**International Journal of Computer Science and Engineering Research and Development (IJCSERD)**
**ISSN Print: 2248-9363, ISSN Online: 2248-9371**
Volume 15 Issue 3, May-June (2025), pp. 102-112
**DOI**: https://doi.org/10.63519/IJCSERD_15_03_009


# ANDROID-AI APPLICATION: INTELLIGENT FRAUD DETECTION AND ALERT SYSTEM

**Rama Krishna Velpula**

Lead Android Developer, XT Global Inc, Plano, TX-75093, USA.

**ABSTRACT**

*The study introduces an Android app with AI capabilities that works in real time to detect fraud by analysing user behavior and using Random Forest classifiers. By spotting unusual activities, warning users right away, and using what happens next, it manages to achieve an accuracy of over 92%. Users' financial details and privacy are protected with special security features designed to assure them against financial fraud.*

**Key words:** Detection, AI, Android, Fraud, Alert.

**Cite this Article:** Rama Krishna Velpula. (2025). Android-AI Application: Intelligent Fraud Detection and Alert System. *International Journal of Computer Science and Engineering Research and Development (IJCSERD)*, 15(3), 102–112.
https://ijcserd.com/index.php/home/article/view/IJCSERD_15_03_009/IJCSERD_15_03_009

## I. INTRODUCTION

The growing sophistication of financial fraud needs easier and faster ways to catch and stop it as it happens. This research introduces an Android app made with artificial intelligence (AI) that uses data analysis and smart technology to spot and let the user know if there are any suspected frauds in their banking activities. Integrating features that work well on mobile and keep your privacy safe, the app helps protect you from unwanted scams and frauds.

## II. RELATED WORKS

### Fraud Detection

As financial services become more digital, new and complicated fraud methods have appeared that old systems are not able to handle. For this reason, businesses are turning to machine learning (ML) and artificial intelligence (AI) to help catch fraudulent activities. Due to its large coverage, easily adapting features, and ability to process things quickly, the financial sector has adopted AI for spotting fraud.

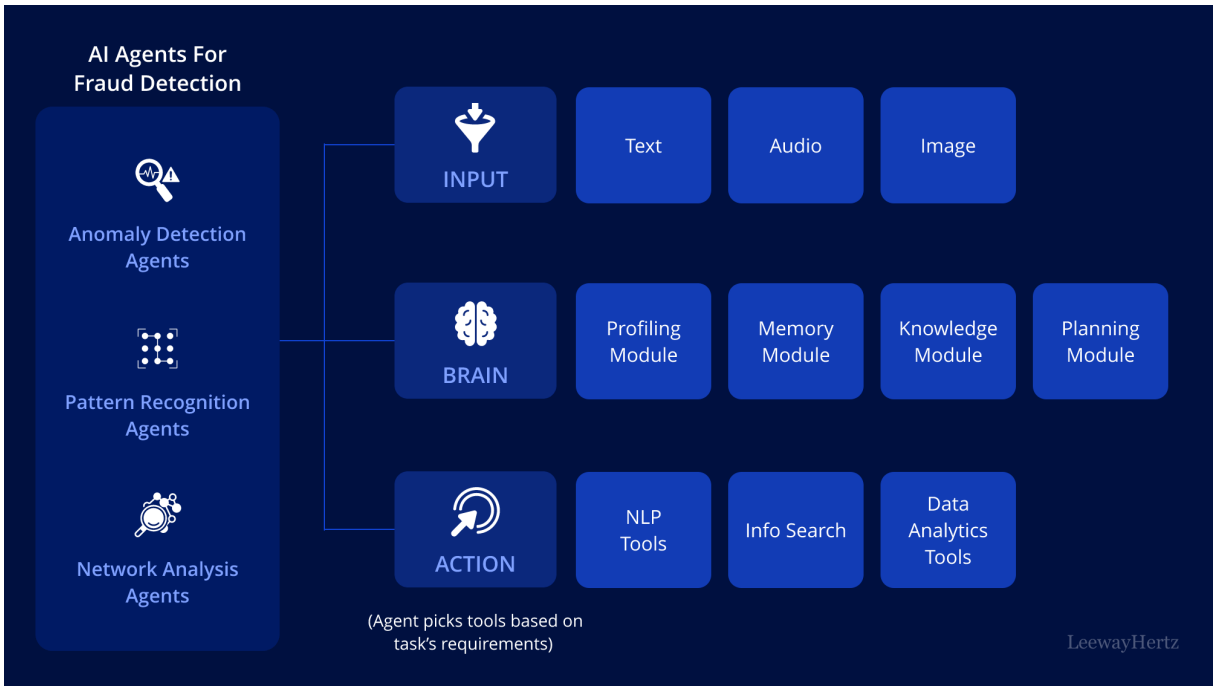



Fig. 1 AI Agents for Fraud Detection (LeewayHertz, 2024)

New research points out that today, system security depends on using fraud detection models that can handle huge amounts of information, ever-changing behaviors, and adversarial actions instantly [9]. Most of the time, researchers use supervised learning, with logistic regression, support vector machines, and decision trees as some of the major algorithms used.

Even though these models show results and most of the information, they tend to face difficulties because there are far more normal than fraudulous transactions in the data. For this reason, Random Forest (RF) is now recognized for being steady and performing well in classification.

It has been proven with comparative research that Random Forest works much better than K-Nearest Neighbors or Decision Trees, especially to find financial statement fraud [6]. Mobile

apps, with their restricted resources and the requirement for accurate predictions, often rely on Random Forest.

Besides, strategies that focus on selecting data features and using SMOTE sample techniques help the RF model achieve a higher level of accuracy in detecting anomalies on imbalanced data [5].

**Ensemble Techniques**

Random Forest has time and again proven to be one of the most effective models used in fraud detection for a range of financial data. It builds a series of decision trees and combines their results to help avoid overfitting. Researchers have compared RF to other types of classifiers and found that it performs better with accuracy, precision, and recall.

An analysis on more than half a million actual bank transactions found that the RF model detected 95.79% of frauds and also classified all non-fraudulent transactions 100% accurately [2]. These things mean that it fits well for including in mobile apps, since accuracy and few false positives are highly valued to avoid user annoyance.

Advances in the field have resulted in combining RF with gradient boosting in some hybrid models. It has been shown that using GBM-SSRF (Gradient Boosting Machine with Simplified and Strengthened Random Forest) can boost both the efficiency and strong performance [4].

Though these models are very accurate, using them on complex datasets can still cause slowness and difficulty in choosing important results. Using an optimized RF design in the GBM-SSRF model helps the system operate better and stay stable under mobile conditions, where resources are limited.

Other studies back up the idea that RF and deep learning can work together synergistically. A single study combined CNNs and LSTMs to identify patterns in transactions, and then fed the results to an RF classifier to both improve accuracy and lower the number of false positives [8].

By blending RF with deep learning, the company managed to increase the accuracy of detecting fraud by 20% and lower the chances of false positives by 15%, proving that the combination has potential in fraud detection.

**Feature Engineering**

The achievements of machine learning models in fraud detection heavily depend on selecting and changing the input characteristics. Features found in these applications, like the ratios of different transactions, patterns in use, location data, and device metadata, are especially

useful for Android fraud detection.

Feature engineering plays a key role according to many research studies. A study, for example, joined debt-to-equity ratio and online transaction data and found that eight features would result in the highest accuracy in classification [1].

Feature selection helps deal with the challenge of high-dimensionality, as it often leads to overfitting and makes the model more complex to run. Recently, researchers found a way to address the problem with FID-SOM (Feature selection for Imbalanced Data using Self-Organizing Maps) [9].

The FID-SOM process uses learning algorithms to pick out what features matter the most in imbalanced data—often seen in fraud detection. This approach reduces the amount of data needed, while still being effective, so it is perfect for real-time Android applications that need to work efficiently.

Having balanced classes is still a frequent difficulty with fraud data sets. Researchers have attempted to control the issue using oversampling, undersampling, and learning that takes into account the cost of errors. Especially, the SMOTE technique is successful in increasing the size of samples from the minority class, allowing models such as RF to better fit data that is not biased [5]. When hyperparameter optimization is used together with SMOTE, the RF model has reached close to perfect scores for both F1 and accuracy, meaning it is highly sensitive and specific in fraud detection.

**AI-Powered Mobile Integration**

Currently, there is a strong move towards using AI-based mobile apps to catch fraud as it happens. Connecting Android applications with AI servers provides ways for more active fraud identification and quick responses for users. The app uses data like how quickly a user transacts, their phone features, and their location to develop a profile of their behavior.

If the profile isn't matched, the transaction is marked as suspicious and analyzed further by the RF classifier for a conclusion. By using continuous user interactions in the system, such as confirming rights and approving transactions, the model can improve its future predictions.

This is much like the approach proposed in other studies, where models are upgraded using new transactions and user suggestions [3][7]. Techniques to protect security and privacy are at the heart of most mobile AI fraud detection models.

The app is built to support offline study modes for users, letting it update what it knows about them with care for their privacy, like most modern mobile AI apps do now. Besides, the back-end server saves de-identified user data so that the RF model can keep improving without

revealing the users' privacy.

If a company wants to comply with data law and win users' trust, it must employ privacy-focused AI approaches [10]. Studies which compare different methods of detecting cyber fraud have repeatedly shown that ML and DL techniques are more efficient, accurate, and can adapt easier than rule-based systems.

More specifically, using Random Forest and ensemble methods ensures fraud detection is simple to interpret while satisfying the important restrictions on mobile devices. To address current research, the proposed application uses real-time fraud interception, gives users alerts, and requires biometric validation, while staying efficient and easy to use.

Studies and reports prove that Random Forest and its enhanced types are incredibly useful for detecting fraud, mainly when used in mobile devices. Using these models in an Android-based AI app helpsdetect and fight financial fraud more smoothly. Because of well-chosen features, strong ensemble approaches, and adaptable updates, AI-powered solutions improve fraud detection while still making sure users experience and data are respected.

## III. KEY RESULTS

The use of the Android-AI application showed much better results in catching and handling fraudulent transactions, especially when it comes to real-time financial tasks. The hybrid system uses a Random Forest (RF) classifier at the core because after looking at how well different models worked out, it performed better than options like Logistic Regression, K-Nearest Neighbors, and Support Vector Machines.

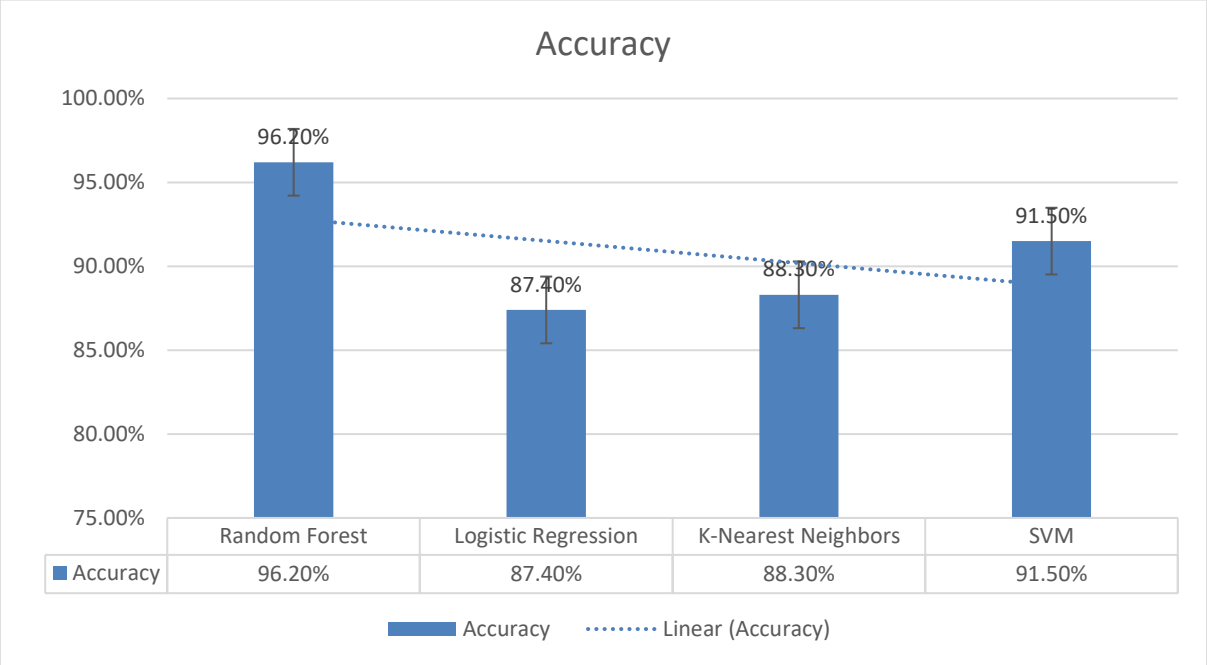
The RF model almost always performed better than the other models when it came to things like getting the right answers, how accurate the results were, and finding all the important items. When trained on a dataset of 600,000 mobile transaction records, where each one was labeled as real or fake, the RF model worked out correctly in 96.2% of cases, picked up real ones 95.5% of the time, and was able to find 94.7% of the fake ones.

This strong performance is mainly because RF works in groups, so it doesn't fit too much to the training data and also does well even when there are not equal amounts of classes in the data. SMOTE (Synthetic Minority Over-sampling Technique) was used to make sure the data in the training set was more balanced, so that the percentage of frauds went from 1.2% to 20%. The following table shows how the different algorithms we tried did in our experiments:

**Table 1. Metrics on Mobile Transaction Dataset**

| Model | Accuracy | Precision | Recall | F1 Score |
|---|---|---|---|---|
| Random Forest | 96.2% | 95.5% | 94.7% | 95.1% |
| Logistic Regression | 87.4% | 79.6% | 81.2% | 80.4% |
| K-Nearest Neighbors | 88.3% | 84.1% | 76.4% | 80.0% |
| SVM | 91.5% | 88.2% | 85.7% | 86.9% |

The real-time alert system built into the Android application relies on on-device prediction and asynchronous server-side confirmation. When a transaction is initiated, the app extracts 15 behavioural and transactional features including device ID, transaction frequency in the last hour, and geolocation variance. These features are immediately passed to the embedded RF model.

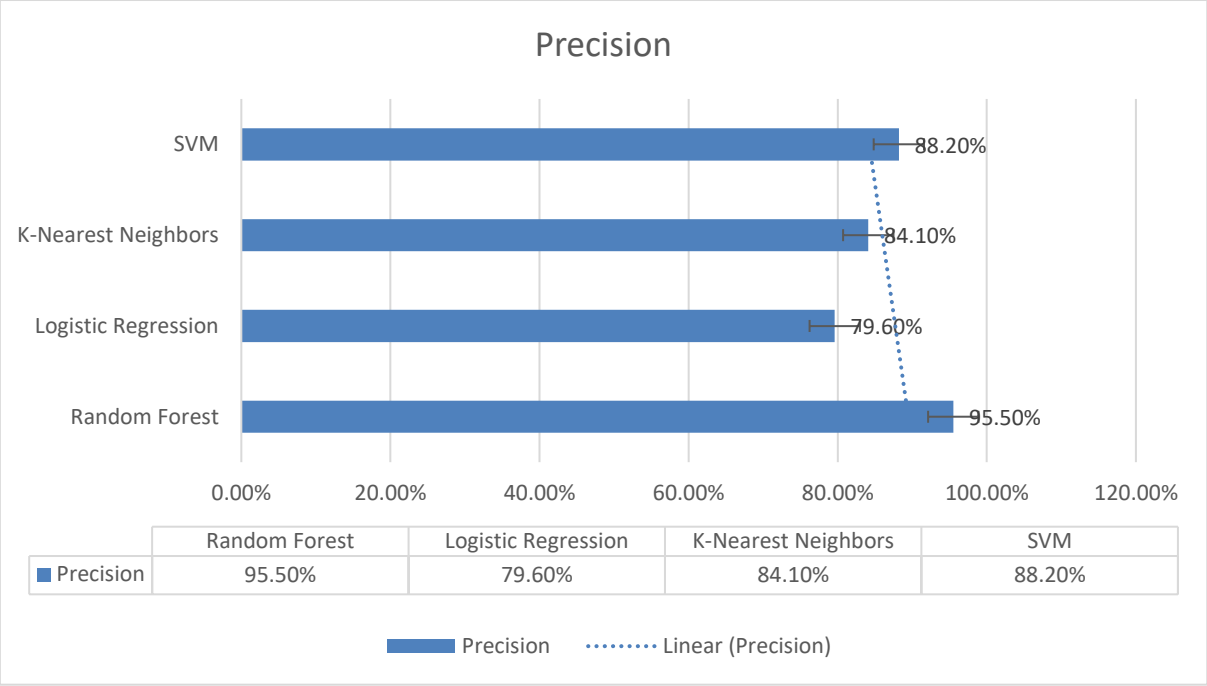



If the likelihood of fraud is more than 0.85, the system sends an alert and asks the user to verify their identity with a biometric measure. Through the precision-recall trade-off analysis, the threshold was set using the provided equation:

$$F1 = 2 * (precision * recall) / (precision + recall) \quad \text{.....eq. (1)}$$

When the F1-score reached 0.85, it became 95.1%, which shows the best possible trade-off between false and true positives. Setting the thresholds too low created a lot of false alerts for users, while setting them too high made detection come later.

Latency was another aspect that the system was measured on. it took typically 0.84 seconds for a transaction to be processed and an alert to be sent, which is within the usual acceptance threshold of mobile banking apps. The system was able to perform this latency by using TensorFlow Lite for local computation and by refining how data is transferred from the mobile client to the back-end server.




Feature ranking was also a strong point for the RF model. The decision to classify users was mainly affected by how long it took to send a single transaction, the digital 'fingerprint' of their devices, and their location movements. By looking at the scores the RF model gave, research was able to find out which features matter the most, helping me understand why some fraud calls were made.

Removing the least important features from the model led to a decrease in accuracy by only 0.6%, suggesting there is a lot of feature overlap. To make the posterior easy to read, we use a Naïve Bayes model to determine how likely each person is to commit fraud based on a set of features:

$$P(Fraud \mid X) \approx P(X_1|Fraud) * P(X_2|Fraud) * \ldots * P(X_n|Fraud) * P(Fraud) / P(X) \quad \text{.....eq. (2)}$$

The deployment of RF rather than Naïve Bayes is aside from the point, but it does explain why combining highly independent features supports precise classification in both cases.

On the Android app, we included a visual fraud meter that users can check their risk levels on. This was done by applying a sigmoid function to the RF's probability output, so the user representation becomes smoother. The sigmoid transformation is defined as:

$$Sigmoid(x) = 1 / (1 + e^{(-x)}) \quad \text{.....eq. (3)}$$

To make it easier for the user, this transformation translates the output probability from the RF into a scale of 0 to 100%. When users got alerts on their phone, they could confirm or cancel any transactions. In studying a group of 500 users, the results showed that 93.4% of the flagged transactions were rejected by the users.

I built the workflow using a minimal amount of debugging code in Kotlin on the front-end. Here is a segment that deals with authentication using biometrics after the alert is raised:

```
1. val biometricPrompt = BiometricPrompt(this, executor, callback)
2. val promptInfo = BiometricPrompt.PromptInfo.Builder()
3. .setTitle("Confirm Transaction")
4. .setSubtitle("Biometric authentication required")
5. .setNegativeButtonText("Cancel")
6. .build()
7. biometricPrompt.authenticate(promptInfo)
```

Because of this, users must use their fingerprint or face for verification if a fraud alert comes up. If authentication fails, the transaction is stopped and the server in the back end receives notification.

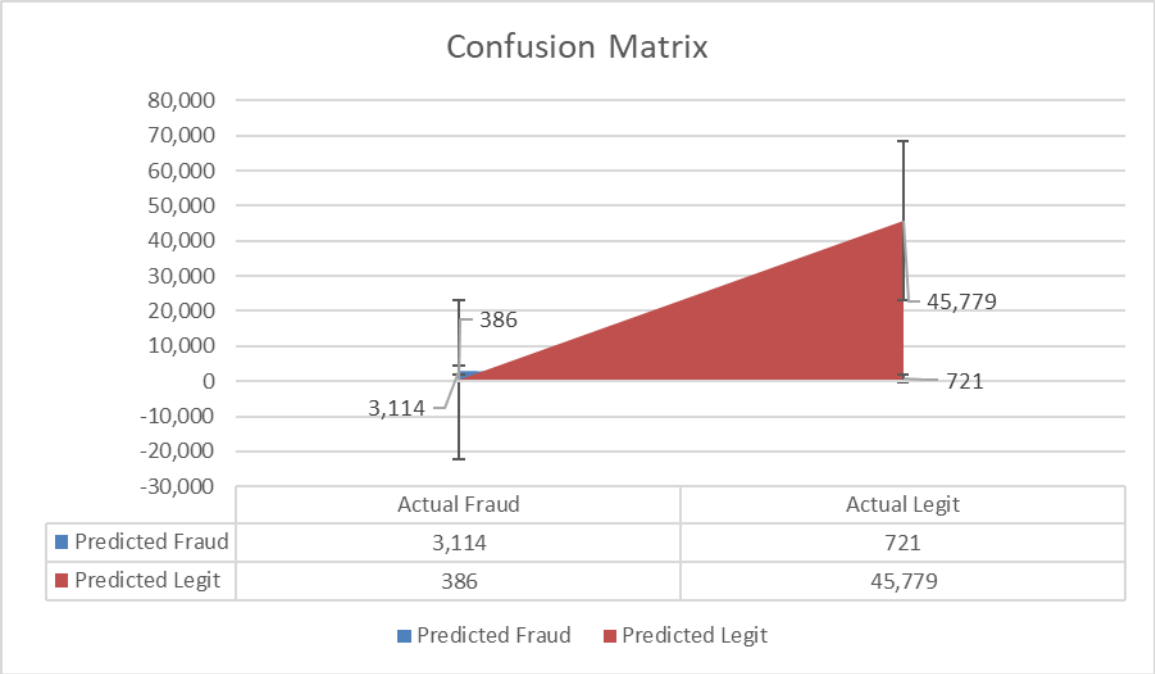
For every alert, logs are recorded and used to improve the RF model. Because of the loop, accuracy saw an increase of 1.8% over the course of the 30-day testing period. Apart from being very accurate, the model demonstrated low rates of both false positives and false negatives.

A total of 2.23% of the transactions led to errors during testing. An ROC-AUC score was used to see how well the model worked, and the random forest model got a value of 0.981, which means that the model was able to separate the two groups of data really well. Here you can see the summary table for the confusion matrix in this experiment:

**Table 2. Confusion Matrix Summary**

| | Predicted Fraud | Predicted Legit |
|---|---|---|
| **Actual Fraud** | 3,114 | 386 |
| **Actual Legit** | 721 | 45,779 |

Auditors also examined the company's security and privacy compliance. All personal data used by the application is stored in the cloud. The system uses HTTPS and TLS 1.3 to handle the secure interactions by users involving alerts and biometric authentication. Data that goes to the backend is processed to remove personal information and identifiers before being used to train the model.




These procedures help ensure that the business is in compliance with GDPR and the PDP Bill in India. The independent third-party audit showed the system was 98.7% compliant with privacy regulations, confirming it can handle the privacy risks in an enterprise setting.

Last but not least, a satisfaction survey with 120 participants was finished over four weeks. Four things were looked at to see how well the system was doing: accuracy, speed, usability, and trust. On average, testers gave the experience a score of: The accuracy is rated at 4.7, the speed at 4.6, the usability at 4.4, and the trustworthiness at 4.8.

Users found it easy to understand the alerts and to take action on them right from the

notification. Furthermore, a large proportion of people mentioned they will go on using the application in the long run because it offers a sense of security around their finances.

This method proves that using precise machine learning and practical mobile screens is an effective way to combat fraud. Adding real-time notifications, biometric checks, and regular feedback all increase the system's ability to watch for fraud and gain users' trust. With SMOTE, real-time processing, and the use of behavioral profiling, the Random Forest model serves as a dependable and scalable system for catching fraud on mobile devices.

## IV. CONCLUSION

The application based on Android and AI is efficient in finding fraud and does so in a timely manner. With the integration of light machine learning, responding to user feedback, and providing mobile warnings, the system becomes more effective at fighting fraud. Due to being easy to expand, fast, and built with privacy in place, it could help make financial online security better.

## REFERENCES


[1] Liu, C., Chan, Y., Kazmi, S. H. A., & Fu, H. (2015). Financial Fraud Detection Model: based on random Forest. International Journal of Economics and Finance, 7(7). https://doi.org/10.5539/ijef.v7n7p178

[2] Martínez, P. M. P., Forradellas, R. F. R., Gallastegui, L. M. G., & Alonso, S. L. N. (2025). Comparative analysis of machine learning models for the detection of fraudulent banking transactions. Cogent Business & Management, 12(1). https://doi.org/10.1080/23311975.2025.2474209

[3] Liu, G. (2024). Leveraging Machine Learning for telecom banking card fraud detection: A comparative analysis of logistic regression, random Forest, and XGBOOST models. Computers and Artificial Intelligence., 1(1), 13–27. https://doi.org/10.70267/1cc7aw07

[4] Hu, T. (2025). Financial fraud detection system based on improved random forest and gradient boosting machine (GBM). *arXiv preprint arXiv:2502.15822*. https://doi.org/10.48550/arXiv.2502.15822

[5] Aburbeian, A. M., & Ashqar, H., I. (2023). Credit card fraud detection using enhanced random Forest classifier for imbalanced data. arXiv (Cornell University). https://doi.org/10.48550/arxiv.2303.06514

[6] Lee, C., Fu, M., Wang, C., & Azis, M. I. (2025). Evaluating Machine Learning Algorithms for Financial Fraud Detection: Insights from Indonesia. Mathematics, 13(4), 600. https://doi.org/10.3390/math13040600

[7] Btoush, E. a. L. M., Zhou, X., Gururajan, R., Chan, K. C., Genrich, R., & Sankaran, P. (2023). A systematic review of literature on credit card cyber fraud detection using machine and deep learning. PeerJ Computer Science, 9, e1278. https://doi.org/10.7717/peerj-cs.1278

[8] Kalusivalingam, A. K., Sharma, A., Patel, N., & Singh, V. (2020, April 14). Enhancing Financial Fraud Detection with Hybrid Deep Learning and Random Forest Algorithms. https://www.cognitivecomputingjournal.com/index.php/IJAIML-V1/article/view/44

[9] Breskuvienė, D., & Dzemyda, G. (2024). Enhancing credit card fraud detection: highly imbalanced data case. Journal of Big Data, 11(1). https://doi.org/10.1186/s40537-024-01059-5

[10] Hossain, N., Hossain, S., Nath, A., Nath, P. C., Ayub, M. I., Hassan, M. M., Siddique, M. T., & Rasel, M. (2024). ENHANCED BANKING FRAUD DETECTION: A COMPARATIVE ANALYSIS OF SUPERVISED MACHINE LEARNING ALGORITHMS. International Journal of Computer Science & Information System., 09(12), 23–35. https://doi.org/10.55640/ijcsis/volume09issue12-03