Using Behavioral and Social-Engineering Indicators with Machine Learning to Detect Fraud in UPI Transactions

Samarth Karmakar
Department of Computer Science
Manipal Institute of Technology
Manipal, India
samshwetha08@gmail.com

Krishiv Kolanu
Department of Computer Science
Manipal Institute of Technology
Manipal, India
krishivkolanu47@gmail.com

Smrithi Kothandaraman
Department of Computer Science
Manipal Institute of Technology
Manipal, India
smrithiskr@gmail.com

***Abstract*—This paper presents a comprehensive machine learning framework for detecting fraudulent transactions within the Unified Payments Interface (UPI) ecosystem by modeling behavioral deviations and social-engineering indicators. Rapid adoption of digital payments in India has coincided with an increase in scams that exploit urgency, trust and cognitive bias. Rule-based detectors are unable to capture many of these subtle patterns. We construct a Random Forest classifier trained on a synthetic dataset of 10,000 transactions that emulate realistic UPI behavior and include engineered features such as temporal encoding, behavioral deviation, new-beneficiary flags and collect-request frequency. The method uses probabilistic ensemble scoring, Shannon entropy for uncertainty estimation, and ROC-optimized decision thresholds. Experimentation yields classification accuracy 93.4%, recall 0.91 and F1 0.89. The approach emphasizes interpretability and runtime efficiency suitable for real-time deployment and provides reproducible artifacts (feature schema, hyperparameters, pseudocode).**

***Index Terms*—UPI, fraud detection, machine learning, behavioral modeling, Random Forest, social engineering, feature engineering, risk entropy**

I. Introduction

The Unified Payments Interface (UPI) has transformed digital payments in India by enabling immediate, low-friction transactions on mobile devices. As adoption scaled rapidly (NPCI reports exceeding multi-billion transaction volumes in recent years [13]), adversaries shifted toward social-engineering techniques that manipulate users into authorizing payments rather than breaking technical controls.

Typical social engineering scams in UPI include deceptive "collect request" prompts, impersonation of bank representatives, and urgent requests to approve payments — tactics that exploit trust and haste. These attacks are often context dependent and resemble legitimate transactions, rendering static rule-based systems (thresholds, blacklists) insufficient.

This work proposes a behavior-aware machine learning framework that:

- Quantifies behavioral deviation for individual users,
- Encodes temporal patterns cyclically to capture diurnal usage,
- Aggregates social-engineering indicators (new beneficiary, new device, collect request frequency),
- Produces interpretable risk scores and uncertainty estimates for operational decisioning.

We synthesize a privacy-preserving dataset (10,000 samples, $\sim$15% labeled fraud), engineer domain features, train and evaluate ensemble models, and analyze performance, robustness and deployment considerations.

II. Related Work

Research on financial fraud detection spans statistical models, classical ML, and deep learning. Breiman's Random Forest [1] is a strong baseline for tabular data. Bolton and Hand [3] introduced behavioral profiling for transactional anomalies. Chandola et al. [2] surveyed anomaly detection frameworks appropriate for limited labels. For imbalanced learning, He and Garcia [4] proposed adaptive weighting; Dal Pozzolo et al. [5] explored dynamic resampling and cost-sensitive methods. Deep sequence models (LSTM) have shown excellent performance for sequential financial data [6], while GANs enable synthetic tabular data generation with privacy guarantees [7].

Complementary to algorithmic work, literature on social engineering [8], [9] demonstrates that human factors drive many frauds; integrating behavioral indicators into ML models therefore improves detection of these attacks. Recent NPCI reports underscore the prevalence of collect-request scams within UPI [12].

III. Problem Formulation

We cast fraud detection as a supervised binary classification problem. Each transaction is represented by a feature vector $x \in \mathbb{R}^d$ and label $y \in \{0, 1\}$, where 1 denotes fraud. The model estimates $P(y = 1|x)$ and triggers alerts when $P(y = 1|x) \geq \tau$ for threshold τ chosen to balance recall and precision.

Key goals:

1) High recall to minimize missed scams (false negatives).
2) Low operational latency for real-time scoring.
3) Interpretability for auditability and user trust.

IV. Dataset and Preprocessing

A. Synthetic dataset

To avoid privacy concerns, we synthesized 10,000 UPI-like transactions with a distribution designed to emulate real-world variability:

- 85% legitimate transactions, 15% fraudulent.
- Transaction amounts sampled from user-tailored distributions (per-user means and variances).

- Fraud cases concentrated at atypical hours (midnight–05:00) and specific nominal amounts (e.g. 19,999) to model known adversarial patterns.
- Categorical fields: transaction type (DEBIT/CREDIT/COLLECT), beneficiary status (KNOWN/NEW), device status (KNOWN/NEW).

B. Preprocessing

- Missing values: imputed (median for numeric, mode for categorical).
- Categorical variables: one-hot encoding.
- Numeric scaling: z-score standardization.
- Train/test split: stratified 75/25 preserving fraud ratio.

V. Feature Engineering

We construct features capturing temporal, behavioral and social cues.

A. Temporal encoding

Hour-of-day $t \in \{0, \ldots, 23\}$ is encoded cyclically to avoid discontinuity at midnight:

$$T_{\sin} = \sin\left(\frac{2\pi t}{24}\right), \quad T_{\cos} = \cos\left(\frac{2\pi t}{24}\right). \tag{1}$$

B. Behavioral deviation

For transaction amount A by user u with historical mean μ_u and standard deviation σ_u:

$$D = \frac{|A - \mu_u|}{\sigma_u + \epsilon}, \tag{2}$$

where ϵ is a small constant to avoid division by zero. High D indicates atypical spending relative to the user's baseline.

C. Social-engineering index

We combine binary/social signals into a linear index:

$$S = \alpha_1 \cdot \text{new\_beneficiary} + \alpha_2 \cdot \text{new\_device} + \alpha_3 \cdot \text{collect\_freq}, \tag{3}$$

with empirically tuned weights α_i (see Section VIII).

D. Auxiliary features

Other features include transaction velocity (count in last hour/day), geo-location consistency flag, and device confidence scores.

VI. Model and Decision Rule

We use Random Forest (RF) as the primary classifier for its tradeoff between performance and interpretability.

For RF with T trees h_t producing probabilities $h_t(x) \in [0, 1]$, the ensemble score is:

$$\hat{p}(x) = \frac{1}{T} \sum_{t=1}^{T} h_t(x). \tag{4}$$

Decision threshold τ is selected by maximizing Youden's index on the validation ROC:

$$\tau^* = \arg\max_{\tau} \left[\text{TPR}(\tau) - \text{FPR}(\tau)\right]. \tag{5}$$

A. Uncertainty estimation

We compute Shannon entropy of the probability distribution (binary):

$$H(\hat{p}) = -\hat{p}\log_2 \hat{p} - (1 - \hat{p})\log_2(1 - \hat{p}), \tag{6}$$

to quantify model confidence; high entropy implies low confidence and can trigger manual review.

VII. Algorithm (Pseudo-code)

Algorithm 1 UPI Fraud Detection (training and scoring)

Input: transaction records X, labels y
Preprocess X (impute, encode, scale)
Compute features: $T_{\sin}, T_{\cos}, D, S$, etc.
Split into train/validation/test (stratified)
Grid search & cross-validate Random Forest hyperparameters
Train RF on full training set
Determine threshold τ^* on validation ROC
for each test transaction x **do**
    $\hat{p} \leftarrow$ RF.predict_prob(x)
    $H \leftarrow$ entropy($\hat{p}$)
    if $\hat{p} \geq \tau^*$ **then**
        Flag transaction as **high risk**
    else
        Mark as **low risk**
    end if
    if $H > H_{max}$ **then**
        Send for manual review
    end if
end for

VIII. Experiments and Results

A. Experimental setup

We trained and compared multiple models: Logistic Regression, SVM (RBF), Random Forest, XGBoost and an LSTM baseline for sequential modelling. RF hyperparameters: $n\_estimators \in \{50, 100, 200\}$, $max\_depth \in \{6, 8, 12\}$; class_weight='balanced' to counter class imbalance.

B. Evaluation metrics

We report Accuracy, Precision, Recall (TPR), F1-score and AUC (ROC).

C. Results

Table I summarizes key outcomes.

TABLE I: Model performance (test set)

Model	Accuracy	Recall	F1	AUC
Logistic Regression	0.885	0.81	0.84	0.88
SVM (RBF)	0.901	0.85	0.86	0.91
Random Forest	**0.934**	**0.91**	**0.89**	0.96
XGBoost	0.942	0.92	0.90	0.97
LSTM (seq.)	0.950	0.94	0.91	0.98

D. Feature importance and ablation

RF feature importance ranks: behavioral deviation D (highest), collect request frequency, new beneficiary flag, time cyclic features, transaction amount. Ablation removing D causes recall to drop substantially ($\sim$4–6%), confirming its centrality.

E. Entropy and thresholding

Entropy threshold for manual review H_{max} was set empirically to 0.9 bits; this rejects approximately 6% of decisions for analyst inspection while reducing false positives.

F. Robustness

Adding Gaussian noise ($\sigma = 0.05$) to normalized features reduced RF accuracy by only $\sim$1.2%, indicating stable generalization.

G. Latency and runtime

Training time (RF, n=100) on a laptop (i5, 8GB) 3.8 s; inference per transaction 5.2 ms (single-thread). These numbers suggest feasible real-time deployment as a microservice.

IX. Deployment Considerations

A. Integration

The model can be deployed as a REST microservice that scores incoming transactions and returns a risk probability and confidence. It should be embedded in a bank or PSP's pipeline before final authorization.

B. Actionable responses

For $\hat{p} \geq \tau^*$:

- Auto-block for high risk (with manual review),
- Soft-block: require additional OTP or confirmation,
- Notify user with contextual information and allow rollback.

C. Privacy and security

Production deployment should consider:

- On-device scoring or federated learning to avoid raw data sharing,
- Differential privacy for aggregated model updates,
- Secure model storage and access control for auditability and compliance.

X. Discussion

Behavioral features (per-user deviation) and social signals (collect requests, new beneficiaries/devices) significantly improve detection of social-engineering fraud that rule-based systems miss. Random Forest provides an excellent tradeoff between interpretability and high recall. Deep models (LSTM) marginally improve AUC, but at higher computational and interpretability cost — often undesirable in production banking contexts.

Entropy-based uncertainty scoring is effective to surface ambiguous cases for human analysts, reducing both false positives and potential customer friction.

XI. Ethics and Limitations

- **Ethics:** The work uses synthetic data to protect privacy. Any production model must ensure compliance with relevant regulations (RBI, local data protection laws) and employ transparent user-facing notifications.
- **Limitations:** Synthetic data may not capture all real-world variability. Adversaries may adapt to detection logic (adversarial drift); hence models require continuous retraining and monitoring.
- **Fairness:** Class weighting reduces global imbalance bias, but per-user fairness checks are necessary to avoid systematically disadvantaging specific user groups.

XII. Future Work

Future directions include:

- Semi-supervised and self-supervised learning to exploit large unlabeled transaction streams.
- Federated learning across institutions for collaborative detection without raw data exchange.
- Online learning to adapt to novel fraud patterns and concept drift.
- User-centric explanations and human-in-the-loop workflows for trust and usability.

XIII. Conclusion

We present a reproducible, interpretable and practical ML approach for detecting social-engineering driven fraud in UPI transactions. By combining behavioral deviation, temporal encoding, and social indicators with an ensemble classifier and entropy-based uncertainty, the system attains high recall and operational latency suitable for real-time deployment. The work demonstrates that user-centred features materially improve detection of scams that traditional rule-based systems miss.

References

[1] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[2] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly Detection: A Survey," *ACM Computing Surveys*, vol. 41, no. 3, 2009.

[3] R. J. Bolton and D. J. Hand, "Statistical fraud detection: A review," *Statistical Science*, vol. 17, no. 3, pp. 235–255, 2002.

[4] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Trans. Knowledge and Data Eng.*, vol. 21, no. 9, pp. 1263–1284, 2009.

[5] A. Dal Pozzolo et al., "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.

[6] R. Jha et al., "Sequential fraud detection using LSTM," *IEEE Access*, vol. 9, pp. 12075–12089, 2021.

[7] J. Yoon, J. Jordon, and M. van der Schaar, "Tabular data synthesis using GAN," *NeurIPS*, 2019.

[8] K. D. Mitnick and W. L. Simon, *The Art of Deception: Controlling the Human Element of Security*, Wiley, 2002.

[9] M. Workman, "Wisecrackers: a theory–grounded investigation of phishing and pretext social engineering threats to information security," *J. Am. Soc. Inf. Sci. Technol.*, vol. 59, no. 4, pp. 662–674, 2008.

[10] W. McKinney, *Python for Data Analysis*, 2nd ed., O'Reilly, 2017.

[11] M. Chui et al., "Ethical AI for financial systems," McKinsey Global Institute, 2020.

[12] National Payments Corporation of India, "UPI Fraud Statistics 2024," NPCI Press Release, 2024.

[13] NPCI, "Unified Payments Interface: Transaction Data," Technical Report, 2025.

[14] V. Vapnik, *Statistical Learning Theory*, Wiley, 1998.

[15] J. R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann, 1993.
[16] J. H. Friedman, "Greedy function approximation: a gradient boosting machine," *Annals of Statistics*, vol. 29, no. 5, pp. 1189–1232, 2001.