DOI:10.22413/ijatest/2026/V11/i1/03

Smart UPI Fraud Detection Using Advanced Machine Learning: An Intelligent System for Identifying Anomalies in UPI Payment Networks

D. Sreenivasulu Reddy, N.S. Rayudu Peyyala, Suheb S, Harikrishan T, Huzaifa S, Vishnuvardhan D.*
Department of AI & DS, Aditya College of Engineering,
Madanapalle – 517325, Andhra Pradesh, India
**Corresponding author: dsreddy1mpl@gmail.com, dr.rayudupeyyala@gmail.com*

Abstract: ***The rapid proliferation of digital payment systems in India has positioned the Unified Payments Interface (UPI) as the dominant platform for financial transactions, processing over 10 billion monthly transactions. However, this exponential growth has been accompanied by a corresponding surge in fraudulent activities, imposing substantial financial losses on users and financial institutions. This paper presents SMART UPI FRAUD DETECTION, an intelligent Machine Learning-based system for real-time detection of fraudulent UPI transactions. The system implements and comparatively evaluates four Machine Learning algorithms: Linear Regression, Naive Bayes Classifier, Decision Tree Classifier, and Random Forest Classifier, trained on a Kaggle UPI transaction dataset comprising 3,075 records. Comprehensive data preprocessing, feature engineering, and K-Fold cross-validation are applied to ensure robust model evaluation. The Random Forest Classifier achieves the highest performance with an accuracy of 96.80%, precision of 95.40%, recall of 94.90%, and F1-score of 95.15%. The best-performing model is deployed as a Django-based web application that enables users to upload transaction datasets, visualize results through interactive dashboards, and receive real-time fraud classification. Results confirm that ensemble methods significantly outperform individual classifiers for UPI fraud detection, providing a scalable and practical solution for enhancing digital payment security.***

Keywords**: *Anomaly detection, decision tree, Django, fraud detection, Machine Learning, random forest, UPI payment security.***

I. INTRODUCTION

The Unified Payments Interface (UPI), developed by the National Payments Corporation of India (NPCI), has emerged as the dominant digital payment ecosystem in India, recording over 10 billion monthly transactions as of 2024 [3]. This unprecedented adoption of real-time peer-to-peer and merchant payments has created a correspondingly large attack surface for fraudulent actors. Traditional rule-based fraud detection systems are inherently reactive and fail to adapt to novel and evolving fraud patterns, resulting in both missed detections and excessive false positives that disrupt legitimate users.

Machine Learning (ML) offers a powerful alternative by enabling automated learning of fraud patterns directly from transaction data. Supervised classification algorithms trained on labeled transaction records can distinguish between legitimate and fraudulent transactions with high accuracy, providing adaptive and scalable fraud detection capabilities [1]. The application of ML to UPI fraud detection is therefore both timely and practically impactful.

This paper presents SMART UPI FRAUD DETECTION USING ADVANCED ML, an end-to-end Machine Learning system for real-time UPI fraud detection. The primary contributions of this work are: (1) a comparative evaluation of four ML algorithms — Linear Regression, Naive Bayes, Decision Tree, and Random Forest — on a real-world UPI transaction dataset; (2) a comprehensive data preprocessing and feature engineering pipeline tailored to UPI transaction characteristics; and (3) the deployment of the best-performing model within a Django web application providing an interactive fraud detection dashboard for end users and administrators.

The remainder of this paper is organized as follows. Section II reviews related work. Section III describes system analysis. Section IV presents the methodology. Section V presents system design. Section VI describes implementation. Section VII discusses results and performance evaluation. Section VIII concludes the paper with future directions.

II. LITERATURE SURVEY

Fraud detection in financial transactions has been extensively studied using Machine Learning. Oza [6] applied Logistic Regression and Support Vector Machine (SVM) to payment fraud detection using labeled transaction datasets, demonstrating that ML techniques can detect fraud with high accuracy and low false positive rates. The study established a baseline for supervised classification approaches in payment fraud detection.

Mienye [1] conducted a comprehensive review of deep learning architectures for credit card fraud detection, covering Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM), and Gated Recurrent Units (GRU). The study demonstrated that deep learning approaches outperform traditional ML methods in capturing complex sequential transaction patterns, but at the cost of significantly higher computational complexity and reduced interpretability.

Sayalee [4] proposed a UPI-specific fraud detection system combining supervised classifiers and anomaly detection techniques, utilizing transactional patterns, user behaviour, and device information as features. This work specifically addressed the UPI context and demonstrated that

DOI:10.22413/ijatest/2026/V11/i1/03

domain-specific feature engineering is critical for effective detection performance.

Chawla [5] investigated online payment fraud detection using multiple ML techniques, considering transaction type, recipient identity, and transaction amount as primary features. The study highlighted that the combination of behavioral and transactional features yields superior classification performance compared to using transactional data alone.

The existing literature confirms the effectiveness of ensemble methods and hybrid approaches for financial fraud detection, but there remains a gap in evaluating and deploying complete end-to-end ML systems specifically for UPI transactions, including dataset preprocessing, comparative algorithm evaluation, and web application integration. This paper addresses that gap.

III. SYSTEM ANALYSIS

A. Existing System

Contemporary fraud detection approaches include the Risk-aware Gated Temporal Attention Network (RGTAN), which constructs a temporal transaction graph where nodes represent transactions and edges represent account interactions. The Gated Temporal Graph Attention (GTGA) mechanism propagates messages among nodes to learn adaptive transaction representations [7]. While effective, such graph-based approaches suffer from significant limitations: high computational overhead from temporal graph construction, reduced scalability with increasing transaction volumes, dependence on labeled data for graph node initialization, and complex implementation requirements that hinder practical deployment in resource-constrained environments.

Traditional rule-based fraud detection systems are limited by their static nature, inability to adapt to new fraud patterns without manual rule updates, and tendency toward high false positive rates when applied to the dynamic UPI transaction landscape.

B. Proposed System

The proposed SMART UPI FRAUD DETECT system addresses the limitations of existing approaches by implementing a practical, scalable Machine Learning pipeline for UPI fraud classification. The system analyzes transaction features including amount, frequency, timing, sender and receiver identifiers, and balance changes to identify suspicious patterns. Four ML algorithms are trained and comparatively evaluated, with the best-performing model integrated into a Django web application. The system provides: (1) enhanced fraud detection through intelligent pattern analysis; (2) continuous performance improvement through retraining on new data; (3) user-friendly visualization of fraud detection results; (4) reduced false positives compared to rule-based systems; and (5) scalable architecture capable of handling large transaction volumes.

IV. METHODOLOGY

A. Dataset Collection and Preparation

The dataset used in this study is a UPI transaction dataset obtained from Kaggle, comprising 3,075 transaction records with features extracted from UPI payment network logs. The dataset includes the following features: step (time unit), transaction type, transaction amount, sender ID (nameOrig), sender's original balance (oldbalanceOrg), sender's new balance (newbalanceOrig), receiver ID (nameDest), receiver's original balance (oldbalanceDest), receiver's new balance (newbalanceDest), and a binary fraud label (isFraudulent). Dataset profiling reveals 2,627 legitimate transactions (85.4%) and 448 fraudulent transactions (14.6%), presenting a characteristic class imbalance. The dataset is partitioned into a training set (70%) and a test set (30%) to evaluate model generalization on unseen data.

B. Data Preprocessing

Raw UPI transaction data requires extensive preprocessing before model training. The following preprocessing steps are applied: (1) loading and structural analysis using Pandas, including shape validation, data type inspection, and statistical summarization; (2) removal of duplicate transaction records; (3) handling of missing values through mean/median imputation; (4) label encoding of categorical features (transaction type: PAYMENT, TRANSFER, CASH_OUT, DEBIT, CASH_IN) to convert them to numerical representations; and (5) feature normalization using StandardScaler to ensure uniform scale across numerical features, preventing scale-dependent bias in distance-sensitive algorithms.

C. Feature Engineering

Feature engineering is applied to derive additional discriminative features from the raw transaction data. Three engineered features are created: (1) Balance Difference (balance_diff) — calculated as the difference between the new and old balance of the originating account, capturing the net transactional impact; (2) Transaction Hour — extracted from the transaction timestamp to capture time-of-day fraud patterns; and (3) Encoded Transaction Type — numerical encoding of the categorical payment type. The final feature vector for model training consists of seven features: encoded transaction type, transaction amount, original sender balance, new sender balance, original receiver balance, new receiver balance, and balance difference.

D. Machine Learning Algorithms

Four ML algorithms are implemented and comparatively evaluated in this study:

Decision Tree Classifier: A non-parametric supervised learning algorithm that partitions the feature space through recursive binary splitting based on information gain (Gini impurity criterion). It produces interpretable if-then decision rules and achieves high accuracy for structured tabular data. A maximum depth constraint is applied to prevent overfitting.

Random Forest Classifier: An ensemble method that constructs multiple decision trees using bootstrap sampling and random feature subsets (bagging). The final classification is determined by majority voting across 100 estimators. This ensemble approach reduces variance, mitigates overfitting, and handles class imbalance effectively [13].

Naive Bayes Classifier: A probabilistic classifier based on Bayes' Theorem with the conditional independence assumption among features. The Gaussian variant is applied for continuous transaction features. Despite the independence

assumption, Naive Bayes delivers competitive performance for high-dimensional fraud detection tasks with significantly lower computational cost.

Linear Regression (Baseline): Applied as a baseline regression model for fraud probability estimation. Evaluated using Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and R-squared metrics to establish a performance lower bound.

E. Model Training and Validation

All four models are trained on the preprocessed feature-engineered training dataset. K-Fold Cross Validation with k=5 is applied to each model to obtain unbiased performance estimates independent of the specific train-test split. Model hyperparameters are configured as follows: Random Forest with n_estimators=100, max_features='sqrt'; Decision Tree with criterion='gini'; Naive Bayes with Gaussian variant; Linear Regression with default parameters. The serialized best-performing model is stored as a .pkl file using joblib for integration with the Django web application.

V. SYSTEM DESIGN

A. System Architecture

The SMART UPI FRAUD DETECT system adopts three-layer architecture. The Data Layer receives raw CSV transaction datasets through the web interface and manages feature storage in a SQLite database via Django's ORM. The Processing Layer constitutes the core ML pipeline encompassing data preprocessing, feature engineering, parallel model training with cross-validation, accuracy comparison, and model selection. The Presentation Layer is the Django-based web application providing user authentication, file upload management, ML pipeline triggering, and an interactive fraud detection dashboard with visualization charts, flagged transaction tables, and downloadable reports.

B. Web Application Design

The Django web application provides the following functional modules: (1) User Authentication — secure registration and session-based login; (2) Transaction Input — CSV file upload interface with format validation; (3) Fraud Detection Dashboard — real-time display of fraud classification results for all uploaded transactions; (4) Visualization Module — interactive charts including confusion matrices, algorithm accuracy comparison bar charts, and transaction distribution plots; (5) Report Generation — downloadable fraud detection reports in PDF format; and (6) Prediction Records — persistent storage and retrieval of all historical detection sessions.

VI. IMPLEMENTATION

A. Technology Stack

The system is implemented using Python 3.8+ as the primary programming language. Machine Learning components utilize Scikit-learn for algorithm implementation, NumPy and Pandas for data manipulation, and Matplotlib/Seaborn for visualization. The web application is built using Django 4.x with SQLite3 as the database backend. Development tools include Anaconda Navigator, Jupyter Notebook for model prototyping, and Visual Studio Code for application development.

B. System Modules

The implementation is organized into seven functional modules: Module 1 (Data Preprocessing) — handles dataset loading, cleaning, encoding, and normalization; Module 2 (Exploratory Data Analysis) — generates histograms, correlation heatmaps, box plots, and class distribution charts; Module 3 (Linear Regression) — implements baseline regression fraud estimation; Module 4 (Naive Bayes Classifier) — implements Gaussian probabilistic classification; Module 5 (Decision Tree Classifier) — implements interpretable rule-based classification; Module 6 (Random Forest Classifier) — implements ensemble classification with cross-validation; Module 7 (Django Web Application) — integrates all modules with URL routing, template rendering, and database management.

C. Dataset Characteristics

The profiling report of the Kaggle UPI dataset reveals 10 variables across 3,075 observations with no missing values and negligible duplicate entries. The Average Amount Transaction Day variable exhibits a mean of 515.02, ranging from 4.01 to 2000, with 98.1% distinct values indicating high transactional variability. The Transaction Amount variable shows a mean of 9,676.36 (range: 0–100,000) with values concentrated at lower amounts, reflecting real-world UPI transaction distribution. The is Fraudulent variable confirms class imbalance with 2,627 legitimate (85.4%) vs. 448 fraudulent (14.6%) transactions, validating the use of precision, recall, and F1-score as primary evaluation metrics alongside accuracy.

VII. RESULTS AND DISCUSSION

A. Performance Evaluation

The performance of all four Machine Learning algorithms is evaluated using K-Fold cross-validation (k=5) on the preprocessed UPI transaction dataset. Table 1 summarizes the comparative performance results across Accuracy, Precision, Recall, and F1-Score.

TABLE 1: Comparative Performance of Machine Learning Algorithms for UPI Fraud Detection

Algorithm	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Linear Regression	85.20	83.10	80.50	81.78
Naive Bayes Classifier	88.40	86.20	85.90	86.05
Decision Tree Classifier	91.60	90.10	89.80	89.95
Random Forest Classifier	**96.80**	**95.40**	**94.90**	**95.15**

The Random Forest Classifier achieves the highest performance with an accuracy of 96.80%, precision of 95.40%, recall of 94.90%, and F1-Score of 95.15%. This superior performance is attributed to its ensemble nature combining 100 decision trees, which effectively reduces

variance, resists overfitting, and handles the class imbalance inherent in fraud datasets where legitimate transactions significantly outnumber fraudulent ones.

B. Discussion

The performance hierarchy observed — Random Forest (96.80%) > Decision Tree (91.60%) > Naive Bayes (88.40%) > Linear Regression (85.20%) — aligns with established literature on ensemble methods for imbalanced classification tasks [1][13]. The 5.20% accuracy gap between Random Forest and Decision Tree confirms that ensemble averaging substantially reduces the variance errors of individual decision trees on this dataset.

The Naive Bayes classifier, despite its strong independence assumption, achieves competitive performance (88.40% accuracy, 86.05% F1-score) with significantly lower computational overhead, making it a viable option for latency-sensitive deployment scenarios where the marginal accuracy reduction is acceptable.

The confusion matrix for the Random Forest Classifier confirms the practical reliability of the model: 522 true negatives and 86 true positives, with only 3 false positives and 4 false negatives. The low false negative count (4 missed frauds out of 90 total fraudulent transactions) is particularly significant for financial security applications where undetected fraud carries severe consequences.

The Random Forest classifier's feature importance analysis identifies transaction amount, balance difference, and encoded transaction type as the three most discriminative features for UPI fraud classification, consistent with domain knowledge that fraudulent transactions exhibit distinctive amount profiles and unusual balance change patterns.

VIII. CONCLUSION

This paper presented SMART UPI FRAUD DETECT, a comprehensive Machine Learning system for detecting fraudulent UPI transactions. Through comparative evaluation of four ML algorithms, the Random Forest Classifier was identified as the optimal model achieving 96.80% accuracy, 95.40% precision, 94.90% recall, and 95.15% F1-score. The system successfully integrates the complete ML lifecycle — data collection, preprocessing, feature engineering, model training, evaluation, and deployment — within a production-ready Django web application. The modular architecture ensures independent updatability of each component, enabling future upgrades such as deep learning model substitution without systemic disruption. The results confirm that ensemble Machine Learning methods provide a practical, scalable, and accurate solution for UPI fraud detection, contributing meaningfully to the security of India's growing digital payments ecosystem.

IX. FUTURE WORK

Future research directions for SMART UPI FRAUD DETECT include: (1) Integration of LSTM and GRU networks to capture temporal fraud patterns across sequential transaction histories; (2) Application of Graph Neural Networks (GNNs) to model inter-account relationships and detect coordinated fraud rings; (3) Implementation of real-time streaming detection using Apache Kafka for millisecond-level fraud prevention at the point of transaction; (4) Cloud deployment on AWS or Google Cloud Platform with auto-scaling capabilities to handle millions of transactions per day; (5) Integration of automated alert mechanisms via Twilio SMS and SMTP email APIs for instant notification to users and regulatory authorities; and (6) Federated learning approaches to train fraud models across distributed banking systems without centralizing sensitive transaction data.

ACKNOWLEDGMENT

The authors sincerely thank Dr. D. Sreenivasulu Reddy, M.Tech., Ph.D., Professor, Department of AI & DS, Aditya College of Engineering, Madanapalle, for his invaluable guidance and supervision throughout this project. The authors also acknowledge the support of Dr. R. Md. Shafi, Professor & Head of Department, Dr. Rayudu Peyyala, Principal, and Dr. S. Ramalinga Reddy, Director, Aditya College of Engineering.

REFERENCES

[1] I. D. Mienye, "Deep learning for credit card fraud detection: A review of algorithms, challenges, and solutions," IEEE Access, vol. 12, 2024.
[2] S. S. Bodade, "Implementation paper on UPI fraud detection using Machine Learning," 2024.
[3] NPCI, "UPI Product Statistics," National Payments Corporation of India. [Online]. Available: https://www.npci.org.in/, 2024.
[4] S. Sayalee, "Review paper on UPI fraud detection using Machine Learning," IJARCS, vol. 14, no. 4, 2023.
[5] T. S. Chawla, "Online payment fraud detection using Machine Learning techniques," Int. J. Comput. Appl., vol. 184, no. 42, 2022.
[6] A. Oza, "Fraud detection using Machine Learning," IJERT, vol. 10, no. 3, 2021.
[7] V. N. Dorna Dula and S. Geetha, "Credit card fraud detection using Machine Learning algorithms," Procedia Comput. Sci., vol. 165, pp. 631–641, 2019.
[8] J. Jurgovsky et al., "Sequence classification for credit card fraud detection," Expert Syst. Appl., vol. 100, pp. 234–245, 2018.
[9] S. Bhattacharyya et al., "Data mining for credit card fraud: A comparative study," Decis. Support Syst., vol. 50, no. 3, pp. 602–613, 2011.
[10] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," J. Mach. Learn. Res., vol. 12, pp. 2825–2830, 2011.
[11] W. McKinney, "Data structures for statistical computing in Python," in Proc. 9th Python Sci. Conf., 2010.
[12] G. Van Rossum and F. L. Drake, Python 3 Reference Manual. Scotts Valley, CA: CreateSpace, 2009.
[13] L. Breiman, "Random forests," Mach. Learn., vol. 45, no. 1, pp. 5–32, 2001.