Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

FedGuard: An Explainable Centralized Fraud-Detection Baseline on PaySim and IEEE-CIS

Naman Dugar
Department of Computer Science and Engineering, Bharati Vidyapeeth's College of Engineering, New Delhi, India https://orcid.org/0009-0005-2119-4128

Anvi Tyagi
Department of Computer Science and Engineering, Bharati Vidyapeeth's College of Engineering, New Delhi, India https://orcid.org/0009-0007-4091-9897

Riyanshu
Department of Computer Science and Engineering, Bharati Vidyapeeth's College of Engineering, New Delhi, India https://orcid.org/0009-0007-6984-2782

Prof. Mohit Tiwari
mohit.tiwari@bharatividyapeeth.edu

Department of Computer Science and Engineering,Bharati Vidyapeeth's College of Engineering,Delhi
https://orcid.org/0000-0003-1836-3451

Research Article

Keywords: fraud detection, class imbalance, explainable AI, TreeSHAP, LightGBM, federated learning (future work)

Posted Date: November 19th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11739412/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License. Read Full License

Additional Declarations: The authors declare no competing interests.

Abstract

Fraud detection is difficult because fraudulent transactions are rare, data differ across domains, and strong predictive models can be hard to review. We present FedGuard, an explainable centralized baseline evaluated separately on two public benchmark domains: PaySim, a synthetic mobile-money dataset with 6.36 million transactions and 0.13% fraud, and IEEE-CIS, an anonymized card-not-present e-commerce dataset with 590,540 labeled transactions and 3.5% fraud. The records were split into train, validation, and test partitions before any preprocessing was fitted. Train-only preprocessing and sampling were used in four model families and 195 candidate fits. Thresholds were selected using validation data to maximize recall subject to an FPR no greater than 1%. On the held-out test split, the selected LightGBM model reached AP 0.997968, recall 99.76%, and FPR 0.003567% on PaySim; for IEEE-CIS, AP was 0.756757, recall 68.80%, and FPR 1.017758%, slightly above the validation target. TreeSHAP contributions for the selected tree model sum with the baseline value to the model raw output; they are not calibrated probabilities. The codebase includes a 55-case software test suite, which checks implementation behavior rather than trained-model performance. This work does not implement federated learning, differential privacy, or secure aggregation.

1. Introduction

Fraud patterns can cross organizational boundaries, while financial transaction records are sensitive and often subject to privacy, contractual, and data-governance controls. These constraints can limit centralized sharing of raw data. A second challenge is operational: risk teams need to understand why a model raised an alert, especially when the classifier is complex.

FedGuard's longer-term goal is to study whether institutions can train useful fraud models without pooling raw customer records. This paper reports the completed centralized baseline only. PaySim and IEEE-CIS are used as separate benchmark domains; they are not records from two participating institutions, and the study does not simulate a multi-institutional deployment. The controls reported here are limited to splitting first, fitting preprocessing on training rows, resampling training rows only, choosing the model and threshold on validation data, and evaluating the selected model once on the held-out test split.

Objectives. The objectives are to (i) audit and prepare two distinct fraud benchmarks using a stratified, reproducible split; (ii) fit all feature filtering, imputation, encoding, scaling, and resampling steps on the training partition only; (iii) compare centralized models using validation Average Precision (AP) and a validation FPR ceiling of 1%; and (iv) provide local TreeSHAP explanations and a demonstration dashboard while keeping the limits of explanation clear.

Contributions. The work makes four contributions:

- a reproducible pipeline for two separate benchmark domains, with train-only preprocessing and sampling and validation/test rows kept outside model fitting;
- a comparison of four model families across 195 candidate fits, with validation-only model selection and a single final evaluation of each selected model on its held-out test split;

- an operational evaluation based on a validation FPR ceiling, including the measured IEEE-CIS test result that slightly exceeds that ceiling;
- a local TreeSHAP explanation path and Streamlit dashboard. The repository includes a 55-case automated software suite; these checks cover code behavior and do not establish the predictive validity of the fitted models.

2. Related Work

Class imbalance is a central challenge in fraud detection. Chawla et al. [1] introduced SMOTE oversampling, while Batista et al. [2] combined it with Edited Nearest Neighbours (ENN) to remove noisy samples near class boundaries. Gradient-boosted trees such as XGBoost [3] and LightGBM [4] are widely used for tabular fraud classification. For evaluating models on highly imbalanced data, Saito and Rehmsmeier [5] showed that precision–recall measures, including Average Precision, are more informative than ROC-AUC.

For model interpretability, Lundberg and Lee [6] introduced SHAP, and Lundberg et al. [7] described TreeSHAP for tree ensembles. McMahan et al. [8] proposed Federated Averaging (FedAvg), while Beutel et al. [9] introduced the Flower research framework. Alhasawi et al. [10] and Zhang et al. [11] report federated approaches to fraud detection. Those studies motivate the project's future work, but their settings are distinct from the centralized experiments reported here.

This study has three practical aims. First, it evaluates PaySim and IEEE-CIS as separate benchmark domains rather than splitting one dataset into simulated institutional clients. Neither dataset establishes a multi-institutional experiment. Second, it connects local model explanations to a review interface while retaining the supplied meaning of anonymized fields. Third, it selects thresholds against an operational FPR ceiling and reports the held-out result even when it falls outside that target. No federated result is reported.

3. Methodology

FedGuard follows five stages, and we apply the same workflow independently to each dataset (Fig. 1).

3.1 Data audit and preprocessing

For each benchmark, labeled rows were stratified into train, validation, and test partitions in a 70/15/15 ratio using seed 42. This assignment was made before fitting any feature filter, imputer, encoder, scaler, sampler, or estimator. Preprocessing was fitted on the training partition and then applied unchanged to validation and test. Validation and test retained the original class prevalence.

PaySim produced 4,453,833 training rows (5,749 fraud), 954,394 validation rows (1,232 fraud), and 954,393 test rows (1,232 fraud). IEEE-CIS produced 413,378 training rows (14,464 fraud), 88,581 validation rows (3,100 fraud), and 88,581 test rows (3,099 fraud). These are separate stratified splits; no institution labels or cross-institution partitions were available.

PaySim predictors included transaction type, amount, oldbalanceOrg, newbalanceOrig, oldbalanceDest, and newbalanceDest, plus derived features: a destination-merchant indicator (destination name starts with M); origin and destination balance deltas (newbalanceOrig - oldbalanceOrg and newbalanceDest - oldbalanceDest); balance-error terms (newbalanceOrig + amount - oldbalanceOrg and oldbalanceDest + amount - newbalanceDest); an origin-zero-after-transaction indicator (newbalanceOrig = = 0); hour of day (step modulo 24); and log(1 + amount). Account identifiers, the target, the internal row index, and the supplied isFlaggedFraud rule flag were excluded from predictors. The final one-hot representation had 18 columns.

For IEEE-CIS, labeled transaction rows were left-joined to the identity table on TransactionID and given a has_identity indicator; TransactionID, the target, and the internal row index were excluded from predictors. Derived fields were transaction_hour = floor(TransactionDT / 3600) modulo 24, transaction_day = floor(TransactionDT / 86400), and log_amt = log(1 + TransactionAmt). A missingness filter calculated on training rows dropped 12 fields—dist2, D7, id_07, id_08, id_18, id_21, id_22, id_23, id_24, id_25, id_26, and id_27—because each exceeded 90% missingness in training. Retained coded and anonymized fields kept their supplied names; their business meaning was not inferred. The transformed matrix contained 1,208 columns.

Numeric fields were median-imputed. Categorical fields used a missing-value token; low-cardinality fields were one-hot encoded with unseen validation/test categories ignored, and six high-cardinality IEEE-CIS fields (card1, card2, addr1, P_emaildomain, R_emaildomain, and DeviceInfo) were frequency encoded using training data. Missing indicators were added for 29 numeric D*/id_* fields. A StandardScaler with with_mean=False was fitted on training data for Logistic Regression only; tree models used the unscaled sparse CSR matrix. Transformed matrices were stored as float32. The accompanying feature inventory lists included, dropped, excluded, engineered, and transformed feature names.

3.2 Class-imbalance handling

Imbalance handling used training rows only. Class weighting left the row distribution unchanged and applied balanced weights in the estimator. For capped SMOTE, the majority class was randomly limited to at most 20 negatives per fraud case on PaySim and 5 negatives per fraud case on IEEE-CIS, then SMOTE generated minority examples to a minority-to-majority ratio of 0.5 (one fraud to two non-fraud cases). PaySim also used the same 20:1 cap followed by SMOTE-ENN, with Edited Nearest Neighbours using three neighbours. No sampler was applied to validation or test. The main 195-fit experiment included three strategies on PaySim and two on IEEE-CIS; a separate 39-fit IEEE-CIS SMOTE-ENN experiment was validation-only and is reported as exploratory.

3.3 Centralized model exploration and selection

We compared Logistic Regression, Random Forest, XGBoost, and LightGBM. For each model family and imbalance strategy, candidates were sampled with scikit-learn ParameterSampler using seed 42. Budgets were 3 Logistic Regression, 6 Random Forest, 15 XGBoost, and 15 LightGBM candidates per strategy, or 39 fits per strategy. PaySim's three strategies produced 117 fits; IEEE-CIS's two main strategies produced 78; together they total 195. Within each family/strategy pair, the candidate with the highest validation AP was

retained. The final model was the retained candidate with the highest validation AP across the main comparisons. The per-family validation table reports each family's strongest strategy and validation metrics; it does not report test scores. Exact validation-winning parameter values are listed below.

Logistic Regression sampled C from a log-uniform distribution between 0.001 and 100. It used the liblinear solver, max_iter = 2,000, seed 42, and balanced class weights only in the class-weight strategy.

Random Forest sampled n_estimators from {100, 150, 200}, max_depth from {12, 16, 20}, min_samples_leaf from {1, 2, 5}, and max_features from {sqrt, 0.5}. It used balanced_subsample weights for the class-weight strategy, seed 42, and all available logical CPU threads.

XGBoost sampled n_estimators from {200, 350, 500}, max_depth from {3, 5, 7}, learning_rate log-uniformly from 0.02 to 0.2, subsample from {0.75, 0.9, 1.0}, colsample_bytree from {0.7, 0.9, 1.0}, min_child_weight from {1, 5, 10}, and reg_lambda log-uniformly from 0.1 to 10. It used binary:logistic, aucpr evaluation, histogram tree construction, seed 42, and scale_pos_weight equal to the training negative-to-positive ratio for class weighting. The recorded run used CUDA for XGBoost.

LightGBM sampled n_estimators from {200, 350, 500}, num_leaves from {15, 31, 63}, max_depth from {-1, 8, 12}, learning_rate log-uniformly from 0.02 to 0.2, subsample from {0.75, 0.9, 1.0}, colsample_bytree from {0.7, 0.9, 1.0}, min_child_samples from {20, 50, 100}, and reg_lambda log-uniformly from 0.1 to 10. It used the binary objective, balanced class weights for that strategy, seed 42, and all available logical CPU threads.

3.3.1 Validation-winning parameter values

PaySim: Logistic Regression (capped SMOTE: C = 4.5705630998014515, solver=liblinear); Random Forest (capped SMOTE: n_estimators = 150, max_depth = 16, min_samples_leaf = 1, max_features=sqrt); XGBoost (capped SMOTE: n_estimators = 350, max_depth = 5, learning_rate = 0.04921421964961083, subsample = 0.9, colsample_bytree = 0.9, min_child_weight = 5, reg_lambda = 1.0968217207529523); LightGBM (class weights: n_estimators = 500, num_leaves = 15, max_depth = 8, learning_rate = 0.07804189087710861, subsample = 1.0, colsample_bytree = 1.0, min_child_samples = 100, reg_lambda = 0.5832486597608726).

IEEE-CIS: Logistic Regression (class weights: C = 4.5705630998014515, solver=liblinear); Random Forest (class weights: n_estimators = 100, max_depth = 20, min_samples_leaf = 2, max_features = 0.5); XGBoost (class weights: n_estimators = 500, max_depth = 7, learning_rate = 0.06695833655111026, subsample = 1.0, colsample_bytree = 0.7, min_child_weight = 5, reg_lambda = 0.8168455894760165); LightGBM (class weights: n_estimators = 500, num_leaves = 63, max_depth = 8, learning_rate = 0.084009446334044, subsample = 0.75, colsample_bytree = 0.7, min_child_samples = 20, reg_lambda = 2.1761183209050428).

3.4 Explainability layer

The selected LightGBM models are explained with TreeSHAP using tree_path_dependent perturbation and model_output="raw". The expected value plus the sum of all feature contributions reconstructs the model raw output (boosting margin/log-odds) within numerical tolerance. This additivity check does not mean the

contributions sum to a probability. The dashboard ranks strong local contributions and phrases their direction as evidence for review; explanations do not establish causation, fairness, or proof of fraud.

3.5 Inference and dashboard integration

A shared inference interface loads the dataset-specific model, fitted preprocessor, threshold, and feature metadata and returns an uncalibrated score, a threshold decision, and a local explanation when available. The local Streamlit dashboard presents separate PaySim and IEEE-CIS workflows. The repository's 55-case automated software suite checks code and interface behavior, including pipeline, artifact, inference, explanation, and dashboard contracts. Passing software tests would not establish predictive validity, generalization, privacy protection, or deployment readiness.

4. Experimental Setup

4.1 Datasets

PaySim [12] is a synthetic agent-based mobile-money benchmark. IEEE-CIS [13] is a card-not-present e-commerce benchmark released through the IEEE-CIS Kaggle competition, with transaction and identity tables and many coded or anonymized fields. They represent separate benchmark domains, not two identified financial institutions, and do not support a claim of multi-institutional evaluation. Table 1 summarizes the data and split sizes.

Table 1
Dataset characteristics.

Property	PaySim	IEEE-CIS
Nature	Synthetic mobile-money benchmark	Anonymized card-not-present e-commerce benchmark
Labeled transactions	6,362,620	590,540
Fraud prevalence	0.1291%	3.499%
Train rows (fraud)	4,453,833 (5,749)	413,378 (14,464)
Validation rows (fraud)	954,394 (1,232)	88,581 (3,100)
Test rows (fraud)	954,393 (1,232)	88,581 (3,099)
Transformed predictors	18	1,208
Main candidate fits	117	78

PaySim includes post-transaction balance features. For that reason, a model using these features is evaluated as a retrospective analysis of completed transactions, not as a scorer for pre-authorization decisions. We also recognize that the dataset is synthetic, which limits how far the results can be generalized.

4.2 Baselines and evaluation protocol

The comparison included Logistic Regression, Random Forest, XGBoost, and LightGBM with dataset-specific imbalance strategies. Average Precision (AP), computed from validation scores, was the primary model-selection measure. For each family's strongest validation configuration, the operating threshold was selected from unique validation score values using score > = threshold as the positive rule. We chose the threshold with the highest recall among those with validation FPR < = 0.01; if recall tied, the first eligible threshold in descending score order was retained. If no threshold met the ceiling, the implementation returned the next representable value above the maximum score. The final family and threshold were chosen without test data. Only the final selected model for each dataset was then evaluated once on its held-out test split. On the held-out test split, we report Average Precision (AP), ROC-AUC, precision, recall, F1-score, and False Positive Rate (FPR) (Table 3), together with the four confusion counts (Table 4).

4.3 Software environment

The captured training run used Python 3.11.5 on Windows 10 with an Intel Core i5-14400F, 32 GiB RAM, and an NVIDIA GeForce RTX 4060 with 8 GiB memory. Recorded package versions were NumPy 2.4.6, pandas 3.0.6, SciPy 1.17.1, scikit-learn 1.9.1, imbalanced-learn 0.14.2, XGBoost 3.2.0, LightGBM 4.7.0, SHAP 0.51.0, Streamlit 1.66.0, and Weights & Biases 0.31.0. Seed 42 was used for splitting, sampling, and estimator randomness. Logistic Regression, Random Forest, and LightGBM ran on CPU; XGBoost used CUDA in the recorded runs. The accompanying pinned requirements snapshot records the captured Python package environment.

5. Results

Table 2 presents the validation-only comparison for the best candidate in each model family. AP ranks model scores independently of a threshold; precision, recall, and FPR use a family-specific validation threshold selected under the same 1% FPR rule. These rows summarize the family-level results across the 195 candidate fits; they are not a row-by-row ledger of every fit and do not call for another test evaluation. LightGBM with class weighting had the highest validation AP on both benchmarks. For the final selected LightGBM models, validation-selected thresholds were 0.0568012 on PaySim and 0.7675237 on IEEE-CIS. Table 3 reports their held-out test results, and Table 4 gives the corresponding confusion counts.

Table 2
Validation-only comparison by model family and benchmark domain.

Dataset	Model	Strategy	AP	ROC-AUC	Precision	Recall	FPR (%)
PaySim	LightGBM	Class weights	0.999986	1.000000	0.983240	1.000000	0.0022
PaySim	XGBoost	Capped SMOTE	0.999970	1.000000	0.968553	1.000000	0.0042
PaySim	Random Forest	Capped SMOTE	0.999909	1.000000	0.895349	1.000000	0.0151
PaySim	Logistic Regression	Capped SMOTE	0.636580	0.996018	0.106539	0.909903	0.9863
IEEE-CIS	LightGBM	Class weights	0.758107	0.960109	0.712744	0.681935	0.9967
IEEE-CIS	XGBoost	Class weights	0.729666	0.953336	0.704333	0.655484	0.9979
IEEE-CIS	Random Forest	Class weights	0.635609	0.937811	0.671791	0.563871	0.9991
IEEE-CIS	Logistic Regression	Class weights	0.469064	0.878558	0.584841	0.385806	0.9932

A separate IEEE-CIS SMOTE-ENN run comprised 39 validation-only candidates and was not part of the 195-fit main comparison. Its best AP values were 0.429363 (Logistic Regression), 0.598264 (Random Forest), 0.651665 (XGBoost), and 0.692170 (LightGBM), all below the corresponding class-weight results; its test split was not opened.

Table 3
Held-out test performance of the selected LightGBM models (class weighting).

Dataset	Model	Threshold	AP	Precision	Recall	F1	FPR (%)	ROC-AUC
PaySim	LightGBM (Class Wt)	0.0568	0.9980	0.9731	0.9976	0.9852	0.0036	0.9987
IEEE-CIS	LightGBM (Class Wt)	0.7675	0.7568	0.7102	0.6880	0.6989	1.0178	0.9591

Table 4
Held-out test confusion counts at the validation-selected thresholds.

Dataset	True positives	False negatives	False positives	True negatives
PaySim	1,229	3	34	953,127
IEEE-CIS	2,132	967	870	84,612

On PaySim, the held-out test contained 1,229 true positives, 3 false negatives, 34 false positives, and 953,127 true negatives. On IEEE-CIS, the held-out test FPR was 1.0178%, slightly above the validation ceiling; the model detected 2,132 of 3,099 fraud cases, with 84,612 true negatives. The validation target is not guaranteed to hold on a separate test split. The software suite checks application and pipeline behavior; it is not a test of predictive performance.

6. Discussion

The results differ between the two benchmark domains. PaySim's near-saturated scores require caution: the data are simulated, fraud patterns are comparatively regular, and post-transaction balance fields reveal information unavailable before authorization. IEEE-CIS gives a harder retrospective benchmark result. At a validation-selected threshold, test recall was 68.80% and FPR was 1.0178%, leaving a substantial share of fraud undetected under an alert rate near the chosen ceiling. Neither dataset supports an inference about performance across actual institutions.

The IEEE-CIS test FPR was 1.0178%, slightly above the 1% target measured on validation data. We report the observed test value without changing the threshold. Retuning on test data would change the evaluation protocol. A future study could assess multiple splits and thresholds with a predeclared safety margin while retaining a separate final test set.

We use TreeSHAP as a local aid for human review. For the selected LightGBM model, the base value and feature contributions sum to the raw model output (margin/log-odds) within tolerance; this sum is not a probability. Features that encode historical disparities may still influence model output, and SHAP does not guarantee fairness. The explanations indicate model behavior for one input and do not establish causation or prove fraud.

Limitations. (i) The system is centralized and pools data for training, so this work provides no privacy guarantee; (ii) PaySim is synthetic and includes post-transaction features; (iii) the results are based on one stratified split per dataset and include no confidence intervals; (iv) we do not yet report a federated or non-federated cross-institution comparison; and (v) the dashboard is a local demonstration rather than a production service.

7. Conclusion and Future Work

This paper reports a centralized, explainable baseline on two separate fraud benchmark domains. The selected LightGBM models achieved held-out test AP of 0.997968 on PaySim and 0.756757 on IEEE-CIS, with the measured IEEE-CIS FPR slightly above the 1% validation target. The results follow a single stratified split per dataset and do not establish federated or multi-institutional performance. The model comparison, split counts, selected thresholds, feature inventory, and captured software environment are documented in the paper and project repository to support review and reproduction.

Future work may study decentralized training with Flower and FedAvg, differential privacy, secure aggregation, temporal feature construction, and a production-oriented service. These are plans rather than

components of the current baseline. Table 5 separates implemented work from planned scope.

Table 5
Implemented work and planned future scope.

Implemented (this paper)	Future scope (planned)
PaySim and IEEE-CIS pipelines	Federated learning (Flower / FedAvg)
Train-only preprocessing; held-out validation/test	No centralized data pooling
Class weighting and capped SMOTE / SMOTE-ENN	Differential privacy (Opacus)
LightGBM selected by AP, validation FPR ≤ 1%	Secure aggregation (HE / SMPC)
TreeSHAP plain-language audit reasons	Real-time stream feature engineering
Streamlit dashboard and shared inference API	FastAPI backend and React UI

Declarations

Competing interests: The authors declare that they have no competing interests.

Funding: No specific funding was received for this work.

Author contributions: Naman Dugar contributed to data audits, dataset splitting, model training and comparison, threshold selection, held-out test evaluation, and SHAP visualisation. Anvi Tyagi developed the shared inference interface, integrated the system with Streamlit, performed artifact checks, and contributed to demonstration hardening. Riyanshu developed the TreeSHAP explainer, evidence-based audit reasons, reusable explanation view, and documentation. Prof. Mohit Tiwari provided academic supervision and contributed to the critical review and revision of the manuscript. All authors contributed to the preparation and revision of the manuscript, read and approved the final version, and agree to be accountable for all aspects of the work.

Data availability and reproduction: Source code, scripts, tests, reports, and documentation are available in the project repository at https://github.com/End-2007/Privacy-Preserving-Multi-Institutional-Financial-Fraud-Detection. PaySim can be obtained from https://www.kaggle.com/datasets/ealaxi/paysim1; IEEE-CIS training files can be obtained from https://www.kaggle.com/competitions/ieee-fraud-detection/data. Access may require a Kaggle account and acceptance of dataset terms. Download the PaySim CSV and IEEE-CIS train_transaction.csv and train_identity.csv into data/raw/ using the filenames in data/README.md. Install the pinned environment in the accompanying requirements file, then run python scripts/prepare_stage1.py, python scripts/train_stage1.py --dataset paysim --no-wandb, and python scripts/train_stage1.py --dataset ieee_cis --no-wandb from the repository root. The training commands write comparison and selected-model reports and evaluate each selected model once on its test split. Attached CSVs provide validation family comparisons and feature lists. Large source data and fitted artifacts are not included in the repository.

Ethics approval and consent to participate: Not applicable. This study uses only previously collected, publicly available benchmark datasets and involves no human participants or animals.

Consent for publication: Not applicable.

Supplementary files: FedGuard_Supplementary_Validation_Results.csv; FedGuard_Supplementary_Strategy_Validation_Results.csv; FedGuard_IEEE_CIS_SMOTE_ENN_Validation_Candidates.csv; FedGuard_Feature_Inventory.csv; FedGuard_Training_Environment_Requirements.txt; FedGuard_Reproduction_Instructions.txt; ieee_cis_metrics.json; and paysim_metrics.json.

The manuscript includes Figure 1 and Tables 1–5.

References

1. Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP (2002) SMOTE: Synthetic Minority Over-sampling Technique. J Artif Intell Res 16:321–357. https://doi.org/10.1613/jair.953
2. Batista GEAPA, Prati RC, Monard MC (2004) A Study of the Behavior of Several Methods for Balancing Machine Learning Training Data. ACM SIGKDD Explorations Newsl 6(1):20–29. https://doi.org/10.1145/1007730.1007735
3. Chen T, Guestrin C XGBoost: A Scalable Tree Boosting System, in Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 785–794. https://doi.org/10.1145/2939672.2939785
4. Ke G, Meng Q, Finley T, Wang T, Chen W, Ma W, Ye Q, Liu T-Y (2017) LightGBM: A Highly Efficient Gradient Boosting Decision Tree, in Advances in Neural Information Processing Systems, vol. 30, pp. 3146–3154. https://proceedings.neurips.cc/paper_files/paper/2017/hash/6449f44a102fde848669bdd9eb6b76fa-Abstract.html
5. Saito T, Rehmsmeier M (2015) The Precision-Recall Plot Is More Informative than the ROC Plot When Evaluating Binary Classifiers on Imbalanced Datasets, PLOS ONE, vol. 10, no. 3, e0118432. https://doi.org/10.1371/journal.pone.0118432
6. Lundberg SM, Lee S-I (2017) A Unified Approach to Interpreting Model Predictions, in Advances in Neural Information Processing Systems, vol. 30, pp. 4765–4774. https://proceedings.neurips.cc/paper_files/paper/2017/hash/8a20a8621978632d76c43dfd28b67767-Abstract.html
7. Lundberg SM, Erion G, Chen H, DeGrave A, Prutkin JM, Nair B, Katz R, Himmelfarb J, Bansal N, Lee S-I (2020) From Local Explanations to Global Understanding with Explainable AI for Trees. Nat Mach Intell 2:56–67. https://doi.org/10.1038/s42256-019-0138-9
8. McMahan HB, Moore E, Ramage D, Hampson S, Agüera B y, Arcas (2017) Communication-Efficient Learning of Deep Networks from Decentralized Data, in Proceedings of the 20th International

Conference on Artificial Intelligence and Statistics, Proceedings of Machine Learning Research, vol. 54, pp. 1273–1282. https://proceedings.mlr.press/v54/mcmahan17a.html

9. Beutel DJ, Topal T, Mathur A, Qiu X, Fernandez-Marques J, Gao Y, Sani L, Li KH, Parcollet T, de Gusmão PPB, Lane ND (2020) Flower: A Friendly Federated Learning Research Framework. https://doi.org/10.48550/arXiv.2007.14390. arXiv:2007.14390
10. Alhasawi Y, Almtrf AA, Asad M (2025) A Federated Approach to Scalable and Trustworthy Financial Fraud Detection. Secur Priv 8(5):e70099. https://doi.org/10.1002/spy2.70099
11. Zhang Z, Liu Z, Li X, Zhang L (2026) HiFraud: Hierarchical Privacy-Preserving Federated Learning with Star-Chain Knowledge Transfer for Cross-Institutional Fraud Detection, Computers, Materials & Continua, vol. 88, no. 2, art. 34. https://doi.org/10.32604/cmc.2026.081922
12. Lopez-Rojas EA, Elmir A, Axelsson S (2016) PaySim: A Financial Mobile Money Simulator for Fraud Detection, in Proceedings of the 28th European Modeling and Simulation Symposium, pp. 249–255. https://www.msc-les.org/proceedings/emss/2016/EMSS2016_249.pdf
13. IEEE Computational Intelligence Societyand, Corporation V, Fraud Detection IEEE-CIS (2019) Kaggle competition dataset, https://www.kaggle.com/competitions/ieee-fraud-detection/data

Figures

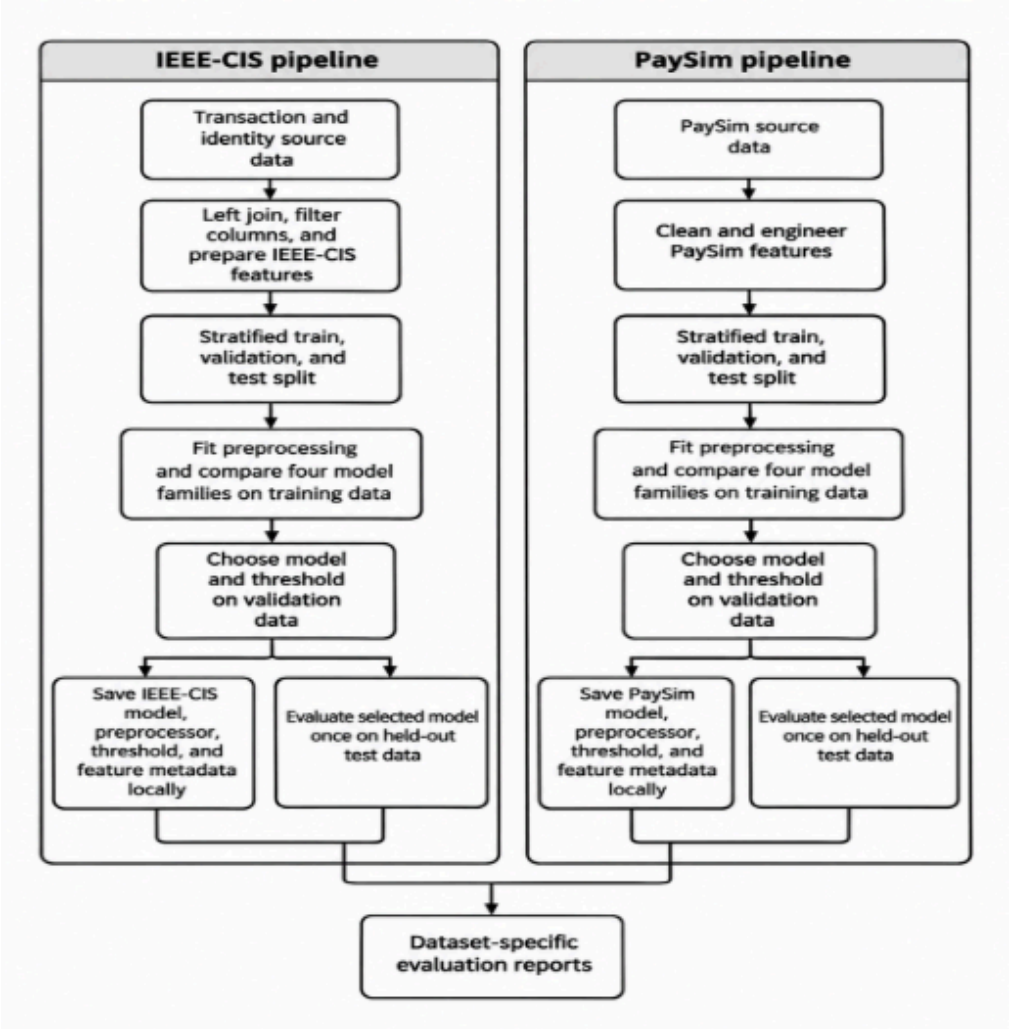


Figure 1

FedGuard pipelines run independently for each dataset. Each dataset is split, preprocessed, modelled and thresholded on its own; the selected model is then evaluated once on its held-out test data.

Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- FedGuardFeatureInventory.csv
- FedGuardIEEECISSMOTEENNValidationCandidates.csv
- FedGuardReproductionInstructions.txt
- FedGuardSupplementaryStrategyValidationResults.csv
- FedGuardSupplementaryValidationResults.csv
- FedGuardTrainingEnvironmentRequirements.txt
- ieeecismetrics.json
- paysimmetrics.json