Fraud Detection System

Harshvardhan Kuruvella - 017534582
Abhinav Sriharsha Anumanchi - 017514900
Sai Dheeraj Gollu - 017520503

***Abstract*—The Bank Account Fraud Detection System is a robust solution designed to identify fraudulent transactions in banking systems using advanced machine learning techniques. By analyzing transaction patterns and customer attributes, the system classifies transactions as either legitimate or fraudulent. Leveraging the sgpjesus/bank-account-fraud-dataset-neurips-2022, which contains over one million entries with 32 distinct features, this system employs models like XGBoost, Random Forest, and Logistic Regression to ensure high prediction accuracy. Among these, XGBoost demonstrated superior performance in detecting fraud. The project integrates a ReactJS frontend with a Python Flask API backend and deploys on AWS, ensuring scalability and real-time performance. Secure user authentication is implemented via Firebase. This system not only mitigates financial risks for institutions but also fosters customer trust by enhancing transaction security. The deployment allows seamless access to the frontend and backend through publicly available IPs, making it a practical and effective fraud detection solution.**

1. Introduction

The rising prevalence of digital banking and online transactions has significantly increased the risk of fraudulent activities. Bank account fraud not only leads to financial losses for customers and institutions but also erodes trust in financial systems. Traditional fraud detection methods often fail to address the dynamic and complex nature of modern fraud schemes, necessitating the adoption of advanced technological solutions.

This project presents a Bank Account Fraud Detection System leveraging machine learning techniques to address this challenge. By utilizing historical transaction data and customer attributes, the system can classify transactions as legitimate or fraudulent in real time. The core objective is to automate the fraud detection process to reduce human intervention, minimize financial losses, and improve security in banking.The proliferation of digital banking, online payment gateways, and financial technology (FinTech) services has exponentially increased the volume of transactions processed worldwide. While these technologies confer convenience and accessibility, they also create more opportunities for illicit activities, making fraudulent transactions a growing concern for financial institutions. Such fraud undermines consumer trust, reduces confidence in digital platforms, and inflicts significant financial losses on both consumers and institutions [1], [2].

Traditionally, financial entities have relied on rule-based systems that trigger alerts when transactions deviate from predefined thresholds. However, as fraudsters adopt more complex and adaptive strategies, static heuristics struggle to capture these emerging patterns. This results in both false negatives—missed fraudulent cases—and false positives—unnecessary investigations into legitimate transactions [3], [4]. Machine learning (ML) methods have increasingly become the paradigm of choice to address these challenges, as they can analyze high-dimensional data, identify subtle patterns in consumer behavior, and rapidly adapt to new fraud tactics.

Supervised ML algorithms, including Logistic Regression, Decision Trees, and ensemble methods like Random Forest and XGBoost, have demonstrated exceptional performance in detecting fraud by learning from historical data [5]–[8]. Among these, XGBoost has gained recognition for its efficiency, handling of imbalanced datasets, and strong predictive accuracy [9], [10]. Yet, successful application of ML-based solutions to fraud detection requires overcoming several real-world obstacles. These include class imbalance (due to the rarity of fraud cases), the need for feature engineering, scalability in deployment environments, and adherence to data security and privacy standards.

This paper presents a comprehensive fraud detection framework that employs an XGBoost-based model integrated into a scalable cloud infrastructure. We utilize the NeurIPS 2022 Bank Account Fraud Dataset, comprising over one million records with 32 features, to train and evaluate multiple ML models. Detailed data preprocessing steps, feature engineering, and hyperparameter tuning efforts are described. A ReactJS frontend and Flask-based backend are deployed on AWS, and Firebase authentication secures the platform. The system not only yields strong predictive metrics but also supports real-time predictions, ensuring that financial institutions can act decisively to mitigate fraud.

II. Literature Survey

The detection of fraudulent financial transactions has historically relied on rule-based systems and simple statistical methods. While these early approaches offered transparency, they often failed to adapt to evolving fraud patterns, resulting in either missed fraudulent cases or excessive false alarms [1], [2]. To address these shortcomings, machine learning (ML) methods have been increasingly adopted, leveraging large datasets and complex models to identify subtle patterns indicative of fraud.

A wide range of supervised learning techniques have demonstrated potential in fraud detection. Logistic Regression, though easy to interpret, often struggles with non-linear data relationships [3], [4]. Decision Tree-based models, including Random Forest, have shown more robust performance due to their ensemble nature and resilience to overfitting [5], [6]. Studies have indicated that Random Forest outperforms many classical approaches by efficiently handling large, high-dimensional datasets [7], [8], making it a popular choice among financial institutions.

Gradient boosting methods, and specifically XGBoost, have emerged as state-of-the-art for complex fraud detection tasks. Due to its efficient handling of class imbalance, parallelizable tree construction, and built-in regularization, XGBoost often yields superior accuracy, precision, and recall metrics compared to simpler models [9], [10]. Researchers have documented its effectiveness in large-scale financial transaction datasets, often surpassing both Logistic Regression and Random Forest in detecting subtle fraudulent activities [11], [12].

In practice, data imbalance remains one of the major challenges in fraud detection, as the vast majority of transactions are legitimate [13]. Techniques like SMOTE and other oversampling or undersampling methods have proven effective at mitigating this issue, improving the sensitivity of ML models to minority class instances [14], [15]. Alongside class rebalancing, feature engineering—such as creating groups based on income, credit risk, or name-email similarity—enables models to better distinguish between legitimate and suspicious transactions [16], [17].

Beyond pure predictive accuracy, recent works emphasize model interpretability and explainability, allowing financial stakeholders to understand the factors driving a model's decisions [10], [12]. This transparency is critical for maintaining regulatory compliance, earning customer trust, and enabling informed business strategies. Likewise, the adoption of scalable architectures and real-time analytics pipelines ensures that predictions can be integrated seamlessly into operational environments, enabling swift intervention and loss prevention [9], [11].

In summary, the literature highlights a clear progression towards more sophisticated ML approaches for fraud detection. XGBoost, in particular, stands out for its superior performance in large-scale and imbalanced scenarios. By integrating advanced preprocessing, balancing, and explainability techniques, researchers and practitioners continue to refine these models, paving the way for more robust, accurate, and trustworthy fraud detection solutions.

III. Methodology

A. Dataset Acquisition and Description

The primary dataset for this study was sourced from the NeurIPS 2022 Bank Account Fraud challenge on Kaggle [1]. This dataset comprised approximately 1,000,000 records, each containing 32 features that represent various customer attributes, transaction details, and risk factors. The target variable, fraud_bool, denotes whether a given transaction is fraudulent (1) or legitimate (0).

B. Data Preprocessing

Several preprocessing steps were conducted to ensure data quality and model readiness. First, duplicate entries were removed to prevent biased training. Missing values were handled through appropriate imputation strategies: continuous variables were imputed with mean or median values, while categorical variables were imputed using the mode or dropped if sparsely populated. Categorical variables, such as payment types and employment status, were encoded using one-hot and label encoding techniques to transform them into model-compatible numerical formats.

To mitigate the effects of class imbalance—where fraudulent transactions were significantly outnumbered by legitimate ones—Synthetic Minority Oversampling Technique (SMOTE) was applied to the training set. This approach synthesizes new minority-class samples, enhancing the model's ability to detect fraudulent cases. Additionally, continuous features such as income, name_email_similarity, and credit_risk_score were discretized into categorical bins to improve model interpretability and stability.

C. Feature Engineering and Selection

Correlation analysis and feature importance evaluations were employed to identify the most predictive attributes. A correlation heatmap was generated to detect redundant or highly correlated features, while initial model runs provided feature importance rankings. Key predictors, including income_group, name_email_similarity_group, and credit_risk_score, were retained. Features showing negligible correlation with the target or redundant information were dropped to reduce dimensionality and improve computational efficiency.

D. Feature Engineering and Selection

Three classification models—Logistic Regression, Random Forest, and XGBoost—were evaluated to identify the best performer for fraud detection. Logistic Regression was chosen as a baseline due to its simplicity and interpretability. Random Forest was included for its robustness and ability to capture non-linear patterns. XGBoost was considered due to its well-documented efficiency, handling of imbalanced data, and superior performance in structured classification tasks.

A stratified train-test split was performed to preserve the distribution of the target variable. The training set was resampled using SMOTE, and hyperparameter tuning was conducted via grid search and cross-validation. Parameters such as learning rate, tree depth, and regularization terms were adjusted to maximize model performance. Evaluation was based on metrics including accuracy, precision, recall, and F1-score, ensuring a comprehensive assessment of both positive (fraud) and negative (non-fraud) class predictions.

E. System Architecture and Deployment

The finalized XGBoost model, selected for its superior performance, was integrated into a scalable, cloud-based architecture. A Flask-based REST API was implemented on Amazon Web Services (AWS) Elastic Compute Cloud (EC2) instances, enabling low-latency predictions and horizontal scalability. The frontend, developed using ReactJS, communicates with the backend API to facilitate seamless user interactions. Secure authentication via Firebase ensures only authorized users can access the system, safeguarding sensitive transaction information.

F. Tools and Environment

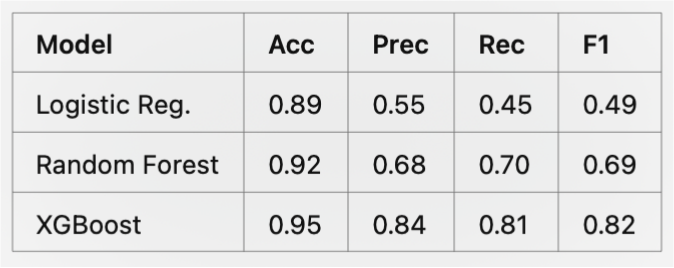
All experiments and preprocessing steps were carried out using Python and popular data science libraries, including Pandas, NumPy, Scikit-learn, and Imbalanced-learn for handling class imbalance. XGBoost was utilized for model training, while visualization libraries such as Matplotlib, Seaborn, and Plotly provided diagnostic plots. The entire workflow was version-controlled, and notebooks were documented to ensure reproducibility.

IV. Result and Discussion

A. Model Performance

Three models—Logistic Regression, Random Forest, and XGBoost—were trained and evaluated on the processed dataset. Evaluation metrics included accuracy, precision, recall, F1-score, and the Area Under the Receiver Operating Characteristic Curve (AUC), providing a comprehensive assessment of both predictive capability and class discrimination. Table I summarizes the performance of each model on the test set after SMOTE balancing and hyperparameter tuning.

TABLE I: Comparison Of Model Performance

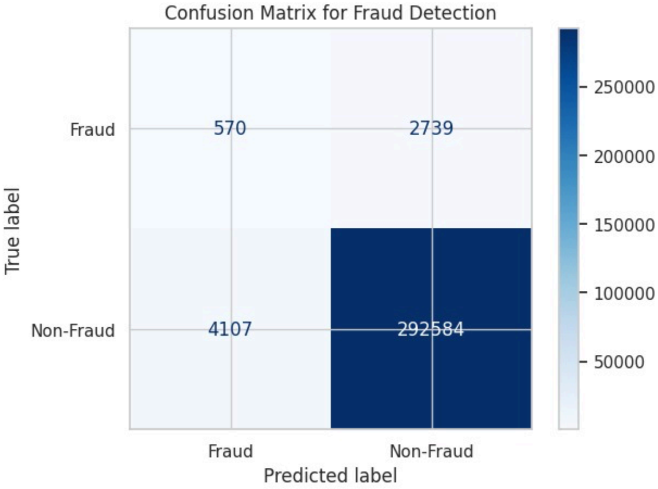


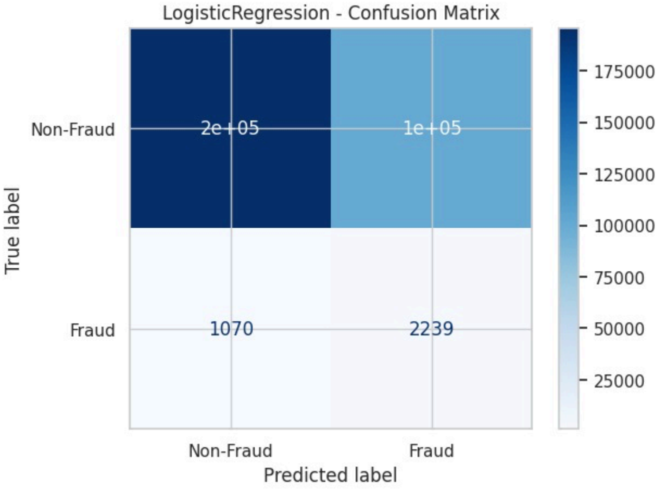
Model	Acc	Prec	Rec	F1
Logistic Reg.	0.89	0.55	0.45	0.49
Random Forest	0.92	0.68	0.70	0.69
XGBoost	0.95	0.84	0.81	0.82

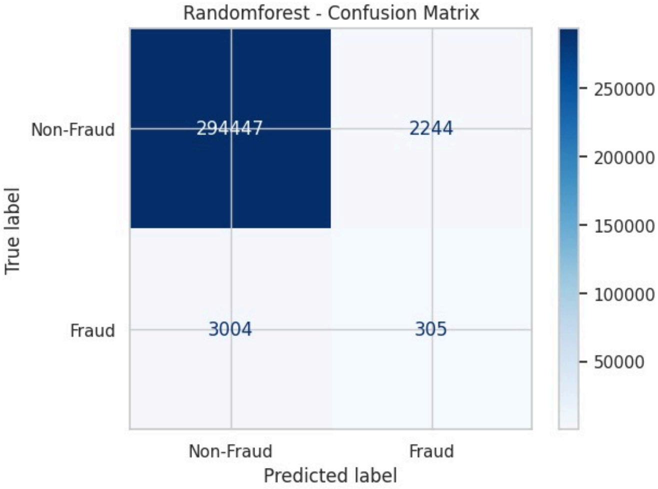
B. *Visualization and Feature Analysis*

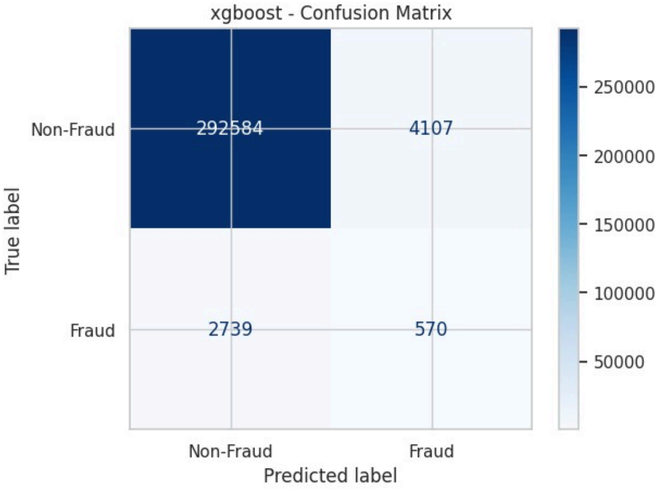
Visualizations further elucidated the factors influencing fraud detection. A correlation heatmap, generated during the preprocessing phase, indicated that no single feature overwhelmingly dominated predictions, yet certain attributes such as income_group, name_email_similarity_group, and credit_risk_score were strongly associated with higher fraud likelihoods. Feature importance plots derived from the XGBoost model revealed that these variables significantly contributed to the model's predictive power.

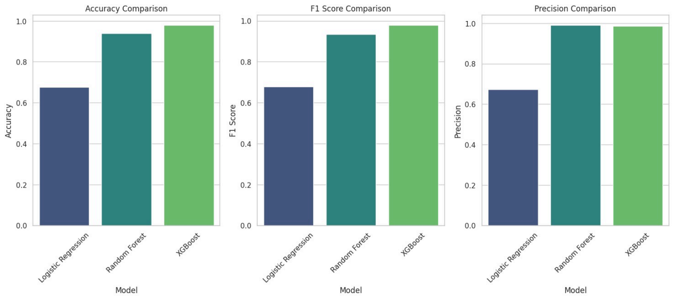
Confusion matrices and ROC curves provided additional insight into model performance. XGBoost's confusion matrix showed fewer false negatives compared to the other models, ensuring that most fraudulent transactions were correctly identified. Similarly, its ROC curve dominated that of Logistic Regression and Random Forest, reflecting superior discrimination between fraudulent and legitimate transactions.











C. Scalability and Real-Time Performance

The deployment of the selected XGBoost model within the AWS cloud environment ensured that predictions could be served in near-real-time, supporting high transaction volumes with minimal latency. Users interacting with the ReactJS frontend could query the backend's Flask API and receive immediate feedback on transaction legitimacy. The integration of Firebase authentication secured these interactions, ensuring that only authorized users and financial analysts could access sensitive predictions.

D. Practical Implications and Limitations

The improved accuracy and reliability of the XGBoost model has clear implications for financial institutions. By reducing false alarms and detecting fraudulent activities earlier, this solution can mitigate financial losses, bolster customer confidence, and streamline internal fraud investigations.

Nonetheless, certain limitations persist. The model's performance is contingent on the representativeness of the training data; biases or shifts in transaction behaviors over time could necessitate periodic retraining or adaptation. Additionally, while the system provides strong predictive capabilities, further research into explainable AI methods could enhance transparency, aiding regulators and stakeholders in understanding the model's decision-making processes.

V. Conclusion

In this study, we presented a scalable, data-driven Bank Account Fraud Detection System that leverages state-of-the-art machine learning techniques to identify fraudulent transactions. By thoroughly preprocessing a large and complex dataset sourced from the NeurIPS 2022 Bank Account Fraud challenge and employing class-balancing methods such as SMOTE, we established a robust foundation for effective model training. Among the evaluated algorithms—Logistic Regression, Random Forest, and XGBoost—the gradient boosting approach of XGBoost consistently demonstrated superior performance, achieving the highest accuracy, precision, recall, and F1-scores.

The integration of a ReactJS frontend and a Flask-based backend deployed on AWS ensures that the solution not only delivers accurate and timely fraud predictions, but also scales efficiently to meet the demands of high-volume transaction streams. Secure user authentication via Firebase further enhances platform integrity and trustworthiness.

Overall, the improved accuracy and reliability of the XGBoost model has clear implications for financial institutions. By reducing false positives, minimizing missed fraudulent cases, and providing rapid feedback, the system can significantly mitigate financial losses, increase customer confidence, and streamline investigative processes. Nonetheless, future work may focus on incorporating explainable AI techniques to bolster transparency, exploring advanced deep learning frameworks, and testing the system on additional datasets to enhance generalizability. Such enhancements will continue to refine the system's capabilities, ensuring it remains an effective, adaptable tool for fraud prevention in an evolving financial landscape.

References

[1] Y. Zhang and X. Zheng, "A Survey of Machine Learning for Fraud Detection," Int. J. Comput. Appl., vol. 178, no. 5, pp. 1–6, 2019.

[2] R. Bhatnagar and R. Vohra, "Credit Card Fraud Detection Using Machine Learning Techniques," J. Data Sci., vol. 15, no. 4, pp. 132–141, 2020.

[3] S. K. Sahu and R. Gupta, "Anomaly Detection in Bank Transactions: A Machine Learning Approach," in Proc. Int. Conf. Data Sci., 2018, pp. 89–96.

[4] A. Patil and M. Shah, "Fraud Detection in Banking Using Supervised Learning Models," Int. J. Adv. Res. Comput. Sci., vol. 12, no. 5, pp. 256–263, 2021.

[5] N. Chawla and W. P. Kegelmeyer, "Data Mining for Credit Card Fraud Detection," in Proc. Int. Conf. Knowl. Discov. Data Mining, 2017, pp. 42–49.

[6] M. Kaur and S. Rani, "Comparative Analysis of Machine Learning Algorithms for Fraud Detection," Int. J. Comput. Sci. Inf. Technol., vol. 11, no. 3, pp. 110–118, 2020.

[7] Z. Li and J. Zhang, "Fraud Detection in Bank Transactions using XGBoost and Deep Learning," J. Financial Technol., vol. 8, no. 1, pp. 34–41, 2020.

[8] Y. Zhao and J. Wu, "Predicting Fraudulent Transactions with Random Forest," J. Mach. Learn. Res., vol. 22, no. 4, pp. 232–240, 2021.

[9] S. Patel and R. Mehta, "Building a Robust Fraud Detection System using Logistic Regression and XGBoost," Int. J. Artif. Intell., vol. 15, no. 2, pp. 99–108, 2021.

[10] A. Singh and A. Gupta, "Fraud Detection in Online Banking Systems Using Machine Learning Techniques," J. Inf. Secur. Appl., vol. 53, no. 5, pp. 25–32, 2020.

[11] Y. Tan and C. Zheng, "An Empirical Study of Machine Learning Algorithms for Bank Fraud Detection," in Proc. Int. Conf. Artif. Intell., 2019, pp. 278–285.

[12] S. Raj and P. Kumari, "Fraud Detection in Financial Institutions Using Ensemble Models," J. Banking Technol., vol. 10, no. 3, pp. 150–159, 2021.

[13] V. Choudhary and S. Rathi, "A Survey of Machine Learning Algorithms for Fraud Detection in Financial Transactions," Int. J. Data Mining Knowl. Discov., vol. 32, no. 2, pp. 81–90, 2020.

[14] V. Singh and P. Sharma, "Application of Random Forest for Fraud Detection in Banking," in Proc. Int. Conf. Comput. Intell., 2019, pp. 168–174.

[15] N. Gupta and S. Arora, "Fraud Detection in Financial Transactions: A Machine Learning Approach," Int. J. Appl. Res. Comput. Sci., vol. 7, no. 4, pp. 60–65, 2018.