chip in the card (Yes/No).
6	used_pin_number	Indicates whether the transaction attempt involved entering a PIN number (Yes/No).
7	online_order	Indicates whether the transaction attempt is for an online order (Yes/No).
8	fraud	Indicates whether the transaction is classified as fraudulent or legitimate (Yes/No).

Table 2. Summary of Statistics

No	Column	Type	Max	Min	Mean	Std
1	distance_from_home	Numerical	10632.723670	0.004874	26.628792	65.90784
2	distance_from_last_transaction	Numerical	11851.104560	0.000118	5.036519	25.843093
3	ratio_to_median_purchase_price	Numerical	267.802942	0.004399	1.824182	2.799589
4	repeat_retailer	Categorical (Binary)	1.000000	0.000000	0.881536	0.323157
5	used_chip	Categorical (Binary)	1.000000	0.000000	0.350399	0.477095
6	used_pin_number	Categorical (Binary)	1.000000	0.000000	0.100608	0.300809
7	online_order	Categorical (Binary)	1.000000	0.000000	0.650552	0.476796
8	fraud	Categorical (Binary)	1.000000	0.000000	0.087403	0.282425

2.2. Preparation

This study leverages a credit card fraud dataset with minimal cleaning requirements. In this stage, Exploratory data analysis (EDA) is conducted as the initial step to gain valuable insights into the data distribution and identify potential areas for preprocessing. Understanding these characteristics helps guide subsequent data preparation techniques. Following EDA, the data undergoes the following preprocessing steps:

2.2.1. Splitting

The data is then split into training and testing sets. Splitting to 80% for training and 20% for testing is one of the common approach. The training set is used to build the model, while the unseen testing set provides an unbiased evaluation of the model's generalizability. Splitting the data first before doing any preprocessing process is generally recommended to prevent data leakage to the testing set and unintentionally influence the model during training, leading to overfitting.

2.2.2. Feature Scaling

Feature scaling is applied to standardize the data. Standardization ensures all features have a mean of zero and a standard deviation of one, removing the influence of units and promoting equal contribution from each feature during model training[13] For Standardization purpose, StandardScaller from Python's Scikit-Learn library is used.

2.2.3. Random Undersampling (RUS)

To address class imbalance within the dataset, Random Undersampling (RUS) is employed. This technique strategically removes majority class samples to a desired number[11]. Samples are chosen randomly, aiming to achieve a more balanced class distribution. In this study, RUS is used to choose 200,000 of majority class in 800,000 training set. RUS requires fewer computational resources, but it does come with a potential drawback such as the possibly loss of valuable data insights. The decision to use it depends on the severity of the class imbalance and the chosen modeling approach[14], [15].

2.3. Implementation

This section focuses on the modelling stage, with the implementation of two machine learning algorithms, Logistic Regression and XGBoost. Hyperparameter Tuning with Randomized Search CV will also be implemented with XGBoost. It will be a great step for the model to optimize their performance in identifying fraudulent transactions[6].

2.3.1. Logistic Regression

A logistic regression can be interpreted as a generalized linear model (GLM) when the dependent variable is binary, either 0 or 1 [16], [17]. It has recently been used to analyze the advantages of using measurable independent factors and to see how a group of these factors affects the regression outcomes [5]. Unlike linear regression, which predicts continuous values, logistic regression transforms its linear output using the logistic function (sigmoid function) to produce a probability value between 0 and 1[18]. This probability indicates the likelihood of an observation belonging to the positive class (e.g., fraudulent transaction).

To estimate the probability of to the positive class, linear score is calculated first (z) using equation 1. This score is a weighted sum of the independent variables, where each variable's contribution is determined by its associated weight[19]. Equation 2 then employs the logistic function (σ) to transform this linear score (z) into a probability estimate ($\hat{\boldsymbol{y}}$) between 0 and 1.

$$\boldsymbol{z} = \boldsymbol{w^T} * \boldsymbol{x} + \boldsymbol{b} \tag{1}$$

$$\hat{\boldsymbol{y}} = \boldsymbol{\sigma(z)} = \frac{\mathbf{1}}{(\mathbf{1}+\boldsymbol{e}^{(-\boldsymbol{z})})} \tag{2}$$

Information:

z : Linear Predictor
$\boldsymbol{w^T}$: Transpose of the Weights (w)
x : Independent Variable
b : Bias Term
$\hat{\boldsymbol{y}}$: Predicted Probability
$\boldsymbol{\sigma(z)}$: Logistic Function
e : Exponential Function (2.71828)

This function helps determine the likelihood of fraud. Logistic regression has several advantages, including computational efficiency, making it suitable for large datasets and resource-constrained situations. It can handle various types of variables and does not require normal distribution.[20]. Furthermore, it does well in capturing linear relationships between features and the target variable, leading to accurate classifications. However, logistic regression is not without limitations. It is restricted to binary classification problems only, as it struggles with scenarios involving more than two classes. Additionally, logistic regression struggles with capturing non-linear relationships and may be sensitive to outliers, potentially leading to suboptimal results [20].

2.3.2. XGBoost

Unlike logistic regression, which relies on a single model, XGBoost leverages the power of ensemble learning, specifically a technique known as gradient boosting. Ensemble learning builds robust models by combining predictions from multiple weak learners, often referred to as base learners which is decision trees[21]. Gradient boosting takes this concept further by creating new models. Each new model focuses on correcting the errors made by the previous one, leading to a more accurate ensemble [22]

XGBoost offers several advantages that make it a really good option for different machine learning jobs. Its ability to handle complex, non-linear relationships between features and the target variable is a significant strength compared to simpler models like logistic regression[3]. Moreover, XGBoost's ability to manage sparse data well and can train multiple parts at the same time, which is great for dealing with complex datasets in real-world situations such as credit card fraud.

While XGBoost offers significant advantages, it's not without limitations. One consideration is its computational demands. Training XGBoost models can require more computational resources compared to simpler algorithms like logistic regression. Additionally, XGBoost has a larger set of hyperparameters that require careful tuning for optimal performance[6]. This tuning process can be more complex compared to LogReg, potentially requiring expertise and experimentation.

Despite these limitations, XGBoost remains a powerful tool for classification and regression tasks due to its ability to learn complex patterns, handle large datasets efficiently, and offer valuable insights into feature importance[5].

2.3.3. Hyperparameter Tuning

While machine learning models rely on hyperparameters that influence their performance, the optimal values for these parameters need to be decide. Hyperparameter tuning offers a method to identify the most optimal values to improve the model performance[23], [24].

Several techniques exist for hyperparameter tuning, with grid search and randomized search being common approaches. Grid search carefully checks a set of values one by one, making sure to cover everything broadly. Thus, this thoroughness need a lot of computing resource. [6]. Randomized search offers an alternative by efficiently sampling hyperparameter combinations, reducing the computational resources needed and risk of getting stuck in suboptimal configurations[6].

The importance of hyperparameter tuning is suitable for algorithms like XGBoost. XGBoost, with its ensemble nature and gradient boosting framework, owns a rich set of hyperparameters impacting model complexity, regularization, and learning rate[6]. Tuning these parameters allows for optimization, leading to improved accuracy[6]. In this study, Randomized Search CV will be employed for XGBoost to efficiently explore the hyperparameter space and achieve optimal model performance.

2.4. Evaluation

The implemented models will be evaluated on a held-out test set created by splitting the preprocessed data in 80/20 split ratio. Established metrics like accuracy, precision, recall, F1-score, and AUC-ROC will be used to compare different configurations performance of Logistic Regression and XGBoost, such as with and without hyperparameter tuning Randomized Search CV, as well as with and without Random undersampling. This evaluation will identify the most effective model configuration for credit card fraud detection.

AUC-ROC (Area Under the Receiver Operating Characteristic Curve) serves as a performance metric that summarizes how well a model distinguishes between positive and negative cases. It achieves this by calculating the probability that a randomly chosen positive example ranks higher on the ROC curve (higher True Positive Rate) compared to a randomly chosen negative example[21].

In binary classification tasks, accurately identifying both positive and negative cases is crucial. True positives (TP) represent correctly classified positive examples, while true negatives (TN) represent correctly classified negative examples. In the other hand, false positives (FP) occur when the model mistakenly classifies a negative example as positive, and false negatives (FN) occur when the model misses a positive example, identifying it as negative. It is included in the confusion matrix table is shown in the table 3.

Table 3. Confusion Matrix

Actual Class	Predicted Class	
	1	0
1	TP	FP
0	FN	TN

Accuracy is the ratio of correctly predicted instances to the total instances and is formulated in Equation 3:

$$accuracy = \frac{TP+TN}{TP+TN+FP+FN} \times 100\% \quad (3)$$

Precision measures the proportion of true positive predictions out of all positive predictions made and is formulated in Equation 4:

$$precision = \frac{TP}{TP+FP} \times 100\% \quad (4)$$

Recall quantifies the proportion of actual positive instances that were correctly identified and is formulated in Equation 5:

$$recall = \frac{TP}{TP+FN} \times 100\% \quad (5)$$

F1-score is the harmonic mean of precision and recall, balancing the trade-off between these metrics to provide an overall assessment of a model's performance in identifying positive cases while minimizing false positives. It is formulated in Equation 6:

$$F1 - score = \frac{2 \times precision \times recall}{precision + recall} \quad (6)$$

Information:
TP : True Positive
TN : True Negative
FP : False Positive
FN : False Negative

3. RESULT

This section is a discussion of the study that has been done. Starting from the preprocessing, implementation, and evaluation.

3.1. Preprocessing result

In this stage, several data preprocessing techniques is performed. To avoid contaminating the testing set, the data is first split into training and testing sets with a ratio of 80:20. This ensures that the preprocessing steps are applied only to the training data. The details of the split are shown in Table 4.

Table 4. Data splitting

Training	Testing
80%	20%
800,000	200,000
1,000,000	

Next, feature scaling is applied. Scaling the data before splitting it into training and testing sets is crucial. This prevents the data leakage and statistics used for scaling (e.g., standard deviation and mean) from being influenced by the testing data. Standardscaller is used in this stage for standardization scaling. Standardization transforms the features to have a mean of 0 and a standard deviation of 1[25].

3.2. Sampling result

After scaling, a class imbalance handling technique is surely required for the data, such as a sampling method[11]. In this stage, The train dataset will reduce it rows using random undersampling. It only reduce the majority class rows to 200,000 while minority rows still remain the same. It will boost the percentage of the minority class from 9% to 30%. The result and the before is shown in the figure 2 and 3.

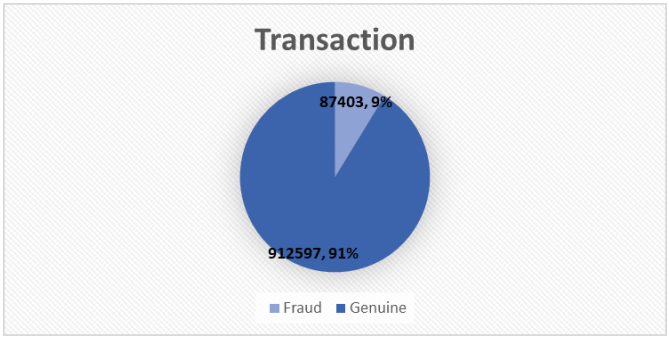

Figure 2. Class percentage before sampling

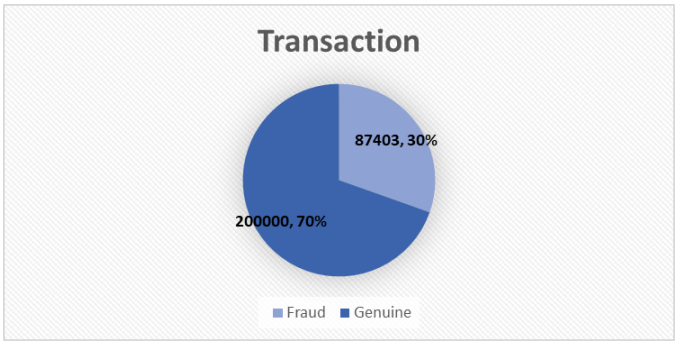

Figure 3. Class percentage after sampling

Based on Figures 3 and 4, it can be observed that the ratio between the minority and majority class has significantly improved after applying Random Undersampling (RUS). Figure 3 likely represents the class distribution before RUS, where the majority class dominates the pie chart. In contrast, Figure 4 presumably shows the distribution after RUS, where the minority class now occupies a noticeably larger portion relative to the majority class. This visual representation suggests that RUS effectively reduced the majority class size, leading to a more balanced class distribution.

3.3. Performance Comparison

After preprocessing the data, configuring the models is crucial for achieving optimal performance. This research investigated several configurations: Logistic Regression with and without Random Undersampling (RUS), and XGBoost with and without RUS, additionally exploring XGBoost with hyperparameter tuning. Due to the large dataset and limited computational resources, Randomized Search CV was employed as a hyperparameter tuning technique.

Following the exploration of various model configurations, this section delves deeper into the performance comparison of these models. The results for each configuration will be analyzed using metrics like accuracy, precision, recall, F1-score, and AUC-ROC. By comparing the performance metrics across configurations, this study aim to identify the model that achieves the most robust and accurate credit card fraud detection

3.3.1. Logistic Regression

Logistic Regression performed with 2 model configurations, with or without Random Undersampling. The classification report is shown in the table 5 and 6.

Table 5. Logistic Regression Report without RUS

	Precision	Recall	F1-Score	Support
0.0	0.9634	0.9933	0.9781	182519
1.0	0.8964	0.6055	0.7228	17481
accuracy			0.9594	200000
macro avg	0.9299	0.7994	0.8504	200000
weighted avg	0.9575	0.9594	0.9558	200000
AUC-ROC	0.7994042037616316			

Table 6. Logistic Regression Report with RUS

	Precision	Recall	F1-Score	Support
0.0	0.9852	0.9703	0.9777	182519
1.0	0.7322	0.8476	0.7857	17481
accuracy			0.9596	200000
macro avg	0.8587	0.9090	0.8817	200000
weighted avg	0.9631	0.9596	0.9609	200000
AUC-ROC	0.908957956266683			

As shown in the figure 5 and 6, It is implied that Logistic Regression exhibits a substantial improvement in AUC-ROC (from 0.7994 to 0.9089) after applying Random Undersampling (RUS), while maintaining a similar overall accuracy (around 0.96). This suggests that the initial model, despite achieving high accuracy, suffers from a class imbalance issue. The dominant class likely biases the model's predictions, leading to a lower AUC-ROC, which reflects the model's ability to differentiate between the positive and negative classes. RUS effectively addresses this imbalance by reducing the majority class size, resulting in a more balanced distribution and a more robust performance measure as evidenced by the significant increase in AUC-ROC.

3.3.2. XGBoost

XGBoost performed with 3 model configurations, the baseline model, with Hyperparameter Tuning using Randomized Search CV, and with Random Undersampling . The result is shown in the table 7 to 9.

Table 7. XGBoost Baseline Model Report

	Precision	Recall	F1-Score	Support
0.0	0.9991	0.9989	0.9990	182519
1.0	0.9886	0.9905	0.9896	17481
accuracy			0.9982	200000
macro avg	0.9939	0.9947	0.9943	200000
weighted avg	0.9982	0.9982	0.9982	200000
AUC-ROC	0.9947068391483049			

Table 8. XGBoost Report with Hyperparameter Tuning

	Precision	Recall	F1-Score	Support
0.0	0.9991	0.9992	0.9991	182519
1.0	0.9915	0.9903	0.9909	17481
accuracy			0.9984	200000
macro avg	0.9953	0.9948	0.9950	200000
weighted avg	0.9984	0.9984	0.9984	200000
AUC-ROC	0.9947607431823108			

Table 9. XGBoost Report with RUS

	Precision	Recall	F1-Score	Support
0.0	0.9996	0.9991	0.9994	182519
1.0	0.9904	0.9963	0.9934	17481
accuracy			0.9988	200000
macro avg	0.9950	0.9977	0.9964	200000
weighted avg	0.9988	0.9988	0.9988	200000
AUC-ROC	0.9976806125661357			

Based on the table 7 to 9, It can be implied that XGBoost performs well on this classification task, achieving high accuracy (around 99.8%) across all configurations. However, hyperparameter tuning and Random Undersampling (RUS) appear to have minimal impact on overall accuracy. The key difference lies in the model's ability to distinguish between classes, measured by AUC-ROC. While the baseline XGBoost achieves a high AUC-ROC (0.9947), both tuning (0.9948) and RUS (0.9977) lead to slight improvements. Notably, RUS has the most significant impact, suggesting that the original data have a class imbalance issue. RUS helps address this imbalance, resulting in a model that can better differentiate between the positive and negative classes, as reflected in the higher AUC-ROC score.

4. DISCUSSIONS

This analysis resulting critical insights into model performance, class imbalance effects, tuning optimization effect, and the strengths of both algorithms. Both XGBoost and Logistic Regression achieved high overall accuracy, with XGBoost at around 99.8% and Logistic Regression at 95.9% across all configurations. However, a closer look reveals a significant difference in their ability to differentiate between positive and negative classes, as indicated by the Area Under the ROC Curve (AUC-ROC).

While Logistic Regression achieved high accuracy, its initial AUC-ROC score (0.7994) hinting at challenges with class imbalance. This is further supported by the analysis of precision and recall. While Logistic Regression demonstrated high precision (0.9634) for the majority class, suggesting it could predict the dominant class well, it struggled with recall (0.6055) for the minority class (class 1), leading to a notable number of false negatives. This highlights Logistic Regression's limitations in handling imbalanced datasets.

Applying Random Undersampling (RUS) to Logistic Regression significantly improves its AUC-ROC score (0.9089). This suggests that RUS effectively addresses class imbalance by reducing the majority class size, leading to a more balanced distribution and consequently, a more robust performance measure. The improvement in recall for the minority class (0.8476) further supports this notion, indicating that RUS helps Logistic Regression identify more true positives in the minority class.

In contrast, XGBoost consistently displayed a strong ability to distinguish between classes, evident from its high AUC-ROC scores (around 0.995) across all setups. This indicates XGBoost's inherent capability in handling class imbalance, possibly due to its ensemble learning approach and capacity to capture intricate data relationships. Even without hyperparameter tuning or class balancing techniques, XGBoost maintained a high AUC-ROC, showcasing its robustness in imbalanced classification tasks. The application of Hyperparameter Tuning and Random Undersampling did not significantly impact its performance, as the model was already performing well.

5. CONCLUSION

This study investigated the effectiveness of Logistic Regression (LogReg) and Extreme Gradient Boosting (XGBoost) for credit card fraud detection. These findings highlight how important it is to solve class imbalance problem, which is a big challenge in this field because there aren't many fraudulent transactions in real-world case. Random Undersampling, a technique that reduces the majority class size, was employed to address this imbalance dataset and successfully improve model performance. It was observed that LogReg required modifications, such as the use of Random Undersampling, to achieve optimal performance, while XGBoost demonstrated decent performance even in its baseline state. The evaluation, using metrics like accuracy, precision, recall, F1-score, and AUC-ROC, revealed that XGBoost consistently outperformed LogReg in identifying fraudulent transactions, especially when hyperparameter tuning with Randomized Search was applied. This suggests that XGBoost's inherent ability to handle complex relationships within data makes it a more robust choice for credit card fraud detection tasks characterized by class imbalance**.**

CONFLICT OF INTEREST

The authors declares that there is no conflict of interest between the authors or with research object in this paper.

REFERENCES

[1] A. RB and S. K. KR, “Credit card fraud detection using artificial neural network,” *Global Transitions Proceedings*, vol. 2, no. 1, pp. 35–41, Jun. 2021, doi: 10.1016/j.gltp.2021.01.006.

[2] J. Pesantez-Narvaez, M. Guillen, and M. Alcañiz, “Predicting motor insurance claims using telematics data—XGboost versus logistic regression,” *Risks*, vol. 7, no. 2, Jun. 2019, doi: 10.3390/risks7020070.

[3] P. Liu, X. J. Li, T. Zhang, and Y. H. Huang, “Comparison between XGboost model and logistic regression model for predicting sepsis after extremely severe burns,” *Journal of International Medical Research*, vol. 52, no. 5, May 2024, doi: 10.1177/03000605241247696.

[4] V. Berisha *et al.*, “Digital medicine and the curse of dimensionality,” *npj Digital Medicine*, vol. 4, no. 1. Nature Research, Dec. 01, 2021. doi: 10.1038/s41746-021-00521-5.

[5] Y. Xu *et al.*, “Predicting ICU Mortality in Rheumatic Heart Disease: Comparison of XGBoost and Logistic Regression,” *Front Cardiovasc Med*, vol. 9, Feb. 2022, doi: 10.3389/fcvm.2022.847206.

[6] E. Elgeldawi, A. Sayed, A. R. Galal, and A. M. Zaki, "Hyperparameter tuning for machine learning algorithms used for arabic sentiment analysis," *Informatics*, vol. 8, no. 4, Dec. 2021, doi: 10.3390/informatics8040079.

[7] N. H. N. B. M. Shahri, S. B. S. Lai, M. B. Mohamad, H. A. B. A. Rahman, and A. Bin Rambli, "Comparing the performance of adaboost, xgboost, and logistic regression for imbalanced data," *Mathematics and Statistics*, vol. 9, no. 3, pp. 379–385, 2021, doi: 10.13189/ms.2021.090320.

[8] S. Wang, Y. Dai, J. Shen, and J. Xuan, "Research on expansion and classification of imbalanced data based on SMOTE algorithm," *Sci Rep*, vol. 11, no. 1, Dec. 2021, doi: 10.1038/s41598-021-03430-5.

[9] E. F. Swana, W. Doorsamy, and P. Bokoro, "Tomek Link and SMOTE Approaches for Machine Fault Classification with an Imbalanced Dataset," *Sensors*, vol. 22, no. 9, May 2022, doi: 10.3390/s22093246.

[10] North Eastern Hill University. Department of Biomedical Engineering, Institute of Electrical and Electronics Engineers. Kolkata Section, IEEE Industry Applications Society, and Institute of Electrical and Electronics Engineers, *International Conference on Computational Performance Evaluation : ComPE 2020 online conference : 2nd-4th July 2020*. doi: https://doi.org/10.1109/ComPE49325.2020.9200087.

[11] R. Mohammed, J. Rawashdeh, and M. Abdullah, "Machine Learning with Oversampling and Undersampling Techniques: Overview Study and Experimental Results," in *2020 11th International Conference on Information and Communication Systems, ICICS 2020*, Institute of Electrical and Electronics Engineers Inc., Apr. 2020, pp. 243–248. doi: 10.1109/ICICS49469.2020.239556.

[12] DHANUSH NARAYANAN R, "Credit Card Fraud Dataset," kaggle.com. Accessed: Jun. 01, 2024. [Online]. Available: https://www.kaggle.com/datasets/dhanushnarayananr/credit-card-fraud

[13] H. Peng and J. Wang, "Unbalanced Data Processing and Machine Learning in Credit Card Fraud Detection," 2022, doi: 10.21203/rs.3.rs-2004320/v1.

[14] N. Nnamoko and I. Korkontzelos, "Efficient treatment of outliers and class imbalance for diabetes prediction," *Artif Intell Med*, vol. 104, Apr. 2020, doi: 10.1016/j.artmed.2020.101815.

[15] B. Liu and G. Tsoumakas, "Dealing with class imbalance in classifier chains via random undersampling," *Knowl Based Syst*, vol. 192, Mar. 2020, doi: 10.1016/j.knosys.2019.105292.

[16] Sri Sairam Engineering College. Department of Information Technology and Institute of Electrical and Electronics Engineers, *2019 proceedings of the 3rd International Conference on Computing and Communications Technologies (ICCCT'19) : February 21-22, 2019, Chennai, India.* doi: https://doi.org/10.1109/ICCCT2.2019.8824930.

[17] A. A. T. Fernandes, D. B. F. Filho, E. C. da Rocha, and W. da Silva Nascimento, "Read this paper if you want to learn logistic regression," *Revista de Sociologia e Politica*, vol. 28, no. 74, pp. 1/1-19/19, 2020, doi: 10.1590/1678-987320287406EN.

[18] E. O. Bayman and F. Dexter, "Multicollinearity in Logistic Regression Models," *Anesth Analg*, vol. 133, no. 2, pp. 362–365, 2021, doi: 10.1213/ANE.0000000000005593.

[19] "Mathematical justification on the origin of the sigmoid in logistic regression," *Central European Management Journal*, 2022, doi: 10.57030/23364890.cemj.30.4.135.

[20] V. H. Nhu *et al.*, "Shallow landslide susceptibility mapping: A comparison between logistic model tree, logistic regression, naïve bayes tree, artificial neural network, and support vector

machine algorithms," *Int J Environ Res Public Health*, vol. 17, no. 8, Apr. 2020, doi: 10.3390/ijerph17082749.

[21] Y. Zhang, J. Tong, Z. Wang, and F. Gao, "Customer Transaction Fraud Detection Using Xgboost Model," in *Proceedings - 2020 International Conference on Computer Engineering and Application, ICCEA 2020*, Institute of Electrical and Electronics Engineers Inc., Mar. 2020, pp. 554–558. doi: 10.1109/ICCEA50009.2020.00122.

[22] H. Jain, A. Khunteta, and S. Srivastava, "Churn Prediction in Telecommunication using Logistic Regression and Logit Boost," in *Procedia Computer Science*, Elsevier B.V., 2020, pp. 101–112. doi: 10.1016/j.procs.2020.03.187.

[23] R. Turner *et al.*, "Bayesian Optimization is Superior to Random Search for Machine Learning Hyperparameter Tuning: Analysis of the Black-Box Optimization Challenge 2020." doi: https://doi.org/10.48550/arXiv.2104.10201.

[24] H. J. P. Weerts, A. C. Mueller, and J. Vanschoren, "Importance of Tuning Hyperparameters of Machine Learning Algorithms," Jul. 2020, doi: https://doi.org/10.48550/arXiv.2007.07588.

[25] H. Benhar, A. Idri, and J. L Fernández-Alemán, "Data preprocessing for heart disease classification: A systematic literature review.," *Computer Methods and Programs in Biomedicine*, vol. 195. Elsevier Ireland Ltd, Oct. 01, 2020. doi: 10.1016/j.cmpb.2020.105635