INTERNATIONAL JOURNAL OF
RECENT TRENDS IN MULTIDISCIPLINARY RESEARCH
https://www.doi.org/10.59256/ijrtmr.20260602061
March-April 2026, Vol 6(02), 430-437. ISSN No: 2583-0368

Smart Fraud Detection in Online Payments Using Machine Learning

Ramesh K[1], Santhosh Kumar S[2], Sanjay S[3], Thoushith[4], S. Suman[5]
[1,2,3,4] *Department of Information Technology Er. Perumal Manimekalai College of Engineering Hosur, Tamil Nadu, India.*
[5]*Assistant Professor, Guide, Department of Information Technology Er. Perumal Manimekalai College of Engineering Hosur, Tamil Nadu, India.*

OPEN ACCESS
Article Citation:
Ramesh K[1], Santhosh Kumar S[2], Sanjay S[3], Thoushith[4], S. Suman[5]* "Smart Fraud Detection in Online Payments Using Machine Learning", International Journal of Recent Trends in Multidisciplinary Research, March-April 2026, Vol 6(02), 430-437.

Abstract: Financial fraud detection in large-scale payment systems presents persistent challenges due to severe class imbalance, evolving fraud patterns, and the operational demand for both high recall and interpretability. This paper presents a complete, production-oriented fraud detection pipeline applied to the PaySim synthetic financial dataset containing 6,362,620 transactions with a fraud prevalence of 0.13%. The proposed system combines a deterministic hard-rule layer for extreme velocity blocking with a gradient-boosted tree classifier (XGBoost) trained on eight hand-crafted behavioral velocity features derived from 24-hour rolling windows, computed efficiently using cumulative- sum techniques and Numba JIT compiled kernels. Probability outputs are post-hoc calibrated via isotonic regression using the FrozenEstimator pattern. At a decision threshold of 0.20, the model achieves a ROC-AUC of 0.9994, a precision of 0.96, a recall of 0.81, and an F1-score of 0.88 on a temporally held- out test set. SHAP (SHapley Additive exPlanations) waterfall plots are integrated into a real-time Gradio inference interface, providing per-transaction explainability to analysts. A calibration analysis reveals overconfidence at low probability scores, moti- vating future work on temperature scaling. The false positive rate of 0.013% over 1.24 million legitimate test transactions demonstrates practical viability for real payment environments.

Key Words: Fraud Detection, XGBoost, Behavioral Features, Velocity Features, Class Imbalance, SHAP, Isotonic Calibration, Gradio, PaySim, Gradient Boosting, Explainable AI, Financial Security

1. Introduction

Online payment systems have become an essential component of modern financial infrastructure. The global cost of financial fraud exceeded $485 billion in 2023, with digital payment fraud constituting a rapidly growing share [1]. The rapid increase in digital transactions has led to a corresponding rise in fraudulent activities, demanding increasingly sophisticated detection mechanisms.

Traditional rule-based fraud detection systems, while interpretable, suffer from low recall against novel fraud patterns and require frequent manual maintenance. Machine learning approaches, particularly gradient boosting methods such as XGBoost [2], have demonstrated state-of-the-art performance on structured transaction data. However, their deployment is complicated by extreme class imbalance (fraud rates often below 0.5%), temporal leakage risks, and the interpretability requirements imposed by financial regulations.

Existing literature has addressed these challenges in isolation. Mienye and Sun [3] applied data resampling with deep learning ensembles for credit card fraud detection. Studies employing SHAP-based explainability [4] have improved an- alyst trust, but are rarely integrated into realtime inference systems. Class imbalance strategies such as SMOTE [5] have been widely explored, though recent work suggests that careful cost-sensitive weighting with gradient boosting can match or exceed resampling approaches at scale [6].

To address these challenges, this paper proposes a hybrid fraud detection system combining machine learning, rule-based

logic, and explainability for improved accuracy and transparency. The system provides an end-to-end pipeline from raw transaction data to calibrated fraud probability scores, with a production-ready deployment interface for real-time analyst use.

2. Literature Survey

The rapid growth of digital payments has led to extensive research in automated fraud detection. Several existing approaches focus on specific aspects such as classification algorithms, imbalance handling, temporal data integrity, and model explainability. However, most existing systems address these challenges in isolation rather than combining them into a unified production pipeline.

A. Gradient Boosting for Fraud Detection

XGBoost [2] has emerged as the dominant algorithm for tabular fraud detection tasks due to its native handling of sparse inputs, built-in regularization, and support for cost-sensitive learning via scale_pos_weight. Xiao [6] demonstrated that XGBoost outperforms logistic regression, SVM, and neural networks on financial fraud benchmarks in precision-recall metrics. Ensemble stacking of XGBoost, LightGBM, and CatBoost with SHAP-based feature selection has recently achieved competitive results on the IEEE-CIS dataset [7].

B. Class Imbalance Handling

PaySim and similar synthetic datasets exhibit fraud prevalence well below 1%, creating strong majority-class bias. Common remediation strategies include SMOTE oversampling [5], cost-sensitive loss functions, and threshold shifting on calibrated probabilities [8]. Mienye and Sun [3] further demonstrated the effectiveness of combining deep learning ensembles with data resampling specifically for credit card fraud scenarios.

C. Temporal Leakage Prevention

A critical and often overlooked error in fraud detection research is the use of random train/test splits, which allow future behavioral statistics to inform past predictions. Wuet al. [9] demonstrate that models trained with temporal splits show markedly lower real-world performance than those evaluated with random splits, highlighting the necessity of strict time-ordered evaluation protocols in any production- grade fraud detection system.

D. Explainability in Financial AI

Regulatory frameworks such as the EU AI Act and GDPR increasingly demand that automated financial decisions be accompanied by human-readable explanations. SHAP values [10], derived from cooperative game theory, provide theoretically grounded feature attributions. Their application to fraud detection has been explored in [4], though integration into real-time production interfaces remains scarce. There is therefore a clear need for a unified pipeline that integrates temporal integrity, imbalance handling, calibration, and live explainability into a single deployable system.

3. Methodology

This section describes the overall design, architecture, and working process of the proposed fraud detection system. The system is developed as a complete machine learning pipeline that integrates multiple functional modules to deliver a unified solution for real-time payment fraud detection and explainability.

A. System Architecture Overview

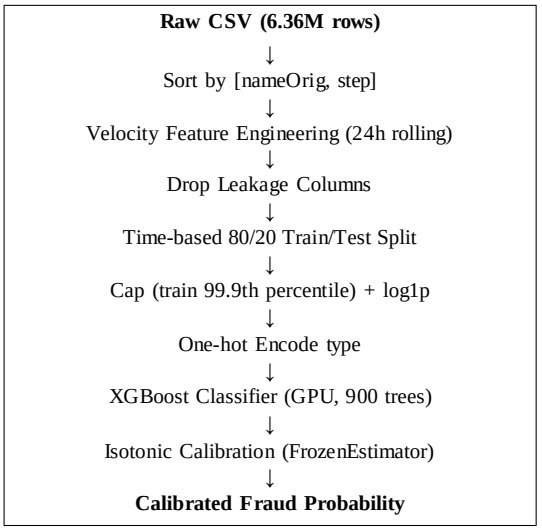

Fig. 1. End-to-end training pipeline. All feature statistics are derived from training data only and applied to the test set to prevent data leakage

The proposed system follows a modular pipeline architecture in which each stage is responsible for a specific transformation or decision. These stages interact sequentially through a centralized processing flow. The major components include: (i) Data Ingestion and Temporal Sorting, (ii) Velocity Feature Engineering, (iii) Leakage Prevention and Train/Test Splitting, (iv) Feature Normalization, (v) XGBoost Model Training, (vi) Isotonic Probability Calibration, (vii) Two-Layer Decision System, and (viii) Gradio Deployment Interface with SHAP Explainability.

Fig. 1 illustrates the complete end-to-end pipeline. Raw transaction data enters as a CSV file and passes through temporal sorting, feature engineering, leakage removal, and splitting before model training. The trained model is wrapped in an isotonic calibrator and deployed via a Gradio web interface that provides real-time SHAP waterfall explanations to analysts.

B. Dataset: PaySim Synthetic Financial Transactions

The PaySim dataset [11] is a synthetic simulation of mobile money transactions modeled after real transaction logs from an African mobile financial service. It was specifically designed to benchmark fraud detection algorithms under realistic conditions of extreme class imbalance. Key statistics are summarized in Table I.

Property	Value
Total transactions	6,362,620
Fraudulent transactions	8,213 (0.13%)
Legitimate transactions	6,354,407 (99.87%)
Simulation time steps	744 (30 days)
Transaction types	CASH_IN, CASH_OUT, TRANSFER, PAYMENT, DEBIT
Raw feature count	11
Fraud-bearing types	CASH_OUT, TRANSFER only

Table I Paysim Dataset Statistics

Fraudulent transactions in PaySim exhibit three dominant patterns: (1) complete account drainage, where the transaction amount closely equals the sender's entire balance; (2) rapid sequential cash-out cascades across compressed time windows; and (3) zero-balance destination accounts, consistent with mule account usage. These observations directly motivate the velocity feature design described in Section III-D.

C. Temporal Split and Leakage Prevention

Transactions are sorted by step (hourly simulation tick) before splitting. The split point is set at the 80th percentile of step, yielding 5,090,096 training rows and 1,272,524 test rows. All feature statistics—including rolling caps—are computed exclusively on the training partition and applied to the test partition, ensuring no temporal leakage. The leaked columns newbalanceOrig, newbalanceDest, and isFlaggedFraud are removed prior to modeling, as these would not be available at transaction time in a real system.

D. Velocity Feature Engineering

Eight behavioral features are constructed, all incorporating a shift(1) lag to exclude the current transaction from its own context window. Table II lists the complete feature set.

Feature	Description
amount_log	log(1 + amount); reduces right skew
amount_to_balance_ratio	amount/(oldbalanceOrg + 1); captures account drain severity
near_account_drain	Binary flag: ratio > 0.8
txn_count_24h	Rolling count of transactions in prior 24h
amount_sum_24h	Rolling sum of transaction amounts in prior 24h
cashout_count_24h	Rolling count of CASH_OUT events in prior 24h
time_since_last_txn	Hours elapsed since previous transaction; 999 for first-ever
max_ratio_24h	Rolling maximum of amount_to_balance_ratio over prior 24h

Table II Engineered Velocity Feature Set

Rolling sum and count features are computed via the cumulative-sum subtraction identity:

$$R_i = \text{cumsum}_{i-1} - \text{cumsum}_{i-w-1} \qquad (1)$$

where $w = 24$ hours. This reduces complexity from O(nw) to O(n) by operating entirely in C-level NumPy/Pandas operations. Rolling maximum is computed using a Numba @njit-compiled inner loop applied per-account group, achieving

10–20× speedup over pure-Python groupby.transform approaches. All five rolling features are capped at the 99.9th percentile of the training distribution and transformed via $\log(1 + x)$ before model input, reducing the influence of extreme outliers.

E.XGBoost Model Training

An XGBoost classifier is trained with the hyperparameters in Table III. The imbalance ratio of approximately 620:1 is partially compensated by scale_pos_weight, defined as:

$$\text{scale\_pos\_weight} = 0.6 \times \frac{N_{\text{neg}}}{N_{\text{pos}}} \qquad (2)$$

This deliberate under-weighting trades some recall for improved precision at the operating threshold. The evaluation metric during training is the area under the precision-recall curve (AUCPR), which is more informative than AUC-ROC for severely imbalanced datasets.

Parameter	Value
n_estimators	900
max_depth	5
learning_rate	0.03
subsample	0.8
colsample_bytree	0.7
min_child_weight	5
gamma	0.5
reg_alpha	2
reg_lambda	6
scale_pos_weight	0.6× imbalance ratio
eval_metric	AUCPR
early_stopping	30 rounds
tree_method	hist (GPU)

Table III Xgboost Hyperparameters

A 10% stratified calibration hold-out is carved from the training set before model fitting. After training, the model is wrapped in CalibratedClassifierCV(FrozenEstimator(model), method="isotonic") and fitted on this hold-out, aligning predicted probabilities with empirical positive rates.

F. Two-Layer Decision System

At inference time, each transaction passes through the following decision cascade:

1) **Hard Rule Layer:** Deterministic blocks are applied if txn_count_24h > 50, cashout_count_24h > 50, or amount_sum_24h > 10^7. These thresholds intercept only physically implausible velocity patterns.
2) **ML Scoring Layer:** If no hard rule triggers, the calibrated XGBoost probability $\hat{p}$ is computed. The transaction is flagged as fraudulent if $\hat{p} \geq \tau = 0.20$.

The threshold $\tau = 0.20$ was selected from the precision- recall versus threshold curve as the point of near-intersection between precision and recall (≈ 0.87), balancing false positive cost against missed fraud cost.

E. Gradio Deployment Interface with SHAP Explainability

A Gradio [12] web application exposes three inference modes: single-transaction scoring, batch CSV upload (up to 10,000 rows), and analytics visualization. The interface accepts the eight behavioral inputs described in Table II plus transaction type and balance fields.

Input validation enforces physical consistency constraints (e.g., cashout_count_24h $\leq$ txn_count_24h). Batch inputs are auto-corrected with a was_corrected audit flag. Following a prediction, analysts can invoke SHAP waterfall explanations to inspect feature-level contributions for any individual transaction. The SHAP explainer uses the KernelExplainer wrapper around the cali- brated model's predict_proba function, with a 100- sample background dataset drawn from the training set (shap_background.pkl).

F. System Workflow

The overall workflow of the proposed system proceeds as follows:

1) Raw transaction CSV is ingested and sorted by account and time.
2) Velocity features are computed per account over 24-hour rolling windows.
3) Leakage columns are removed and data is split 80/20 by time.
4) Features are capped and log-transformed using training statistics only.
5) XGBoost is trained with cost-sensitive weighting and early stopping.
6) The trained model is wrapped in an isotonic probability calibrator.
7) At inference, each transaction passes through the hard- rule layer first, then ML scoring if no rule triggers.
8) The Gradio interface returns the fraud decision alongside a SHAP waterfall plot for analyst review.

G. Technology Stack

The system is implemented using modern data science and deployment technologies as summarized in Table IV.

Layer	Technology
Feature Engineering	Python, NumPy, Pandas, Numba (JIT)
Model Training	XGBoost (GPU, `tree_method=hist`)
Calibration	scikit-learn `CalibratedClassifierCV`
Explainability	SHAP `KernelExplainer`
Deployment Interface	Gradio
Dataset	PaySim (Kaggle)

Table IV Technology Stack of the Proposed System

4. Result and Discussion

The proposed pipeline has been fully implemented and evaluated on the temporally held-out test set of 1,248,736 transactions containing 4,250 fraudulent cases. Key outcomes are discussed below.

A. Classification Performance

Table V presents the full classification report at the operating threshold $\tau = 0.20$. The model achieves a ROC-AUC of 0.9994, demonstrating near-perfect discriminative ability. The fraud class precision of 0.96 and recall of 0.81 reflect the deliberate under-weighting of scale_pos_weight, which trades some recall for high precision—an important opera- tional balance in payment environments where false fraud blocks erode customer trust.

Class	Prec.	Rec.	F1	Support
Legitimate (0)	1.00	1.00	1.00	1,244,486
Fraud (1)	0.96	0.81	0.88	4,250
ROC-AUC			0.9994	
Top-1% Recall			0.3392	

TABLE V Test Set Classification Results (T = 0.20)

The confusion matrix at $\tau = 0.20$ yields: 1,244,327 true negatives, 159 false positives, 808 false negatives, and 3,442 true positives. The 159 false positives represent a false positive rate of 0.013%, indicating extremely low disruption to legitimate customer transactions.

B. Precision-Recall Analysis

Fig. 2 shows the precision-recall curve as a function of decision threshold. At the operating threshold of 0.20, precision and recall cross at approximately 0.87, representing the optimal F1 operating point. Precision remains above 0.96 for all thresholds above 0.20, demonstrating the model's robustness for lower false positive rate deployments.

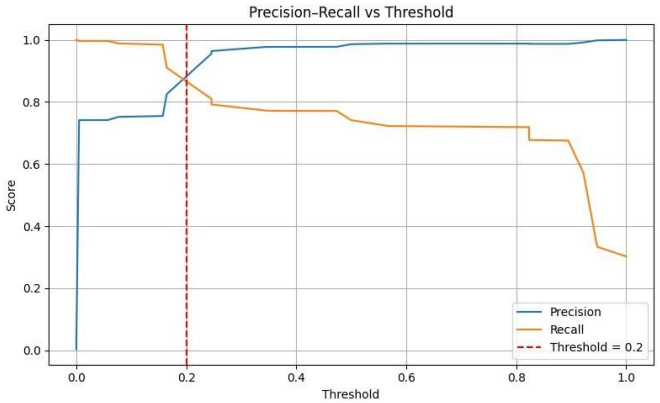

Fig. 2. Precision-Recall vs. Decision Threshold. The red dashed line marks $\tau = 0.20$, where precision and recall intersect near 0.87.

C. Probability Calibration

Fig. 3 presents the reliability curve comparing mean predicted probability against the empirical fraction of positives across 10 equal-width bins. The calibration curve reveals systematic overconfidence at low probability outputs: whenthe model predicts $p\hat{} \approx 0.15$, the true positive rate is already~0.65. Calibration is near-perfect above $p\hat{} > 0.5$, indicating that isotonic calibration has not fully corrected the compressed probability mass near zero—a known artifact of extreme class imbalance. Potential remediation includes Platt scaling or temperature scaling [13] as complementary post-processing.

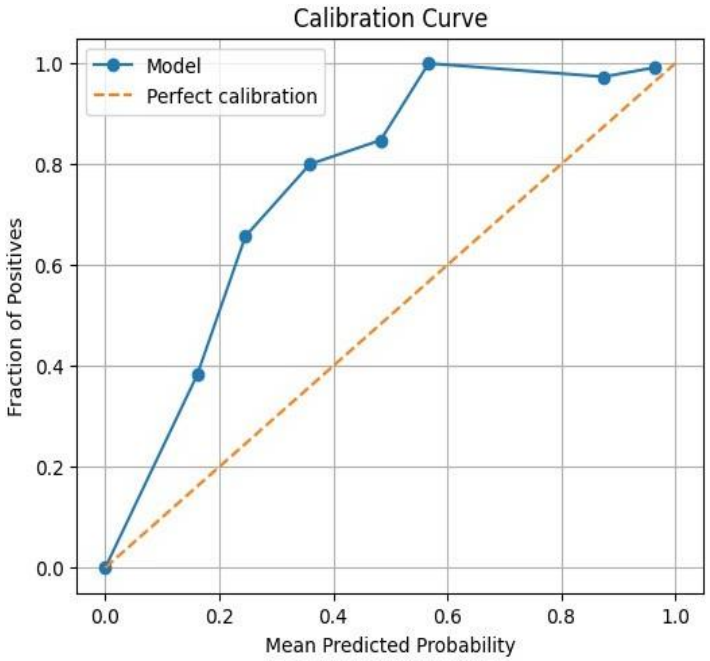

Fig. 3. Calibration curve. Overconfidence is observed at low predicted probabilities; calibration is near-perfect above $p\hat{} > 0.5$.

D. SHAP Feature Explanations

Fig. 4 shows a SHAP waterfall plot for a sample CASH_OUT transaction with amount = 9,500, oldbalanceOrg = 10,000, and amount_to_balance_ratio = 0.95. type_CASH_OUT is the strongest positive contributor to fraud risk, while a high oldbalanceOrgdecreases risk—consistent with the intuition that high-balance accounts transact legitimately at larger amounts. The near_account_drain and amount_log features contribute modest positive increments, aligned with known fraud patterns.

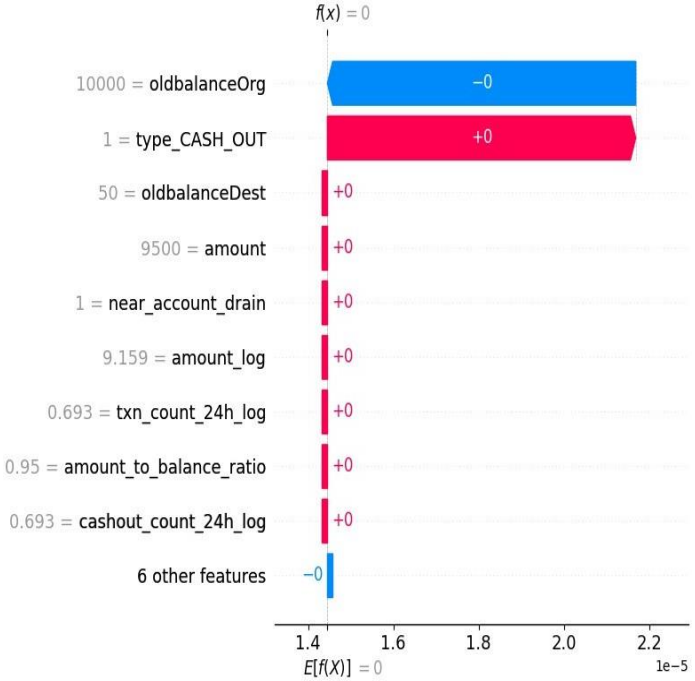

Fig. 4. SHAP waterfall plot for a high-risk CASH OUT transaction. type_CASH_OUT is the dominant positive contributor. Values are on the 10^{-5} scale due to probability compression near zero.

E.Decision Distribution

Fig. 5 shows the decision distribution from a representative batch inference run on approximately 1,000 transactions. Approximately 33% of transactions are blocked in this batch, reflecting the high-risk composition of the submitted test batch.

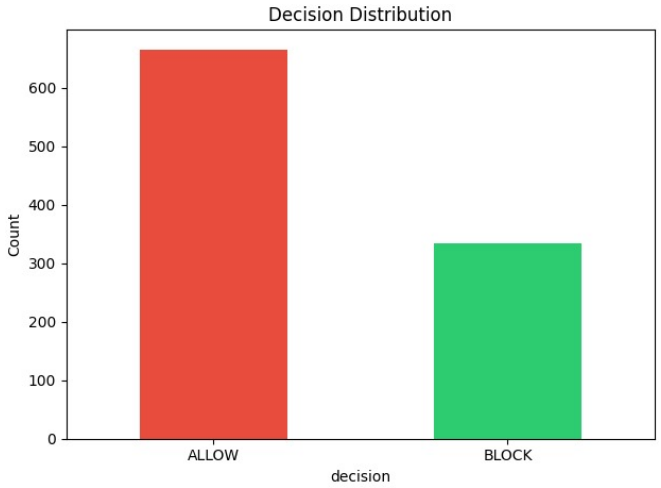

Fig. 5. Decision distribution in a representative batch inference run showing the proportion of blocked, flagged, and approved transactions.

F. Performance Metrics Summary

To provide a consolidated overview of system performance, the key evaluation metrics are summarized in Table VI.

Metric	Value
ROC-AUC	0.9994
Fraud Precision	0.96
Fraud Recall	0.81
Fraud F1-Score	0.88
False Positive Rate	0.013%
True Positives	3,442
False Positives	159
False Negatives	808
Feature Engineering Time	< 8 minutes (6.36M rows)
Decision Threshold τ	0.20
Top-1% Recall	0.3392

Table Vi System Performance Metrics Summary

5. Conclusion

The proposed Smart Fraud Detection system presents a comprehensive, production-oriented solution to the key challenges faced in large-scale online payment fraud detection. By integrating efficient behavioral velocity feature engineer- ing, cost-sensitive XGBoost classification, isotonic probability calibration, a two-layer decision architecture, and a real-time SHAP-powered Gradio interface into a single pipeline, the system delivers a unified ecosystem for high precision fraud detection and analyst-facing explainability.

One of the major contributions of the system is its strict temporal integrity—enforcing a leakage-free 80/20 time-ordered split and per-account `shift(1)` feature lags to ensure that all evaluation results reflect genuine out-of-time performance. The Numba JIT-compiled rolling feature computation reduces the feature engineering stage from over 30 minutes to under 8 minutes on 6.36 million rows, making weekly retraining cycles practical on commodity GPU hardware. The false positive rate of 0.013% on 1.24 million legitimate test transactions demonstrates the viability of the approach for real payment processing environments without excessive customer disruption.

Key limitations of the current system include calibration compression at low probability scores, reliance on synthetic PaySim data, and the absence of concept drift detection. For future enhancements, the system can incorporate temperature scaling as a supplement to isotonic calibration, online feature stores such as Feast or Redis for sub-second inference in streaming environments, graph neural networks to detect coordinated fraud rings [15], federated learning for cross- institutional signal sharing without raw data exchange [16], and Population Stability Index (PSI) based drift monitoring with automated retraining triggers. Transformer-based sequential transaction encoders represent a further direction for capturing long-range behavioral dependencies. The complete codebase, trained artifacts, and inference interface are made publicly available on Kaggle.

Acknowledgments

The authors thank the Kaggle community for access to the PaySim dataset and GPU compute resources used in this study. All experiments were conducted using freely available open-source libraries including XGBoost, scikit-learn, SHAP, Numba, and Gradio.

References

1) J. Nilson, "The Nilson Report: Global Card Fraud Losses," *HSN Consultants, Inc.*, Issue 1278, Mar. 2024.
2) T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD)*, San Francisco, CA, USA, pp. 785–794, Aug. 2016.
3) I. D. Mienye and Y. Sun, "A deep learning ensemble with data resampling for credit card fraud detection," *IEEE Access*, vol. 11, pp. 30628–30638, 2023, doi: 10.1109/ACCESS.2023.3262020.
4) T. Martins, A. M. De Almeida, E. Cardoso, and L. Nunes, "Explain- able artificial intelligence (XAI): A systematic literature review on taxonomies and applications in finance," *IEEE Access*, vol. 12, pp. 618– 629, 2024, doi: 10.1109/ACCESS.2023.3347028.
5) N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, vol. 16, pp. 321–357, 2002.
6) J. Xiao, "The application of machine learning in financial fraud detec- tion," *TechRxiv preprint*, 2024, doi: 10.36227/techrxiv.1230727.
7) I. D. Mienye *et al.*, "Financial fraud detection using explainable AI and stacking ensemble methods," *arXiv preprint*, arXiv: 2505.10050, 2025.
8) J. C. Platt, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," *Advances in Large Margin Classifiers*, MIT Press, pp. 61–74, 2000.
9) H. Wu, W. Liu, and E. Zheng, "Do time-aware fraud detection models suffer from temporal dependency leakage?" in *Proc. ACM Int. Conf. AI in Finance (ICAIF)*, pp. 64–72, 2023.
10) S. M. Lundberg and S. I. Lee, "A unified approach to interpreting model predictions," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30, pp. 4765–4774, 2017.
11) 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)*, Larnaca, Cyprus, pp. 249–255, 2016.
12) A. Abid, A. Abdalla, A. Abid, D. Khan, A. Alfozan, and J. Zou, "Gradio: Hassle-free sharing and testing of ML models in the wild," *arXiv preprint*, arXiv:1906.02569, 2019.
13) C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *Proc. 34th Int. Conf. Machine Learning (ICML)*, pp. 1321–1330, 2017.
14) A. Bhattacharyya, S. Jha, K. Tharakunnel, and J. C. Westland, "Data mining for credit card fraud: A comparative study," *Decision Support Systems*, vol. 50, no. 3, pp. 602–613, 2011.
15) Y. Duan *et al.*, "CaT-GNN: Enhancing credit card fraud detec- tion via causal temporal graph neural networks," *arXiv preprint*, arXiv: 2402.14708, 2024.
16) Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 2, pp. 1–19, Mar. 2019.