International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 10 | Oct - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

Sentinel AI: An AI-Powered UPI Fraud Detection System

Mohamed Rizwan TH[1], Mohamed Salman P[1], Prateeka N[1], Priyadharshini V[1], R. RajiniGanth [2]

[1]Department of Computer Science and Engineering, SNS College of Engineering, Coimbatore, India

[2]Assistant Professor, CSE Department, SNS College of Engineering, Coimbatore, India

---

Abstract - Fraud in Unified Payments Interface (UPI) transactions is rare, often small in value, and moves quickly through mule accounts, while fraud labels arrive late and false alarms disturb honest customers. Sentinel AI is a research prototype that scores each UPI-style payment before it completes and gives human reviewers clear evidence. A feature engine compares every payment with the sender's own history, using amount deviation, velocity, splitting indicators, sequence timing, device and location context, and receiver-side network statistics, all computed only from information available at the moment of payment. An XGBoost model estimates a risk score, and a rules engine and threshold engine convert it into one of three outcomes: allow, step-up confirmation, or hold for review. Each held payment carries short plain-language reasons, and analysts can approve, confirm fraud, or escalate. An admin console supports threshold and rule changes, feature-group switches, an account-relationship graph, metrics, drift monitoring, and an append-only audit log. The prototype uses a FastAPI backend, a PostgreSQL database, and a simulated UPI app with synthetic data, no real money, and no bank connection. Observations reported here describe prototype behavior only; validation on real data and a literature and patent review are planned before any claim of real-world effectiveness or novelty. A risk score is a signal for investigation, not proof of fraud.

Key Words: UPI fraud detection, explainable machine learning, XGBoost, behavior profiling, mule accounts, human-in-the-loop review.

I. INTRODUCTION

India's digital payment ecosystem has grown rapidly, and the Unified Payments Interface (UPI) now sits at its centre, handling about 24.5 billion transactions worth nearly ₹29.8 lakh crore in a single month (August 2026). Instant, irreversible settlement makes payments convenient, but it also leaves very little time to stop a fraudulent transfer. Government figures place reported UPI fraud at 12.64 lakh incidents worth ₹981 crore in 2024-25 [18]. Banks, payment providers, and customers must therefore decide on each payment within moments and with limited context.

Fraud is also hard to find. Roughly one fraudulent payment appears among every 147,000 genuine ones, and the average fraud is only a few thousand rupees. Simple amount limits and static rules miss small-value fraud, rapid bursts of payments, amounts split below a threshold, account takeovers on new devices, and mule accounts that pass money along within seconds. Overly strict systems cause the opposite problem by blocking honest customers. Reviewers also need to know why a payment was flagged, and operators need to adjust settings without rebuilding the system.

Sentinel AI is proposed as an explainable, real-time fraud-risk platform for UPI-style payments that scores each transaction before it completes. Unlike methods that judge every payment in isolation, it compares a payment with the sender's own history, recent activity, and the receiver's relationships. Every feature uses only information available at the moment of payment, so the score never depends on future data. An XGBoost model produces a risk score, and a rules engine and threshold engine convert it into one of three outcomes: allow, step-up confirmation, or hold for review. Each held payment carries short plain-language reasons so that an analyst can act on evidence. A risk score is a signal for investigation, not proof of fraud.

To demonstrate the framework, the prototype includes a simulated UPI app, whose payments trigger live requests to the server, and an admin console in which analysts approve, confirm fraud, or escalate held payments and administrators manage thresholds, rules, and monitoring. The prototype uses no real money and has no connection to any bank or payment network. Analyst decisions become labels for retraining, while unreviewed payments are treated as unknown rather than legitimate, which limits the bias that fraud-detection systems inherit from their own alerts. Every configuration change, rule edit, and alert decision is recorded in an append-only audit log.

This paper makes three contributions. First, it combines behavior profiling, sequence analysis, device and location context, and account-network analysis in one leakage-free, pre-completion scoring pipeline. Second, it couples explainable scoring with administrator control, audit logging, and drift monitoring in a single workflow from the customer's payment to the analyst's decision. Third, it defines an evaluation design in which each feature group is tested against a baseline on a time-based test set, so its value is measured and not assumed. The results come from synthetic data and reflect prototype behavior only; validation on real data and a full literature and patent review are needed before any claim of real-world effectiveness or novelty.

II. LITERATURE REVIEW

Fraud detection in payments is a well-studied machine learning problem with three well-documented difficulties: customer behavior and fraud tactics change over time (concept drift), genuine transactions heavily outnumber frauds (class imbalance), and fraud labels arrive late (verification latency) [1]. These are also the conditions of UPI, where government figures put fraud at roughly one in 147,000 payments. Sentinel AI is designed around them through time-based evaluation, precision-recall metrics, and drift monitoring.

A second issue is where labels come from. Recent labels come mostly from investigators checking alerts, which creates sample selection bias [1]. Earlier work proposed handling investigator feedback and delayed labels separately, using two classifiers whose outputs are aggregated [2]. Sentinel AI follows this lesson in a simple form: labels are saved only from reviewed alerts, together with the time the label became known, and unreviewed payments are treated as unknown rather than legitimate.

Customer behavior profiling is the most established idea in the field. Whitrow et al. compared judging each transaction alone with judging it against aggregated history and found that aggregation helps in many but not all cases, that the length of the aggregation window strongly affects results, and that aggregation may resist population drift better [3]. Weston et al. used peer-group analysis to flag transactions that deviate strongly from their peer group [4]. Bahnsen et al. showed that transaction time of day should be treated as a periodic variable and modeled with the von Mises distribution [5]. Sentinel AI applies these ideas through amount-against-usual-spending, rolling-window, and hour-of-day features, and treats the window lengths as settings to test.

Velocity checks and splitting detection are common in industry practice, and patents describe real-time engines built on behavior profiles and velocity counters (for example, US 8458069 and US 11875355) [6]. No verified academic study that evaluates transaction splitting on its own was identified. Sentinel AI therefore treats splitting as a supporting signal that is measured in experiments, never as proof of fraud.

The order and timing of transactions carry further information. Jurgovsky et al. found that an LSTM sequence model improved detection on card-present transactions over a static model, that both benefited from manual aggregates, and that the two caught different frauds [7]; the accompanying thesis adds that LSTM modeling suits card-present sequences but not card-not-present ones and recommends simple, transparent methods [8]. Lucas et al. report that HMM-based sequence features raised precision-recall AUC by 15.1% over state-of-the-art feature engineering [9]. Nguyen et al. distinguish real-time detection from posterior detection, in which later transactions can be used to block a card retrospectively [10]. Because Sentinel AI scores before a payment completes, it uses transparent sequence features such as time gaps and amount trends computed from past data only.

Network analysis helps find coordinated fraud. Van Vlasselaer et al. combined customer spending history with network features from the graph of cardholders and merchants in APATE and report that the combination produced the best models, with AUC above 0.98 [11]. A recent review of graph neural networks notes that transaction networks help detect complex fraud such as money laundering and card fraud [12]. Mule accounts are a stated concern in India, where Home Ministry material reports more than 24 lakh suspect mule accounts flagged [18]. Sentinel AI uses interpretable graph statistics, namely receiver in-degree, distinct senders, and links to confirmed-fraud accounts, rather than a graph neural network, so that analysts can see why a payment looks suspicious. Device and location signals add context: external information about countries and calendar events improved classification most noticeably on card-not-present transactions [8]. No verified public UPI dataset with real device or IP signals was identified, so Sentinel AI simulates these signals and marks them as demonstration only.

Chen and Guestrin introduced XGBoost, a scalable tree-boosting system widely used to achieve state-of-the-art results on machine learning challenges [13]. Grinsztajn et al. found that tree-based models remain state-of-the-art on medium-sized tabular data of about 10,000 samples [14]. Fraud datasets are much larger, so this supports the choice without proving it; Sentinel AI therefore compares XGBoost against a logistic-regression baseline. For explanation, SHAP assigns each feature an importance value for a particular prediction [15]. Sentinel AI uses the equivalent per-feature contributions from XGBoost to create short reasons for each alert, written as signals and never as proof, because an explanation describes the model and not the cause of fraud.

Public data is limited. PaySim was created to simulate mobile money transactions because sensitive financial data is hard to share [16]. Its Kaggle version represents fraud as taking over accounts, emptying funds by transfer, and cashing out, and its documentation warns that transactions detected as fraud are cancelled, so the four balance columns must not be used for fraud detection. The Bank Account Fraud suite offers privacy-preserving tabular datasets from real bank account-opening fraud with temporal dynamics, class imbalance, and built-in bias; it suits drift testing but covers account opening, not payments [17]. Results on synthetic data therefore show prototype behavior only.

UPI fraud control in India already relies on several layers, described here using only what government replies and published reports state publicly; the internal systems of individual banks are not public. UPI itself provides device binding, two-factor authentication through a PIN, daily transaction limits, and curbs on use cases that fraudsters abuse. These controls confirm who is paying and from which device but do not judge whether the payment looks unusual for that customer; a fraudster who obtains the PIN or tricks the customer into approving a payment can still pass them [18]. NPCI provides banks a free fraud-monitoring solution with rule-development capabilities to alert on and decline transactions, and uses AI/ML-based models to give banks a risk score based on the history of money flows; NPCI usually alerts the banks, and the final action sits with each bank [18]. For mule accounts, the Reserve Bank Innovation Hub built MuleHunter.AI, which scores accounts from transaction data, customer profiles, and ecosystem learning and has been adopted by 29 banks, while a Digital Payments Intelligence Platform pilot with seven banks shares data to train models that flag anomalous transactions [19].

The remaining layers act after the money has moved. Victims report through the National Cybercrime Reporting Portal and the helpline 1930, and RBI runs the Central Payments Fraud Information Registry for regulated entities. Home Ministry figures show about ₹8,189 crore saved against more than ₹55,050 crore reported from 2021 to 2025, roughly 15%, although the two figures are defined differently and the ratio is approximate [18]. RBI circulars of 2017 limit customer liability in unauthorized electronic transactions [20], and awareness campaigns educate users, but neither stops an individual fraudulent payment. Student and journal papers also apply Random Forest, AdaBoost, CNN, and LSTM models to UPI fraud, but their data sources and evaluation methods could not be verified, so they are treated as indicative only.

Several limitations of the existing system follow. Static checks verify the user and cap exposure but do not judge whether a payment is unusual for its sender. Hand-written rules need constant upkeep and can miss small-value fraud, rapid bursts, and split payments. Public descriptions mention risk scores and alerts but not plain-language reasons that an analyst can inspect. Each bank sees only its own customers, cross-bank sharing is at pilot stage, and mule-account scoring is not yet universal. Labels arrive late and inherit the blind spots of earlier alerts [1]; recovery is reactive; and public descriptions do not show how much each type of signal adds, nor do they describe drift monitoring or administrator control.

The studies above treat behavior, sequence, network, and context features mostly separately, and on card data. No verified public benchmark was found that combines them for UPI-style transfers while also covering analyst review, labels, administrator control, and drift monitoring. Each selected feature already exists in prior research or patents, so their combination is not automatically novel. Sentinel AI's intended contribution is a reproducible, explainable prototype that measures what each feature group adds and complements the existing layers rather than replacing them; real-world effectiveness and novelty remain unproven until it is tested on real data and checked against a full literature and patent search.

III. METHODOLOGY

A. Study Scope and Design Approach

The work is a design-oriented case study of a research prototype. The platform builds on the layers already in place (authentication, NPCI monitoring, mule detection, and complaint reporting) and adds per-payment scoring, plain-language reasons, administrator control, and measured feature value in one transparent and auditable workflow. The prototype is a single FastAPI backend with a PostgreSQL database and lightweight web front ends, including a simulated UPI app. It uses synthetic data, no real money, and no bank connection. The primary actors are customers, who make payments; analysts, who investigate alerts; and administrators, who control policy. Each role has role-based access.

B. Functional Model and Transaction Data

The functional model combines four groups of features, each computed only from data available before the current payment, so no future information leaks into the score. Behavior, velocity, and sequence features compare the payment with the sender's own past: how unusual the amount is, whether the payee is new, how many payments occurred in the last five minutes, hour, and day, whether repeated small payments approach a threshold, and how the timing and order of recent payments have changed. This targets small-value fraud, rapid bursts, and split payments that fixed limits miss.

Network features examine the receiver: how many distinct senders paid it recently, how much it received and forwarded, whether it is linked to accounts with confirmed fraud, and whether other senders paid it the same amount in a short window. These simple, interpretable graph statistics help expose mule accounts and fraud rings, and an admin graph view shows the account's neighbors. Only confirmed labels already known at that moment are used. Device and location features check whether the payment comes from a new device, a new IP address, or a changed city, and whether a location change is too fast to be plausible; in this prototype these signals are simulated and labeled "demonstration only."

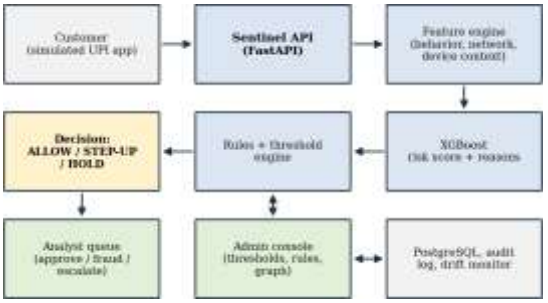


Fig. 1. Proposed system overview of Sentinel AI.

C. Three-Outcome Payment Scoring Workflow

Every payment request goes to the Sentinel API before any money moves. The server validates the request, builds features from stored history, estimates a risk score with an XGBoost model, and returns one of three outcomes:

- **ALLOW:** the payment completes.
- **STEP-UP:** the user confirms with an extra "Is this you?" step.
- **HOLD:** the payment waits for an analyst.

The server owns the ledger, and the app never reports its own risk. This turns fraud control from a mostly after-the-fact process into a decision made at the moment of payment. Each held payment appears in an analyst queue with its risk score and its top reasons, written as signals, such as an amount far above this sender's usual, a first payment to this payee, a device not seen before, or a receiver linked to accounts with confirmed fraud. The analyst can approve, confirm fraud, or escalate. Customers never see scores or reasons.

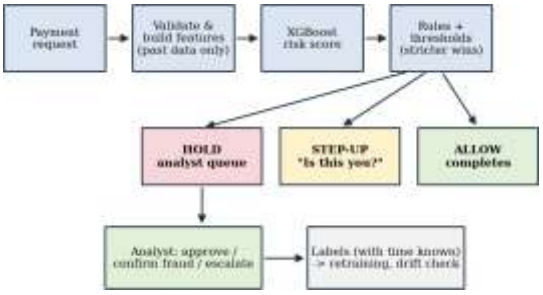


Fig. 2. Proposed payment scoring workflow and three outcomes.

D. System Architecture and Data Flow

The architecture separates the presentation layer, application layer, machine-learning layer, and data layer. The presentation layer provides the simulated UPI app, the analyst queue, and the admin console. The application layer validates requests, checks roles, owns the ledger, and applies the rules engine and threshold engine to the model's score. When a rule and the model disagree, the stricter outcome wins, so a rule can only make a decision safer. Authorized administrators can adjust the two risk thresholds, add or edit rules, and switch feature groups on and off without retraining the model.

The machine-learning layer holds the feature engine, the XGBoost model, and the per-feature contributions used to write alert reasons. The data layer is a PostgreSQL database that stores transactions, analyst outcomes, configuration, and an append-only audit log. Analyst outcomes become labels for retraining, saved with the time they became known; unreviewed payments are treated as unknown and not as legitimate, which limits the selection bias that alert-based labels create. A drift monitor compares recent data with the training data using population stability and statistical tests and tracks performance on reviewed alerts. Defined review thresholds trigger an investigation and, if needed, retraining.

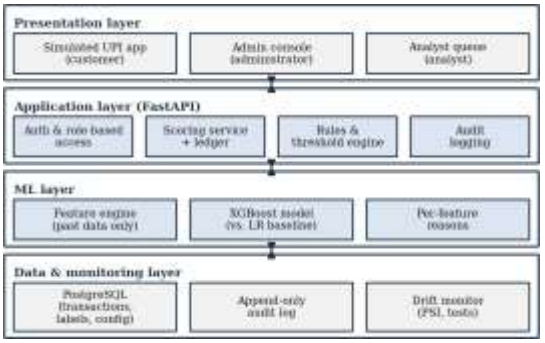


Fig. 3. Proposed system architecture and data flow.

E. Security, Privacy, and Responsible AI

Access is role-based, passwords are hashed, UPI IDs are masked in analyst views, and the prototype uses synthetic data only, with no real money and no connection to any bank or payment network. Every configuration change, rule edit, and alert decision is written to the append-only audit log. Scores and reasons are visible only to analysts and administrators. A risk score is evidence for investigation, not proof of fraud, and the reasons are written as signals because an explanation describes the model and not the cause of fraud. Treating unreviewed payments as unknown reduces the bias that systems inherit from their own alerts, and the simulated device and location signals must be validated on real labeled data before any claim of effectiveness.

IV. RESULTS

A. Prototype and Design Outcomes

The prototype demonstrates one workflow, from the moment a customer taps Pay to an analyst's decision. The simulated UPI app (Fig. 4) shows a demo account with one-tap demo payees. The payment screen (Fig. 5) lets the tester enter a payee and amount and optionally switch on demo context signals to simulate an attack. Each payment triggers a live request to the Sentinel server, which returns allow, step-up, or hold.



Fig. 4. Simulated UPI demo app (customer home screen).



Fig. 5. Payment trigger screen of the demo app.

Staff reach the Sentinel AI Console through role-based login for administrators and analysts (Fig. 6). The real-time alert queue (Fig. 7) lists held payments with sender and receiver, amount, risk score, top model-derived reasons, status, and the analyst actions approve, fraud, and escalate. The transaction search and forensics view (Fig. 8) lists transactions with their decision (for example HOLD or STEP_UP) and status to support later review. Other console sections include the account graph, thresholds and toggles, business rules, metrics and analytics, a data-drift report, and the audit trail.



Fig. 6. Sentinel AI Console login for administrators and analysts.



Fig. 7. Real-time alert queue showing held payments, risk scores, and reasons.



Fig. 8. Transaction search and forensics view.

These screens are demonstration outputs on synthetic data. The risk scores and amounts shown are illustrative placeholders and are not evidence of model accuracy, and the device and location signals are simulated. At the workflow level, the reported outcomes are the three-outcome decision (allow, step-up, hold), the analyst review loop with audit logging, and administrator control over thresholds, rules, and feature groups. The evaluation metrics named in Section V-B are not reported here as observed results.

V. DISCUSSION

A. Design Implications

The principal design benefit is that fraud control moves from a mostly reactive process to a decision made at the moment of payment. Because each payment is scored before it completes, a suspicious transfer can be confirmed or held while there is still time to act. Comparing each payment with the sender's own history of amounts, timing, payees, and devices is intended to flag unusual activity that a single fixed limit would miss, while velocity and splitting indicators target rapid bursts and repeated small payments built to stay under a threshold. Account-network features show when many senders pay one receiver, when money is passed on quickly, or when an account is linked to confirmed fraud.

The three outcomes are intended to give a balanced customer experience: most honest users pay without friction, while only riskier payments receive extra checks or review. Every held payment carries short plain-language reasons, so analysts can act on evidence and not on a bare score. Administrators can change thresholds, rules, and feature groups without retraining the model, every change and alert decision is recorded in an audit log, and role-based access, hashed passwords, masked identifiers, and synthetic data support privacy. Analyst outcomes become labels for retraining, and drift monitoring allows the system to be reviewed as fraud tactics evolve. These benefits are design goals; their magnitude has not yet been measured in the available results.

B. Limitations and Validation Plan

The main limitation is that all results come from synthetic data and show prototype behavior only. The device and location signals are simulated, and synthetic data can carry artifacts of the generator that

a model may learn, so features should be audited for them. Explanations describe the model and not the cause of fraud. The prototype is not connected to any bank or payment network, and its effectiveness on real payments is unknown.

The planned evaluation adds each feature group one at a time and compares it with a baseline on the same time-based test set, using precision, recall, PR-AUC, false-positive rate, and latency; XGBoost is also compared with a logistic-regression baseline, and negative results are reported too, so the contribution of each signal is measured and not assumed. Further validation should use real, authorized data, independent security and fairness testing, including monitoring of performance across customer groups, and a full literature and patent search before any claim of novelty. Until these checks are completed, Sentinel AI should be described as a research prototype and a complement to existing fraud-control layers, not a replacement for them.

VI. CONCLUSION AND FUTURE WORK

Existing UPI controls, such as PIN and device checks, rule-based monitoring, mule-account detection, and complaint-based reporting, have made the network safer. Static checks, fragmented views, and after-the-fact recovery still leave gaps in judging each payment as it happens, explaining decisions to reviewers, and adapting to new fraud tactics. Sentinel AI addresses these gaps with an explainable, behavior- and network-aware system that scores each payment before it completes. An XGBoost model estimates the risk, and a rules and threshold engine turns it into one of three outcomes, so honest customers face little friction while riskier payments receive extra checks.

The key contribution is the integration of explainable scoring with administrator control and continuous monitoring. Every feature uses only information available at the moment of payment, each feature group is compared with a baseline on a time-based test set, administrators can change policy without retraining, analyst outcomes become labels for future models, and drift monitoring tracks changes in data and performance. The prototype shows how a simulated UPI app, a scoring backend, and an admin console work together as one workflow. Validation on real, authorized data, a full literature and patent review, and independent testing are needed before any claim of real-world effectiveness or novelty.

Future work includes integration with real-time payment processing, banking systems, and fraud-investigation platforms through secure APIs; stronger explainability that highlights the transaction features behind each score; graph analytics, including graph neural networks, for mule accounts and coordinated fraud; adaptive learning and concept-drift monitoring; multimodal risk analysis combining transaction history, device, geography, and account relationships; advanced anomaly detection for unseen patterns; a dedicated investigation dashboard with analyst feedback; privacy-preserving collaboration through federated learning; optimization for high-volume transaction streams; and continuous evaluation with precision, recall, F1-score, false-positive rate, and latency, including monitoring across customer groups.

ACKNOWLEDGMENT

The authors acknowledge the Department of Computer Science and Engineering, SNS College of Engineering, Coimbatore, for supporting the academic project. The authors also thank the project guide for guidance during the design and documentation process.

REFERENCES

[1] A. Dal Pozzolo, G. Boracchi, O. Caelen, C. Alippi, and G. Bontempi, “Credit card fraud detection: A realistic modeling and a novel learning strategy,” IEEE Trans. Neural Netw. Learn. Syst., vol. 29, no. 8, pp. 3784–3797, 2018.

[2] A. Dal Pozzolo, G. Boracchi, O. Caelen, C. Alippi, and G. Bontempi, “Credit card fraud detection and concept-drift adaptation with delayed supervised information,” in Proc. Int. Joint Conf. Neural Networks (IJCNN), 2015.

[3] C. Whitrow, D. J. Hand, P. Juszczak, D. Weston, and N. M. Adams, “Transaction aggregation as a strategy for credit card fraud detection,” Data Min. Knowl. Discov., vol. 18, no. 1, pp. 30–55, 2009.

[4] D. J. Weston, D. J. Hand, N. M. Adams, C. Whitrow, and P. Juszczak, “Plastic card fraud detection using peer group analysis,” Adv. Data Anal. Classif., vol. 2, no. 1, pp. 45–62, 2008.

[5] A. Correa Bahnsen, D. Aouada, A. Stojanovic, and B. Ottersten, “Feature engineering strategies for credit card fraud detection,” Expert Syst. Appl., vol. 51, pp. 134–142, 2016.

[6] U.S. Patent 8,458,069 and U.S. Patent 11,875,355, real-time fraud detection systems based on behavior profiles and velocity counters.

[7] J. Jurgovsky et al., “Sequence classification for credit-card fraud detection,” Expert Syst. Appl., 2018.

[8] J. Jurgovsky, Context-Aware Credit Card Fraud Detection, Ph.D. dissertation, Univ. of Passau / INSA Lyon, 2019.

[9] Y. Lucas et al., “Multiple perspectives HMM-based feature engineering for credit card fraud detection,” in Proc. ACM Symp. Applied Computing (SAC), 2019, arXiv:1905.06247.

[10] V. B. Nguyen, K. G. Dastidar, M. Granitzer, and W. Siblini, “The importance of future information in credit card fraud detection,” in Proc. 25th Int. Conf. Artificial Intelligence and Statistics (AISTATS), vol. 151, pp. 10067–10077, 2022.

[11] V. Van Vlasselaer, C. Bravo, O. Caelen, T. Eliassi-Rad, L. Akoglu, M. Snoeck, and B. Baesens, “APATE: A novel approach for automated credit card transaction fraud detection using network-based extensions,” Decis. Support Syst., vol. 75, pp. 38–48, 2015.

[12] D. Cheng, Y. Zou, S. Xiang, and C. Jiang, “Graph neural networks for financial fraud detection: A review,” Frontiers of Computer Science, vol. 19, no. 9, Art. no. 199609, 2025.

[13] T. Chen and C. Guestrin, “XGBoost: A scalable tree boosting system,” in Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, 2016, pp. 785–794.

[14] L. Grinsztajn, E. Oyallon, and G. Varoquaux, “Why do tree-based models still outperform deep learning on typical tabular data?” in Adv. Neural Inf. Process. Syst. (NeurIPS), vol. 35, 2022.

[15] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in Adv. Neural Inf. Process. Syst. (NeurIPS), vol. 30, 2017.

[16] E. A. Lopez-Rojas, A. Elmir, and S. Axelsson, "PaySim: A financial mobile money simulator for fraud detection," in Proc. 28th European Modeling and Simulation Symp. (EMSS), 2016.

[17] S. Jesus et al., "Turning the tables: Biased, imbalanced, dynamic tabular datasets for ML evaluation," in Adv. Neural Inf. Process. Syst. (NeurIPS) Datasets and Benchmarks Track, vol. 35, 2022.

[18] Ministry of Finance, Government of India, "Measures to strengthen cybersecurity in the financial sector," Press Information Bureau, Government of India, 2025.

[19] Reserve Bank Innovation Hub, "MuleHunter.AI: AI-powered mule account detection system," and "Digital Payments Intelligence Platform," RBIH, India.

[20] Reserve Bank of India, "Customer protection – Limiting liability of customers in unauthorised electronic banking transactions," circular, July 2017.