# Financial Transaction Fraud Detection and Risk Intelligence Platform

C. VidyaRaj[1], Deepashree M[2], Pranav Jagdish[3], Nishant[4], Pulkit Garg[5], H Nandi Prasad[6]
[1,2,3,4,5,6]*Department of Computer Science and Engineering, The National Institute of Engineering, Mysuru India*

***Abstract*—Due to the increasing volume of transactions processed through digital payment systems, detecting financial fraud is now more critical than ever before. While older rule-based systems were able to identify fraudulent activities at one point, they can no longer keep pace with new forms of fraud. As such, many organizations lose large amounts of money each year due to fraud and customers become dissatisfied. In this article we describe FraudGuard-AI, an integrated financial fraud detection and risk intelligence platform. FraudGuard-AI utilizes machine learning, explainable artificial intelligence (AI), real-time streaming capabilities and adaptive multi-factor authentication. To classify fraudulent transactions, FraudGuard-AI employs XGBoost-based gradient boosting. To understand how well the model performs, TreeSHAP is used. Transactions are handled rapidly by utilizing Apache Kafka. FraudGuard-AI was evaluated using two different types of datasets: the IEEE-CIS dataset for e-commerce and the PaySim dataset for mobile banking. Evaluation showed that FraudGuard-AI achieved a ROC-AUC score of 0.974 when applied to the IEEE-CIS dataset and was capable of processing greater than 12,000 transactions per second when evaluating the PaySim dataset. A sliding window methodology is employed to monitor transactional behavior and account anomalies.**

***Index Terms*—Financial Fraud Detection, XGBoost, Tree-SHAP, Apache Kafka, Real-Time Streaming, Risk Scoring, Adap-tive Authentication.**

## I. INTRODUCTION

Digital payments have revolutionized the way we handle money globally. Digital payment systems, mobile banking and e-commerce websites provide many benefits; however, they also present opportunities for individuals to commit fraud. According to experts, the amount of money lost due to credit card fraud will increase to $49.32 Billion U.S.dollar by 2030. Therefore, developing effective methods to detect fraud is essential for banks, payment providers and regulators. Traditional approaches to detecting fraud rely upon predetermined rules to identify fraudulent behavior.

Traditional approaches have generated false-positives that lead to customer dissatisfaction and higher cost. As fraudsters' techniques become increasingly sophisticated through methods such as identity theft and Artificial Intelligence (AI), traditional approaches will no longer suffice. An alternative method for detecting fraud includes using Machine Learning (ML) and Deep Learning (DL). Models created via ML/DL can be trained on transactional data and then retrained as fraudsters develop new methods. Examples of successful ML/DL models include XGBoost, LightGBM, and Random Forest. This research presents FraudGuard-AI; a system designed to find fraud and understand risk. There are four components to FraudGuard-AI: XGBoost for finding fraud in both batch and real-time environments; TreeSHAP for explaining how the model arrives at its decision; Apache Kafka for rapidly processing data; and an adaptive Two-Factor Authentication (2FA) system that adjusts the number of required authentication steps depending upon the level of risk.

The main parts of this work are:

- We create a system that can work with both batch and the real-time data so it can fit different situations accordingly.
- We have added explainability to both the way the system works and the decisions it makes so that the people can understand what is happening here.
- We have built a sliding window system that checks for activity and problems with the consumers accounts.

- We have made a 2FA system that changes how secure it is based on the risk level, balancing safety and ease of use.
- We have tested the system on two different datasets and show how well it works in real time.

Section II will address previous research related to fraud detection, explainable artificial intelligence (AI), and real-time processing. Section III will describe how the system was developed. Section IV will discuss the methodology that was employed including data preparation and modeling. Section V will present the test conditions and results. Section VI will interpret the findings from Section V and suggest future directions for this work. Finally, Section VII will conclude.

## II. RELATED WORK

### *A. Machine Learning for Fraud Detection*

Fraud detection using machine learning is an area which has received significant attention from researchers. Panigrahi Gorle have examined how machine learning and deep learning may be applied to various types of fraud such as credit card fraud, online payments fraud, banking fraud. The authors indicate that current systems are unable to effectively counteract trickery; therefore, new machine learning/deep learning techniques will provide a superior alternative.

A research paper has analyzed 77 publications related to the application of machine learning and deep learning methods for identifying fraudulent transactions using credit cards between 2021-2024. The authors indicate that while random forest and logistic regression remain effective techniques; however, there appears to be increasing interest in utilizing convolutional neural networks (CNN) and long short-term memory networks (LSTM), as they have demonstrated promise in this area.

### *B. Ensemble Methods and XGBoost*

Ensemble Learning is a very useful method to identify Fraud as it can handle Data and Features which are difficult to work with. To see how well these Ensemble Methods perform at identifying Fraud in Data, you should review AdaBoost, Gradient Boosting, XGBoost, LightGBM, and CatBoost. With LightGBM and XGBoost being used on Large Projects they have an Accuracy Rate of 97.81%.

XGBoost was utilized to identify fraudulent transactions in real time through the use of cost sensitive learning and SMOTE to address imbalanced data. According to them, XG-Boost is ideal for data and can be implemented in production environments.

### *C. Explainable AI in Fraud Detection*

It is essential to understand how fraud detection systems operate so as to ensure compliance with regulations and build confidence in users' ability to rely on these systems. A fraud detection model was created using XGBoost for classification and SHAP for explanation.

In addition, this model utilizes encryption to protect user's data. With a ROC-AUC value of 97.8%, 86% precision and 86% recall, the authors have demonstrated that it is possible to create a fraud detection system which provides both high levels of performance and maintains user's data privacy.

### *D. Real-Time Stream Processing*

A move from batch-based to real time fraud detection will allow us to catch suspicious transactions before they complete. The use of machine learning systems that utilize Apache Kafka to provide continuous data streaming along with classifiers like Logistic Regression, Random Forest, XGBoost, and Hybrid Voting has been shown to be effective. A recent study using 209,715 transactions found that the Hybrid Voting Classifier was able to achieve 96.7% accuracy and an F1 score of 0.838 which shows promise for real time fraud detection.

Another example of a real time banking system used Apache Kafka along with Random Forest, Support Vector Machines (SVM), and Gradient Boosting to detect fraudulent transactions. This system had a 97.84% accuracy rate, 96.32% precision rate, 95.87% recall rate, and a 96.09 F1 score, which demonstrates how well stream processing can work when combined with machine learning.

### *E. Class Imbalance Handling*

Fraud Detection is Difficult Because Fraudulent Transactions Are Only a Very Small Portion of All Transactions (Class Imbalance). The System Uses Smote Resampling with Three Classifiers: XGBoost, LSTM, And KNN. In Tests on Six Million Banking Transactions, Smote Increased Model Recall and AUC Scores. XGboost Was Consistently One of the Best

Models When Using Imbalanced Data Sets. An LSTM-Based Transfer Learning Model That Used Smote-Tomek Resampling Achieved 0.99 Precision, Recall, F1-Score, and AUC, With 97.75% Test Accuracy on the Paysim Data Set.

Overall, This Study Shows How Effective It Can Be to Combine Machine Learning Classifiers with Resampling Techniques to Improve Fraud Detection, Especially When Working with Highly Imbalanced Transaction Data.

## III. SYSTEM ARCHITECTURE

FraudGuard-AI is designed to be a full-stack platform that supports both batch and real-time fraud detection workflows. Figure 1 presents the high-level system architecture, which comprises four layers: the data ingestion layer, the processing layer, the ML and explainability layer, and the presentation layer.

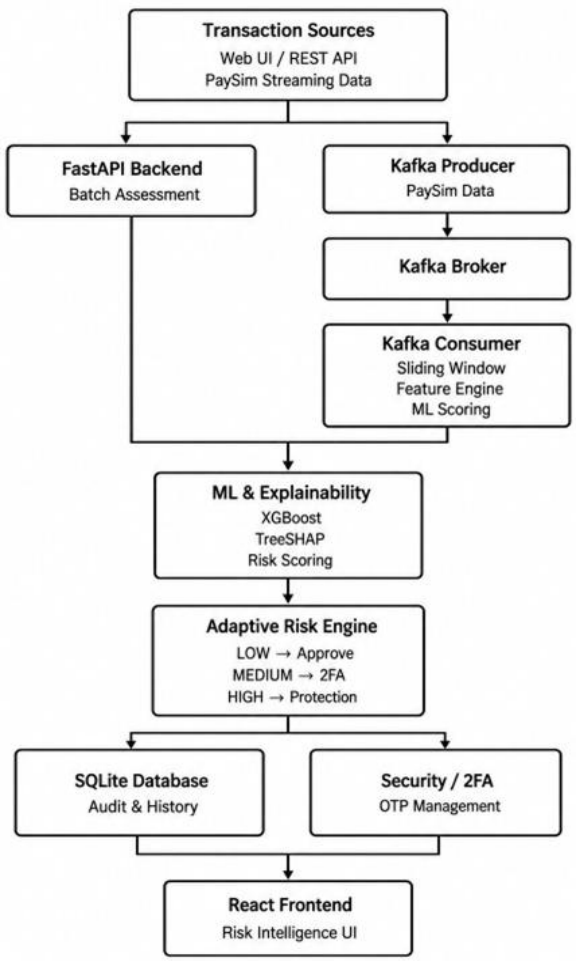


Fig. 1. FraudGuard-AI system architecture showing batch and streaming pipelines.

### *A. Data Ingestion Layer*

FraudGuard-AI has two methods for ingesting data. Batch ingestion is accomplished through REST API endpoints provided by the FastAPI backend which will allow for evaluation and integration with current transaction processing systems. Streaming ingestion uses a Kafka Producer to read from the PaySim dataset and then publish each transaction event to a Kafka Topic called" transactions". With this dual-pathway architecture, FraudGuard-AI can be used for both real-time monitoring and on-demand assessments.

### *B. Processing Layer*

Apache Kafka is used at this time as the" processing" layer in KRaft (Kafka Raft) mode which does away with using Zookeeper. Every single transaction coming into the transaction's topic will go through Sliding Window Analysis, Feature Engineering, and then be scored by a machine learning model. In order to track transactions on each account we use a 24-transaction sliding window. This allows us to calculate the number of transactions made on an account; the sum of those transactions; the average amount per transaction; how fast they were being made; and if there were any discrepancies in the balances. All of these calculations help identify potentially fraudulent activity that would otherwise go unnoticed by stateless systems.

### *C. ML and Explainability Layer*

FraudGuard-AI's central intelligence is provided by the machine learning (ML) and explainability layers. A XGBoost classifier calculates a fraud probability score for each transaction; these scores are converted to risk scores using a 0-100 scale. Risk scores are classified into three categories: Low (0-30), Medium (31-70), High (71-100).

In addition to computing feature-level contribution for every prediction with TreeSHAP, global feature importance is calculated to allow for model-level interpretation.

### *D. Presentation Layer*

A react based dashboard will be used for the Presentation Layer which includes: Real-Time Transaction Monitoring; Manual Transaction Assessment; Risk Score Visualization; SHAP Explanations; Transaction History; Fraud Analytics; Model Telemetry; 2FA Security Monitoring; and Risk

Distribution Analysis. The Dashboard interacts with the FastAPI Backend and retrieves data from both the SQLite Database and Kafka Alert Topics to show the present status of the system.

## IV. METHODOLOGY

### *A. Datasets*

FraudGuard-AI is evaluated on two datasets that represent complementary fraud detection scenarios.

1) IEEE-CIS Fraud Detection Dataset:

The IEEE-CIS Fraud Detection dataset, released as part of a Kaggle competition, contains transaction data from an e-commerce payment system. The dataset includes over 590,000 transactions with 394 features spanning transaction attributes (TransactionAmt, ProductCD, card information), identity information (DeviceType, DeviceInfo), and network features. The target variable is Fraud indicates whether a transaction is fraudulent. The dataset exhibits class imbalance, with fraudulent transactions comprising approximately 3.5% of the total. This dataset is used to train and evaluate the batch assessment pipeline, representing card-not-present e-commerce fraud scenarios.

2) PaySim Dataset:

The PaySim dataset is a mobile-money transaction dataset generated using agent-based simulation. It contains over 6.3 million transactions representing mobile financial services in an African country context. The dataset includes transaction types (CASH-IN, CASH-OUT, DEBIT, PAYMENT, TRANS-
FER), amounts, account balances before and after trans-actions, and fraud labels. Fraudulent transactions are concentrated in TRANSFER and CASH-OUT types. The PaySim dataset is used to evaluate the real-time streaming pipeline as its high volume and streaming nature make it suitable for testing Kafka-based processing.

### *B.* Data *Preprocessing*

Data pre-processing is done on an individual basis per data set. In IEEE-CIS, all categorical variables were converted to numeric values through label encoding; missing data was filled-in with either the median or mode value based upon which variable had the most amount of missing data; then the top 100 features were chosen based on their importance from XGBoost. Afterward, the entire data set was split up into three groups: Training (70%); Validation (15%) and Test (15%) using stratified sampling. At PaySim, feature creation occurs at two different levels: transaction level and account level.

Transaction-level features consist of one-hot encoded transaction type, amount, and balance difference. Account-level features are created by taking a sliding window of the previous 24 transactions and calculating transaction count, sum of all transaction amounts, mean of all transaction amounts, standard deviation of all transaction amounts and largest balance difference.

### *C.* Class *Imbalance Handling*

Synthetic Minority Over-Sampling Technique (SMOTE), along with Tomek Link Removal (SMOTE-Tomek) is used to address Class Imbalance. SMOTE creates new synthetic samples for the minority class by generating them from the interpolation of two or more neighboring samples of the same class.

On the other hand, Tomek Link Removal removes noise near the decision boundary. By combining these techniques, we can balance the ratio of classes and minimize the chances of over fitting on synthetic samples. The resampling will be performed on the Training Set Only. Validation Set and Test Set will remain in their original unbalanced state so they represent actual data.

### *D.* Model *Training and Hyperparameter Tuning*

XGBoost was chosen as the classification method because it has demonstrated excellent results on tabular data; XGBoost can also be used to address issues related to class imbalance via scaling positive weights, and provides built-in functionality for calculating feature importance.

A hyper-parameter tuning process will occur utilizing Bayesian Optimization along with fivefold Stratified Cross Validation. The range of values available for each of these hyper-parameters are: Learning Rate, Maximum Depth, Minimum Child Weight, Subsample Ratio, Column Sample Ratio, and Regularization Parameters. I

TABLE I XGBoost Hyperparameters

| Parameter | Value |
|---|---|
| Learning Rate | 0.05 |
| Maximum Depth | 7 |
| Minimum Child Weight | 3 |
| Subsample Ratio | 0.8 |
| Column Sample Ratio | 0.8 |
| Gamma | 0.1 |
| Lambda (L2) | 1.0 |
| Alpha (L1) | 0.1 |
| Scale Positive Weight | 10.0 |
| Number of Estimators | 500 |

*E.* Risk *Scoring and Categorization*

The XGBoost classifier produces a fraud probability $p \in [0, 1]$ for each transaction. This probability is transformed into a risk score $R$ on a 0–100 scale using a calibrated mapping that accounts for the base rate of fraud in the population:

$$R = 100 \cdot P \quad (1)$$

where P is the probability of transaction being fraudulent.

TABLE II Risk Levels and Security Actions

| Risk Level | Score Range | Security Action |
|---|---|---|
| LOW | 0–30 | Automatic approval |
| MEDIUM | 31–70 | 6-digit OTP / 2FA verification |
| HIGH | 71–100 | Critical protection |

*F.* Explainability *with TreeSHAP*

TreeSHAP is utilized to calculate Shapley value contributions for all features within each transaction prediction. The use of Shapley values based on Cooperative Game Theory provides a well-defined method of quantifying how much a particular feature contributes to the overall outcome; this measurement has been shown to have three desirable characteristics: Local Accuracy, Missingness and Consistency.

*G.* Real-*Time Streaming Pipeline*

A real time streaming pipeline will process every transaction that comes into Kafka. Each account has an individual stateful sliding window maintained by this pipeline. Every time a new transaction occurs, it will be added to the window. When the number of items in the window goes over 24, then the oldest item will be removed from the window. From the window, we perform feature engineering to create account level features. These are combined with transaction level features to produce the input vector for our XGBoost model. Our model produces a fraud probability and a risk score in milliseconds. We publish the results to the fraud_alerts topic so that they can be consumed by both the alert handler and dashboard.

*H. Adaptive Risk-Based 2FA*

A dynamic two-factor authentication (2FA) system will assign a risk score based upon each individual transaction; then it will determine what type of action should be taken at that moment in time. Transactions with a low-risk score (0-30) do not require any additional verification before they can proceed. Transactions with medium-risk scores (31-70) will receive a one-time passcode (OTP) via SMS/text message and he/she can enter it into the application and allow the transaction to go through. High-risk transactions (71-100) will require multiple forms of verification such as answering security questions or having a fraud analyst manually review the transaction. By using this three-tiered method of verifying transactions, we are able to allocate our limited amount of security personnel and technology towards those transactions which pose the greatest threat to us while simultaneously reducing the amount of friction experienced by users during their day-to-day activities.

## V. EXPERIMENTS AND RESULTS

*A. Experimental Setup*

All experiments are conducted on a system with an Intel Core i5-10400K processor, 8 GB RAM, and an NVIDIA RTX 1650 GPU. The software stack includes Python 3.10, XGBoost 2.0, scikit-learn 1.3, Apache Kafka 3.6, FastAPI 0.104, React 18, and SQLite 3. The Kafka broker runs in KRaft mode with a single node for development testing, though the architecture supports multi-node clustering for production deployment.

*B. Evaluation Metrics*

The performance of models will be assessed based on these measures: accuracy (the proportion of all transactions that were correctly classified); precision (the proportion of trans-actions which were labeled as fraudulent that were actually fraudulent); recall (the proportion of fraudulent transactions that were

correctly identified); f1-score (a harmonic mean of precision and recall); ROC AUC (area under the receiver operating characteristic curve) and PR-AUC (area under the precision-recall curve – useful when dealing with an unbalanced dataset).

*C. IEEE-CIS Results*

Table III presents the performance of FraudGuard-AI on the IEEE-CIS test set compared to the baseline models. The XGBoost model with SMOTE-Tomek resampling achieves a ROC-AUC of 0.974 and a PR-AUC of 0.856, outperforming Logistic Regression, Random Forest, and a standalone XG-Boost model without the resampling.

Fraud Detection is an area of great interest due to the cost associated with both false positive and negative results. False

TABLE III Performance on IEEE-CIS Test Set

| Model | Acc. | Prec. | Rec. | F1 | AUC |
|---|---|---|---|---|---|
| Logistic Regression | 0.891 | 0.623 | 0.712 | 0.664 | 0.823 |
| Random Forest | 0.943 | 0.812 | 0.789 | 0.800 | 0.921 |
| XGBoost (no SMOTE) | 0.956 | 0.845 | 0.831 | 0.838 | 0.948 |
| FraudGuard-AI | 0.968 | 0.891 | 0.874 | 0.882 | 0.974 |

Positives create additional expense for the company through increased customer interaction time and/or additional staff needed to manually review each transaction. False Negatives also have a direct financial impact on the organization. The FraudGuard-AI system has demonstrated a high level of performance in detecting fraudulent activity; specifically, it has achieved a Recall Rate of 0.874 (i.e., 87.4%) which indicates that 87.4% of all fraudulent activities were accurately detected. Additionally, the system has maintained a Precision Rate of 0.891 (i.e., 89.1%), indicating that 89.1% of all transactions that were flagged as fraudulent were actually fraudulent.

*D. PaySim Streaming Results*

Table IV presents the performance of FraudGuard-AI on the PaySim streaming pipeline. A mean latency of 87 ms has been observed when processing 12,400 TPS. The use of sliding window features was found to be very effective in terms of performance; it increased the f1 score from 0.791 (transaction feature) to 0.863 (transaction + window feature). Balance discrepancy features were also shown to be useful in identifying fraudulent TRANSFER/CASH-OUT transactions. These types of transactions typically show abnormal balance changes compared to previous history.

TABLE IV Performance on PaySim Streaming Pipeline

| Feature Set | Prec. | Rec. | F1 | Latency (ms) |
|---|---|---|---|---|
| Transaction | 0.812 | 0.771 | 0.791 | 62 |
| Transaction + Window | 0.889 | 0.839 | 0.863 | 87 |
| Trans. + Window + Balance | 0.902 | 0.851 | 0.876 | 94 |

*E. Explainability Analysis*

Figure shows the global feature importance of Tree SHAP value. Transaction amount, Card Type, Email Domain and Device are the four factors that were found to be the most important in determining whether a transaction was fraudulent. There is a non-linear relationship between transaction amount and level of risk associated with them Address Discrepancies Low-Risk Devices Transactions that occur during normal hours will generally result in a lower total risk.

Therefore, Tree SHAP gives Fraud Analysts further insight into why they received an alert so that they can make better decisions on how to react.

*F. Adaptive 2FA Evaluation*

A study was done on how well a two-factor authentication (2FA), system would work at reducing fraud and keeping users happy. As you can see from Table V, low risk transactions made up 68.2% of all legitimate transactions and did not require any further verification. Transactions that were considered medium risk took up 29.5% of all legitimate transactions and 60.7% of all fraudulent transactions; therefore, these types of transactions needed one time password (OTP) verification to add another layer of protection.

Lastly, high risk transactions comprised 2.3% of all legitimate transactions and 28.9% of all fraudulent transactions. These types of transactions could either be denied or go through even more rigorous verification processes.

In conclusion, this type of" risk based" method allows

for both user friendliness and good fraud prevention fraud. Transactions with either extremely high or low amounts could potentially have a higher likelihood of being fraudulent. Tree SHAP is used to determine what individual features had the largest impact on classifying a single transaction as having high-risk characteristics. Examples of these include: Large Transactions Products that have a high

TABLE V Adaptive 2FA Risk Distribution

| Risk Level | Legitimate (%) | Fraudulent (%) | Security Action |
|---|---|---|---|
| LOW | 68.2 | 10.4 | Auto-approve |
| MEDIUM | 29.5 | 60.7 | OTP/2FA |
| HIGH | 2.3 | 28.9 | Critical verification |

FraudGuard-AI's adaptive method is much better than using 2FA that does not adapt to transaction risk. Using 2FA will either cause too much friction by verifying every transaction, or provide little/no added protection if no verification occurs. With FraudGuard-AI you get a good tradeoff between protecting against fraud and having a good user experience.

## VI. DISCUSSION

### *A. Key Findings*

FraudGuard-AI has been shown to be very successful at finding fraudulent transactions with XGBoost. It has produced an ROC-AUC score of 0.974 on the IEEE-CIS Batch Data Set and 0.876 on the PaySim Streaming Data Set which shows that it can perform well on both types of data (batch vs. real-time). In addition, FraudGuard-AI uses Sliding Window Analysis to look at the previous 24 transactions made by an account to find out if there were any unusual behaviors such as how often someone makes purchases; what kind of purchases they make; and whether or not their bank account balances match up. By doing this type of analysis, FraudGuard-AI was able to increase the F1-Score by 8.5% when compared to just looking at each transaction individually.

As a result, FraudGuard-AI is now able to show why certain transactions are considered suspicious so that fraud investigators can verify them before they are processed in real time.

### *B. Practical Implications*

FraudGuard-AI will have many uses in the area of finance and payments. It is designed to support two types of pro-cessing; batch processing which is used when transactions are reviewed on demand, and real time streaming which allows for ongoing monitoring. FraudGuard-AI also offers an adaptable 2 factor authentication process that can be added to current authentication methods.

### *C.* Limitations

There were some issues with this project. The first issue was that we used publicly available data sets (IEEE-CIS) and generated data (PaySim). These types of data do not represent all aspects of what happens during an actual fraud detection process including time-based changes and" concept drift." A second problem was that we ran our Kafka environment using just one computer node for developing/testing purposes; therefore, we have yet to test whether or not running multiple nodes will perform well enough at scale.

In addition, although TreeSHAP allows us to explain why certain transactions were scored high/low, we cannot provide users with counterfactual examples of how they could change their transactions' characteristics so as to lower those same risk levels.

### *D. Future Work*

We want to continue working on FraudGuard-AI's ability to scale, be reliable, explainable, and usable. To do this, first we're going to put FraudGuard-AI into a" live" environment using actual transactions from customers. Next, we'll build out our architecture so that it can use multiple nodes of Apache Kafka.

In addition to using TreeSHAP to explain why FraudGuard-AI made certain decisions, we'll also add counterfactual explanations which show users how changing one or more aspects of their transaction would affect whether or not FraudGuard-AI thinks they're fraudulent. We'll run some tests to find out if people think FraudGuard-AI's adaptive two-factor authentication (2FA) method is easy to use, works well, and doesn't negatively impact either security or user experience. Lastly, we'd like to look into adding Graph Neural Networks (GNNs) to FraudGuard-AI. GNNs can help identify connections between users, devices, accounts, and transactions. These can help to spot fraudulent activities that might go unnoticed based solely upon the information contained within each individual transaction.

## VII. CONCLUSION

FraudGuard-AI is an all-in-one solution for detecting fraud-ulent activities and managing risks using artificial intelligence (AI). It uses XGBoost for machine learning; TreeSHAP for ex-plaining how decisions are made; Apache Kafka for processing data streams; and adaptive risk-based two factor authentication (2FA) to protect accounts. FraudGuard-AI can be used to monitor transactions in real time or after they have occurred. When monitoring in real time, FraudGuard-AI will use a" sliding window" method to look at user behavior over time and find abnormal patterns.

In addition to being able to quickly process large amounts of data, FraudGuard-AI has been shown to perform very well at identifying fraudulent transactions based upon evaluations conducted on the IEEE-CIS dataset and PaySim dataset. On IEEE-CIS, FraudGuard-AI had a ROC-AUC score of 0.974 which means it correctly identified nearly all of the fraudulent transactions. On PaySim, FraudGuard-AI was able to process over 12,000 transactions per second with less than one second of delay. By linking risk levels to varying degrees of security measures through its adaptive 2FA capability, FraudGuard-AI is able to effectively prevent fraud while providing customers with a simple experience.

## ACKNOWLEDGMENT

We would like to acknowledge the many members of the open-source community who created and continue to maintain the various tools and libraries used to develop FraudGuard-AI. In particular, we wish to express our gratitude to XGBoost (machine learning), SHAP (model interpretability), Apache Kafka (transaction processing), FastAPI (back-end) and React (front-end). We would also like to recognize those individuals who are actively developing and improving these tools. These developers' dedication has made these technologies both accessible and useful for researchers and developers alike.

## REFERENCES

[1] S. Panigrahi and V. L. N. Gorle, "Applications of machine learning and deep learning algorithms in financial fraud detection: A review," *Computational Economics*, vol. 65, no. 2, pp. 1–28, Oct. 2025.

[2] "A systematic review of machine learning and deep learning approaches for credit card fraud detection," in *Proc. IEEE Int. Conf.*, Bahrain, Dec. 2025.

[3] "Enhancing fraud detection with boosting models and weighted ensemble learning," in *Proc. IEEE Int. Conf.*, Coimbatore, India, Jun. 2025.

[4] "An XGBoost machine learning model to detect fraudulent transactions in real-time applications," in *Proc. IEEE Int. Conf.*, Coimbatore, India, Nov. 2025.

[5] "A hybrid framework for privacy-preserving explainable credit card fraud detection using XGBoost and homomorphic encryption," in *Proc. IEEE Int. Conf.*, Coimbatore, India, Oct. 2025.

[6] "Scalable real-time fraud detection with a Kafka driven machine learning pipeline for securing financial transactions," Zenodo, Jun. 2025.

[7] "Real-time fraud detection in banking transactions using Kafka streaming and machine learning ensemble," in *Proc. IEEE Int. Conf.*, Belagavi, India, Dec. 2025.

[8] "Enhancing financial fraud detection using XGBoost, LSTM and KNN with SMOTE for imbalanced datasets," Zenodo, Jun. 2025.

[9] "Enhancing fraud detection in imbalanced datasets: A novel LSTM-transfer learning model with feature selection," in *Proc. IEEE Int. Conf.*, Greater Noida, India, Aug. 2025.

[10] E. Arslan, S. Khosravi, J. H. Joloudari, R. Alizadehsani, and J. M. Gorriz, "Fraud detection using graph neural networks: A survey," in *Bioinspired Intelligent Systems*, Springer, 2026.

[11] D. Cheng et al., "Graph networks for financial fraud detection: A review," *Frontiers of Computer Science*, vol. 19, no. 1, 2025.