



International Journal of Artificial Intelligence and Machine Learning

Publisher's Home Page: <https://www.svedbergopen.com/>



Research Paper

Open Access

An Imbalance-Sensitive Multi-Sampling Ensemble Framework with Validation-Driven Threshold Optimization for Banking Transaction Fraud Detection

Neha Vyas¹, Ruby Bhatt²

¹Research Scholar, Computer Science, Medicaps University, Indore, Madhya Pradesh, 452009. E-Mail: neha27upadhyay@gmail.com

²Associate Professor, Computer Science, Medicaps University, Indore, Madhya Pradesh, 452009.

E-Mail: profrubybhatt15@gmail.com, ORCID: 0000-0003-1872-8828

Abstract

Banking transaction fraud detection remains difficult because fraudulent events are rare, behaviourally heterogeneous and continuously evolving. Two methodological weaknesses recur in the literature: preprocessing or resampling statistics estimated on the full dataset, which leaks information into the evaluation partitions, and a fixed decision threshold of 0.50, which is rarely optimal under extreme skew. This paper presents an imbalance-sensitive multi-sampling ensemble framework with validation-driven threshold optimization that addresses both. A corpus of 240,000 banking transactions containing 4,782 fraudulent records (1.99%) is partitioned by stratified sampling into 70% training, 15% validation and 15% test subsets. Every imputation, encoding and scaling parameter is estimated on the training subset alone and applied unchanged to the remaining partitions, and all resampling is confined to the training subset. Seven resampling operators are crossed with five ensemble classifiers to yield 35 controlled configurations. For each configuration a decision threshold is selected by maximizing the validation F₁-score over $\tau \in \{0.01, 0.02, \dots, 0.99\}$ and is then applied unchanged to the held-out test subset. Performance is reported through precision, recall, F₁-score, ROC-AUC, PR-AUC and confusion-matrix analysis. Random Forest and Extra Trees are the most robust learners across all seven operators; the ADASYN-Extra Trees configuration recovers all 717 fraudulent test transactions with no false positives at a selected threshold of 0.6983. The framework offers a reproducible and auditable protocol for highly imbalanced financial fraud screening. The implementation is available at <https://github.com/neha27upadhyay-cmyk/Fraud-Detection-using-ML>

Keyword: Banking transaction fraud detection, class imbalance, data leakage, decision-threshold optimization, ensemble learning, PR-AUC, resampling, SMOTE.

I. INTRODUCTION

Digital banking, electronic commerce and mobile payment services have driven a sustained increase in the volume, velocity and variety of card transactions. The same infrastructure that makes payment convenient also widens the attack surface available to fraudsters. Fraudulent transactions impose direct monetary loss, operational disruption, reputational damage and regulatory exposure, and they are deliberately designed to resemble legitimate activity. An effective detection system must therefore recognise unusual and drifting behavioural patterns while keeping the false-alarm rate low enough that manual review remains economically viable.

Rule-based screening, transaction ceilings and expert-curated blacklists are transparent and easy to deploy, but they are static: they cannot represent nonlinear feature interactions, they do not generalise to previously unseen fraud patterns, and they degrade as customer behaviour evolves. Machine learning and deep learning have consequently become the dominant paradigm. Optimized deep event-based networks have been shown to outperform conventional classifiers on benchmark card-fraud corpora [1], and predictive quality has been improved further by representation learning combined with class balancing, for example through redundancy reduction, feature boosting and spiral oversampling [2]. Graph neural networks and autoencoders have been used to learn relational and latent transaction structure for real-time prevention [3], reinforcement-learning subset selection has been used to close the distribution gap between heterogeneous source and target domains [4], and Kolmogorov-Arnold networks have been offered as a parameter-efficient and more interpretable alternative to multilayer perceptrons [5].

Behavioural and temporal modelling forms a second research thread. A hierarchical gated network has been used to learn feature-level and transaction-level behavioural information jointly [6], which matters because fraud that appears innocuous in isolation often becomes visible once sequential context is supplied. Because fraud patterns drift, static detectors lose accuracy over time; a combination of feature

selection, ADASYN, convolutional learning and adaptive drift detection has been shown to handle both gradual and abrupt changes in transaction behaviour [7].

Class imbalance remains the single largest obstacle. In the corpus used here only 4,782 of 240,000 transactions are fraudulent, a prevalence of 1.99%. A classifier that predicts the majority class unconditionally attains 98.01% accuracy while detecting nothing, so accuracy alone cannot support model selection. Precision, recall, F₁-score, ROC-AUC and, in particular, PR-AUC are required to characterise minority-class sensitivity, alert reliability and ranking quality. The severity of the skew in this corpus is itself the justification for a comparative study of resampling operators rather than the assumption that one balancing technique suffices.

An extensive family of resampling operators has been developed in response. Random Over-Sampling duplicates minority records; SMOTE and ADASYN synthesise new minority records by interpolation; Borderline-SMOTE concentrates synthesis near the decision boundary; Random Under-Sampling discards majority records; and hybrid schemes such as SMOTEENN and SMOTETomek combine synthesis with neighbourhood cleaning. A Mahalanobis-distance hybrid sampler paired with Random Forest has been shown to address imbalance and noisy class boundaries simultaneously [8]. The relative merit of any operator, however, depends jointly on the geometry of the dataset and on the inductive bias of the downstream learner, which argues for systematic comparison rather than for the adoption of a single balancing method.

Robustness is a further concern. Reinforcement-learning adversarial attacks demonstrate that a motivated fraudster can adapt behaviour to evade a trained detector [9], so strong performance on a static test partition does not by itself imply resilience. Feature selection and class balancing improve generalisation only partially: ADASYN combined with recursive feature elimination and cross-validation yields substantial gains for Random Forest, XGBoost and LightGBM [10], but such gains are overstated whenever resampling or preprocessing precedes partitioning.

That last point identifies the first of two methodological gaps this paper targets. Imputation, encoding, scaling, feature selection and synthetic sampling are frequently applied to the entire dataset before the data are split. Statistics derived from validation and test observations then enter the model indirectly, and synthetic or duplicated records may appear on both sides of the partition boundary. The resulting performance estimates are optimistic and are not reproducible in deployment. A defensible protocol estimates every preprocessing parameter from the training subset alone, confines resampling to the training subset, uses the validation subset only for configuration choices, and reserves the test subset for a single final measurement.

The second gap is the default classification threshold of 0.50. When the positive class is rare and the costs of a false positive and a false negative differ sharply, that value is arbitrary. Lowering the threshold raises recall at the cost of investigator workload; raising it suppresses alerts at the cost of undetected fraud. Ensemble learning with Random Under-Sampling and two-stage thresholding has already shown that decision-boundary adjustment produces measurable gains [12]. It follows that the threshold should be treated as a parameter to be selected on validation data, separately for each model.

Hybrid deep architectures have also been proposed to represent complex fraud behaviour; a CNN-LSTM model with an attention mechanism combines spatial extraction, sequential modelling and relevance weighting [11]. Such models can be accurate but are computationally heavier and less interpretable than tree ensembles. For structured banking data, Random Forest, Extra Trees, XGBoost, LightGBM and CatBoost remain attractive because they model nonlinear interactions efficiently and emit calibrated probability scores that are directly amenable to threshold optimization.

This study therefore proposes a leakage-aware, imbalance-sensitive framework for banking transaction fraud detection. The pipeline comprises identifier removal, median imputation, categorical encoding, temporal feature extraction, numerical standardization, stratified partitioning, training-only resampling, ensemble model development, validation-driven threshold optimization and a single final test evaluation. Seven resampling operators are crossed with five ensemble classifiers to give 35 configurations, each evaluated under identical partitions, hyperparameters and random seed. The principal contributions are as follows.

- 1) A unified factorial protocol that crosses seven resampling operators with five ensemble classifiers, producing 35 controlled configurations that are directly comparable because partitions, hyperparameters and seed are held constant.
- 2) A leakage-free preprocessing and resampling protocol in which imputation, encoding, scaling and class balancing are fitted on the training subset alone and applied unchanged to the validation and test subsets.
- 3) A configuration-specific decision threshold, selected by maximizing the validation F₁-score over a 99-point grid and transferred unchanged to the test subset, in place of the conventional fixed value of 0.50.
- 4) An evaluation protocol centred on precision, recall, F₁-score, PR-AUC and confusion-matrix analysis rather than on accuracy or ROC-AUC, both of which are optimistic under a 1.99% positive rate.

- 5) A direct comparison of bagging and boosting ensembles under identical balancing conditions, which isolates the interaction between the sampling operator and the learner.
- 6) A deployment-oriented reading of the results that considers resampled training-set size and computational cost alongside predictive quality.

The remainder of the paper is organised as follows. Section II reviews related work. Section III develops the proposed framework, covering problem formulation, architecture, leakage-free data preparation, class-imbalance handling, ensemble learning, threshold optimization and evaluation metrics. Section IV describes the implementation and experimental setup. Section V reports and discusses the results. Section VI concludes and identifies future work.

II. RELATED WORK

A. Deep and Representation Learning for Fraud Detection

Adil et al. introduced OptDevNet, an optimized deep event-based network that improves on conventional classifiers evaluated on limited benchmark sets [1]. Ni et al. combined feature redundancy reduction, feature boosting and spiral oversampling to strengthen minority-class representation and discriminative learning [2]. Alarfaj and Shahzadi demonstrated that graph neural networks and autoencoders can capture relational and latent transaction patterns for real-time prevention [3]. Zhu et al. addressed domain discrepancy in heterogeneous transaction environments through reinforcement-learning source-instance selection [4]. Le et al. reported improved accuracy and parameter interpretability from efficient KolmogorovArnold networks relative to multilayer perceptrons [5].

B. Behavioural and Temporal Modelling

Xie et al. constructed a hierarchical gated network that learns feature-oriented and transaction-oriented behavioural information jointly [6]. Lawati et al. integrated feature selection, ADASYN, convolutional learning and adaptive drift detection to track evolving fraud patterns [7]. A spatial-temporal gated network has been used to capture correlations of time and location across consecutive transactions [13], and time-dependent recurrent learning with attention has been used to estimate behavioural periodicity from transaction histories [14]. Goudar et al. addressed concept drift and catastrophic forgetting through adaptive bagging over temporally separated fraud datasets [42].

C. Class Imbalance, Ensembles and Cost Sensitivity

Xie and Huang combined Mahalanobis-distance SMOTEENN with Random Forest to learn from imbalanced and noisy transaction data [8]. Ileberi and Sun reported substantial gains from ADASYN with recursive feature elimination and cross-validation across XGBoost, Random Forest and LightGBM [10], and separately analysed cost-sensitive oversampling that prioritises financially significant fraudulent transactions instead of treating all minority samples alike [17]. Almubark eliminated outliers and improved minority detection with a sampling-based ensemble and two-stage thresholding [12]. Zhu et al. proposed noisy-sample-removed undersampling to discard uncertain observations during majority reduction [21]. Zhao et al. integrated cost awareness into LightGBM to reduce the dilution of false-negative penalties under extreme imbalance [23]. Sulimani directed sequential learners towards repeatedly misclassified transactions through an AdaBoost-LSTM hybrid [20], and Nama and Banu used adaptive generative synthetic sampling to preserve dense minority structure while suppressing noise [25].

D. Federated, Explainable and Optimization-Based Approaches

Aurna et al. proposed FedFusion for institutions whose feature spaces differ [15], and Wu et al. jointly addressed client drift and membership leakage in a federated setting [16]. Manonmani et al. combined encrypted aggregation with adaptive gradient clipping for robustness under non-IID data [37]. Ghalwash et al. increased recall and reduced false negatives through DBSCAN-augmented disjunctive voting [29], and Abbasimehr and Fanai paired a Bayesian-optimized supervised autoencoder with blending ensembles [30]. Abualhaj et al. jointly optimized features and class balance using multi-objective Harris Hawks optimization with K-means SMOTE [31], while Padhi et al. combined osprey-optimization feature selection with stacking ensembles [35]. Ni et al. modelled relational structure through hierarchical sampling, metric-based graph construction and adversarial GraphSAGE [33]. Awad et al. linked ensemble predictions to interpretable explanations using SHAP [36], Zavvar et al. combined SMOTE, autoencoders, convolutional layers and attention to trade sensitivity against precision [38], and Chekira et al. evaluated learned aggregation rules for ensemble combination [28]. Almaraz-Rivera et al. applied self-supervised tabular learning to card-not-present fraud [24], El Alami et al. showed that quantum classifiers are highly sensitive to feature-map and ansatz choice [40], and Çavdar and Bozanta confirmed that Random Forest and tuned XGBoost deliver strong accuracy, stability and explainability on real-world data [41]. RugelesDiaz et al. argued for merchant-specific modelling with continuous retraining [39], and Alrasheedi found Random Forest, decision trees and multilayer perceptrons competitive on both balanced and imbalanced data [34].

E. Surveys and Research Gap

Recent surveys converge on a consistent set of open problems. Mienye and Jere reviewed CNN, RNN, LSTM and GRU methods and the metrics appropriate to each [18]; Kalid et al. observed that most studies address class imbalance but underestimate class overlap [19]; Dastidar et al. identified data scarcity, behavioural change, skewed distributions and the need for contextual representation as central gaps [22]; Arora and Bhardwaj reviewed methods, testing environments and deployment challenges [26]; Gamal et al. showed that balanced machine and deep learners combined with explanation mechanisms achieve high detection rates [27]; and Singh et al. benchmarked diverse classifiers across multiple datasets, reporting strong but computationally expensive performance from advanced ensemble and tabular models [32].

Three gaps persist. First, most studies examine a small number of samplers or a small number of learners, so it is not possible to attribute an observed gain to the balancing strategy, to the learning algorithm, or to their interaction. Second, leakage controls are frequently unstated, and preprocessing or resampling is not always demonstrably separated from the evaluation partitions. Third, a fixed threshold of 0.50 is commonly assumed even though the optimal operating point on the precision–recall curve differs from model to model, and accuracy or ROC-AUC is often emphasised despite being optimistic under extreme skew. The framework developed in Section III addresses all three within a single protocol.

III. PROPOSED FRAMEWORK

This section develops the proposed imbalance-sensitive framework. It begins with the problem statement and notation, presents the overall architecture, and then describes each stage in the order in which it executes: data preparation, class balancing, ensemble learning, threshold optimization and evaluation.

A. Problem Formulation and Notation

Let the banking transaction corpus be

$$\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N, \quad y_i \in \{0, 1\} \quad (1)$$

where x_i is the predictor vector of the i th transaction, $y_i = 1$ denotes fraud, $y_i = 0$ denotes a normal transaction and $N = 240,000$. The positive-class prior is $\pi = P(y = 1) \approx 0.0199$, so the corpus is skewed at approximately 49 : 1. The objective is to learn a scoring function $f : \mathbb{R}^d \rightarrow [0, 1]$ together with a decision threshold $\tau \in (0, 1)$ such that the induced classifier $\mathbb{P}[f(x) \geq \tau]$ maximises the harmonic mean of precision and recall on data never seen during any stage of model construction. Two constraints are imposed. First, no statistic derived from the validation or test partitions may influence any transformation, resampling operation, model fit or threshold choice. Second, τ is not fixed at 0.50 but is selected per configuration.

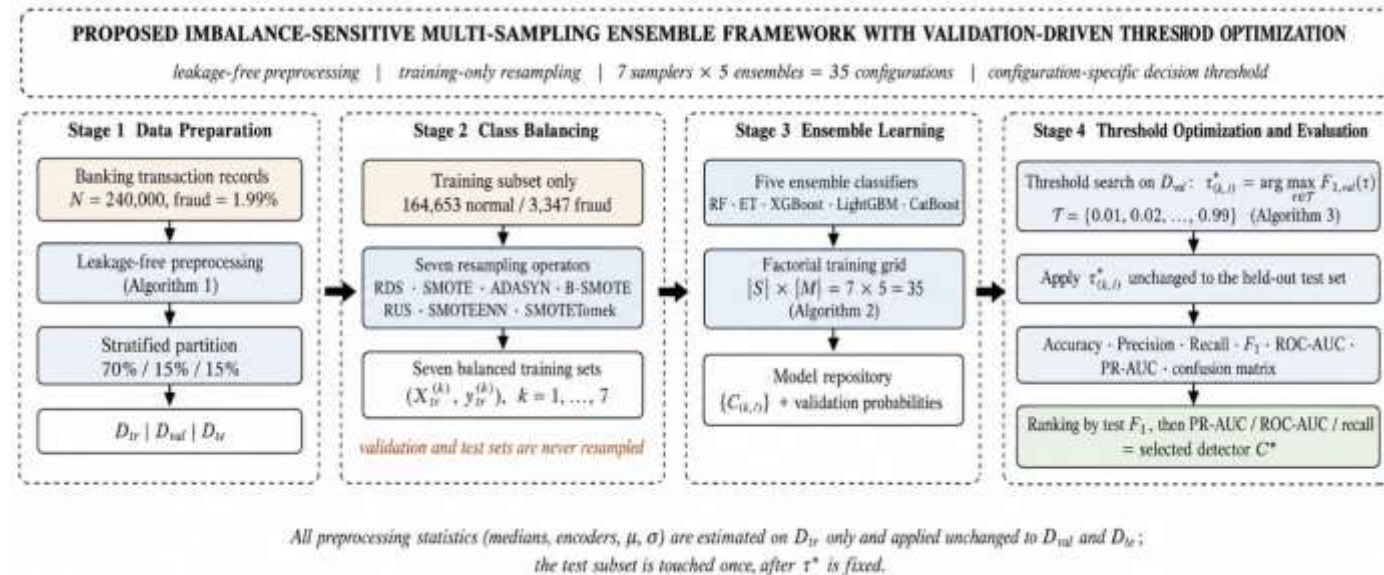


Fig. 1. Overall architecture of the proposed imbalance-sensitive multi-sampling ensemble framework. Stage 1 prepares and partitions the data, Stage 2 balances the training subset only, Stage 3 fits the 7×5 factorial grid of ensemble models, and Stage 4 selects a configuration-specific decision threshold on validation data before a single evaluation on the held-out test subset.

B. Overall Architecture

Fig. 1 shows the four stages of the framework. Stage 1 converts raw transaction records into a model-ready matrix and partitions them by stratified sampling. Stage 2 applies seven resampling operators to the training subset alone, producing seven balanced training distributions. Stage 3 fits five ensemble classifiers on each of those distributions, yielding 35 configurations. Stage 4 selects a decision threshold for every configuration on the validation subset, transfers it unchanged to the test subset, computes the evaluation metrics and ranks the configurations. The validation and test partitions are never resampled, and the test partition is scored exactly once, after every configuration choice has been fixed.

C. Dataset and Leakage-Free Preprocessing

The corpus contains 240,000 banking transactions, of which 4,782 are fraudulent. Table I lists every attribute together with the operation applied to it. Three identifier fields are removed because a learner can otherwise memorise individual records instead of learning generalisable fraud behaviour, and Country is excluded from the final predictor set. Date and time attributes are parsed and expanded into six temporal predictors and are then discarded. Missing numerical values are replaced by the training-set median, categorical attributes are label encoded, and all retained predictors are standardized. The binary target Fraud_Flag encodes normal transactions as 0 and fraudulent transactions as 1.

Fig. 2 shows the sequence. Missing numerical values are replaced by

$$x'_{ij} = x_{ij} \text{ if } x_{ij} \text{ is observed, else } m_j \quad (2)$$

where $m_j = \text{median}(X_{tr,j})$ is computed on the training subset. Each retained predictor is then standardized by

$$z_{ij} = (x'_{ij} - \mu^{tr}_j) / \sigma^{tr}_j \quad (3)$$

with μ^{tr}_j and σ^{tr}_j likewise estimated from the training subset only. Applying (2) and (3) with parameters learned anywhere else is precisely the leakage mechanism criticised in Section II-E. Algorithm 1 states the full procedure.

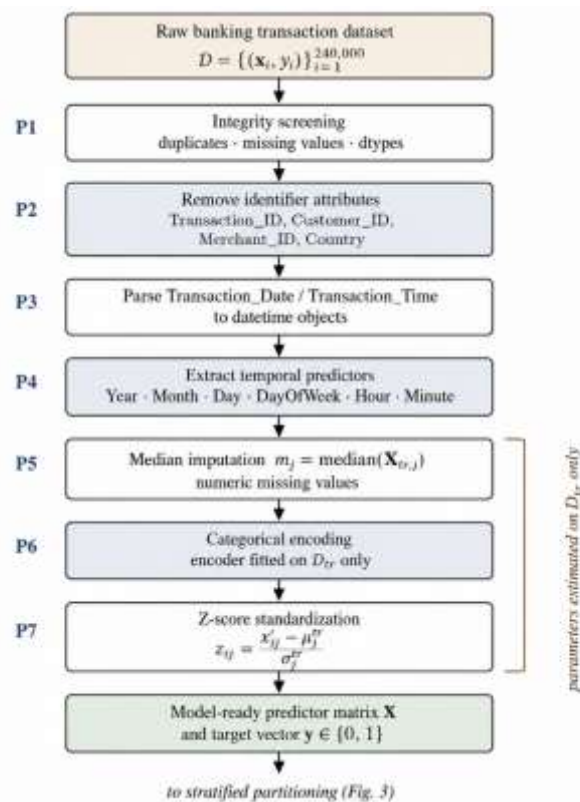


Fig. 2. Leakage-free preprocessing pipeline. Steps P5–P7 estimate their parameters on the training subset alone and apply them unchanged to the validation and test subsets.

Algorithm 1 Leakage-free preprocessing and stratified partitioning

Input: $\mathcal{D} = \{(x_i, y_i)\}, i = 1, \dots, N$ with $N = 240,000, y_i \in \{0, 1\}$.

Params: $r_{tr} = 0.70, r_{val} = 0.15, r_{te} = 0.15, \text{seed} = 42$.

Output: Processed $(X_{tr}, y_{tr}), (X_{val}, y_{val}), (X_{te}, y_{te})$ and the fitted preprocessing parameters.

1. Load the corpus and designate Fraud_Flag as the target.
2. Screen for duplicate records, missing values, invalid ranges and inconsistent data types.

3. Drop Transaction_ID, Customer_ID and Merchant_ID, which invite record memorization rather than pattern learning; drop Country.
4. Parse Transaction_Date and Transaction_Time to datetime objects.
5. Derive Year, Month, Day, DayOfWeek from the date and Hour, Minute from the time; discard the original date and time attributes.
6. Encode the target as $y_i = 1$ for fraud and $y_i = 0$ otherwise, and separate X from y .
7. Partition $\mathcal{D}$ by stratified sampling into mutually exclusive subsets in the ratio 70 : 15 : 15, preserving the fraud prevalence in each.
8. Identify numerical and categorical predictor groups from $\mathcal{D}_{tr}$ only.
9. Compute the training median m_j for every numerical feature and impute missing values with it.
10. Fit the categorical encoder on $\mathcal{D}_{tr}$ and apply it, without refitting, to all three subsets.
11. Compute μ^{tr}_j and σ^{tr}_j on the training subset and standardize every retained predictor with them.
12. Persist the medians, encoders, feature order, means and standard deviations for reproducible transformation of future transactions.
13. Return the three processed subsets together with the fitted parameters.

TABLE I. DATASET ATTRIBUTES AND THE PREPROCESSING APPLIED TO EACH

No.	Attribute	Category	Data type	Role in the study	Preprocessing applied
1	Transaction_ID	Identifier	Text	Unique transaction reference	Removed before training
2	Customer_ID	Identifier	Text	Customer associated with the transaction	Removed to prevent identity memorization
3	Merchant_ID	Identifier	Text	Merchant involved in the transaction	Removed before training
4	Transaction_Date	Temporal	Date	Calendar date of the transaction	Parsed to datetime; dropped after feature extraction
5	Transaction_Time	Temporal	Time	Execution time of the transaction	Parsed to datetime; dropped after feature extraction
6	Country	Geographical	Categorical	Country associated with the transaction	Excluded from the final predictor set
7	Year	Derived temporal	Integer	Transaction year	Extracted from Transaction_Date; standardized
8	Month	Derived temporal	Integer	Transaction month, 1–12	Extracted from Transaction_Date; standardized
9	Day	Derived temporal	Integer	Day of month	Extracted from Transaction_Date; standardized
10	DayOfWeek	Derived temporal	Integer	Weekday index, 0–6	Extracted from Transaction_Date; standardized
11	Hour	Derived temporal	Integer	Transaction hour, 0–23	Extracted from Transaction_Time; standardized
12	Minute	Derived temporal	Integer	Transaction minute, 0–59	Extracted from Transaction_Time; standardized
13	Numerical predictors	Transaction characteristics	Numerical	Quantitative properties of the transaction	Median imputation; Z-score standardization

14	Categorical predictors	Transaction characteristics	Categorical	Transaction, customer, merchant, payment and channel attributes	Label encoding fitted on training data; standardized
15	Fraud_Flag	Target	Binary	Indicates whether a transaction is normal or fraudulent	Encoded as normal = 0, fraud = 1

TABLE II. CLASS DISTRIBUTION ACROSS THE STRATIFIED PARTITIONS

Subset	Share (%)	Transactions	Normal	Fraud	Normal (%)	Fraud (%)	Intended use
Complete corpus	100	240,000	235,218	4,782	98.0075	1.9925	Full dataset
Training	70	168,000	164,653	3,347	98.0077	1.9923	Preprocessing fit, resampling, model training
Validation	15	36,000	35,282	718	98.0056	1.9944	Threshold selection and model configuration
Test	15	36,000	35,283	717	98.0083	1.9917	Single unbiased final evaluation

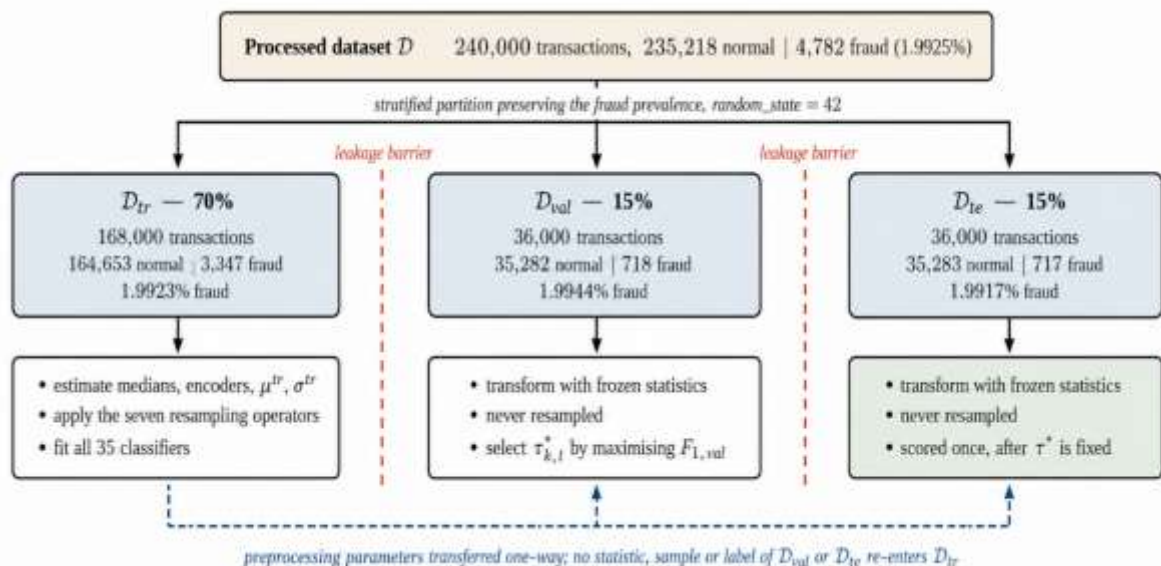


Fig. 3. Stratified partitioning and the leakage-prevention framework. The fraud prevalence is preserved to within 0.003 percentage points across the three subsets, and preprocessing parameters cross the leakage barriers in one direction only.

D. Stratified Partitioning and the Leakage Barrier

Fig. 3 shows the partitioning strategy and the controls that enforce separation. The corpus is divided into mutually exclusive subsets

$$\mathcal{D}_{tr} \cap \mathcal{D}_{val} = \mathcal{D}_{tr} \cap \mathcal{D}_{te} = \mathcal{D}_{val} \cap \mathcal{D}_{te} = \emptyset \quad (4)$$

in the ratio 70 : 15 : 15. Because the split is stratified, the fraud prevalence is preserved almost exactly in every subset, as Table II confirms: 1.9923% in training, 1.9944% in validation and 1.9917% in test against 1.9925% overall. The training subset supports preprocessing estimation, resampling and model fitting; the validation subset supports threshold selection only; the test subset is scored once. Preprocessing parameters cross the barrier in one direction only, and no sample, label or statistic of $\mathcal{D}_{val}$ or $\mathcal{D}_{te}$ re-enters $\mathcal{D}_{tr}$.

E. Training-Only Class-Imbalance Handling

Seven operators are compared, spanning three families. Random Over-Sampling duplicates minority records; SMOTE interpolates between minority neighbours; ADASYN concentrates synthesis where the local minority density is lowest; Borderline-SMOTE synthesises near the decision boundary; Random Under-Sampling discards majority records; and SMOTENN and SMOTETomek pair synthesis with edited-nearest-neighbour or Tomek-link cleaning. Each operator S_k produces

$$(X^{(k)}_{tr}, y^{(k)}_{tr}) = S_k(X_{tr}, y_{tr}), \quad k = 1, \dots, 7 \quad (5)$$

Fig. 4 shows the taxonomy and Table III reports the resulting distributions. The four oversamplers and the two hybrids expand the training subset to between 326,621 and 329,385 records, whereas Random Under-Sampling compresses it to 6,694 records a fiftyfold reduction that has direct consequences for training cost. ADASYN and SMOTEENN depart slightly from exact parity, at 50.0120% and 50.4115% fraud respectively, because both couple generation to local neighbourhood structure. Every operator is fitted on the training subset alone.

TABLE III. CONFIGURATION AND RESULTING TRAINING DISTRIBUTION OF THE SEVEN RESAMPLING OPERATORS

Operator	Family	Configuration	Normal after	Fraud after	Total	Fraud (%)	Objective
Random Over-Sampling	Oversampling	random_state = 42	164,653	164,653	329,306	50.0000	Balance the classes by duplicating minority records
SMOTE	Synthetic oversampling	k_neighbors = 5	164,653	164,653	329,306	50.0000	Interpolate new fraud records between neighbours
ADASYN	Adaptive synthetic	n_neighbors = 5	164,653	164,732	329,385	50.0120	Concentrate synthesis in difficult minority regions
Borderline-SMOTE	Boundary-focused	k_neighbors = 5	164,653	164,653	329,306	50.0000	Synthesise fraud records near the class boundary
Random Under-Sampling	Undersampling	random_state = 42	3,347	3,347	6,694	50.0000	Reduce the majority class by random removal
SMOTEENN	Hybrid	default SMOTE + ENN	161,968	164,653	326,621	50.4115	Synthesise, then clean by edited nearest neighbours
SMOTETomek	Hybrid	default SMOTE + Tomek	164,653	164,653	329,306	50.0000	Synthesise, then remove Tomek-link overlaps

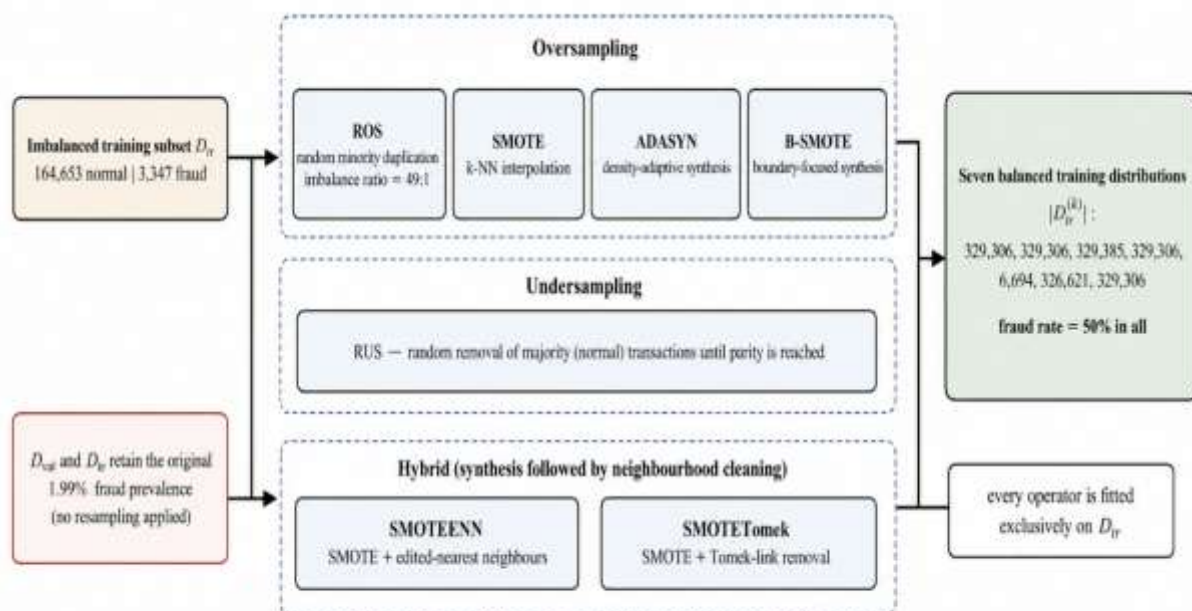


Fig. 4. Training-only sampling framework. Seven operators drawn from three families are fitted on the training subset alone; the validation and test subsets retain their original 1.99% fraud prevalence.

F. Multi-Model Ensemble Learning

Five ensemble learners are trained independently on each of the seven balanced training sets. Random Forest and Extra Trees represent bagging; XGBoost, LightGBM and CatBoost represent gradient boosting. The factorial design yields

$$|\mathcal{C}| = |\mathcal{S}| \times |\mathcal{M}| = 7 \times 5 = 35 \quad (6)$$

configurations. Model parameters are obtained by minimizing the empirical loss on the resampled training set,

$$\theta_{k,l}^* = \operatorname{argmin}_{\theta} (1/N_k) \sum_i \mathcal{L}(y_i^{(k)}, M_l(x_i^{(k)}; \theta)) \quad (7)$$

and the fitted model emits a fraud probability for every validation observation,

$$\hat{p}^{(k,l)}_{val,i} = P(y_i = 1 | x_{val,i}; C_{k,l}) \quad (8)$$

Table IV lists the hyperparameters, which are identical across all seven operators. Holding them fixed is deliberate: it removes tuning as a confounder, so any observed difference is attributable to the sampler, the learner, or their interaction. Algorithm 2 states the training loop.

Algorithm 2 Multi-sampling and multi-model training

Input: (X_{tr}, y_{tr}) , (X_{val}, y_{val}) ; sampler set $\mathcal{S}$ of size 7; classifier set $\mathcal{M}$ of size 5 with the hyperparameters of Table IV.

Output: A repository of 35 trained configurations with their training and validation probability outputs.

1. Initialise an empty repository for models, probability vectors and resampled class counts.
2. For each sampler $S_k \in \mathcal{S}$:
3. Fit S_k on the training subset alone and generate $(X^{(k)}_{tr}, y^{(k)}_{tr})$.
4. Record the majority and minority counts produced by the operator.
5. Leave $\mathcal{D}_{val}$ and $\mathcal{D}_{te}$ unchanged and unsampled.
6. For each classifier $M_l \in \mathcal{M}$:
7. Instantiate M_l with its fixed hyperparameters and fit it on $(X^{(k)}_{tr}, y^{(k)}_{tr})$ by minimizing the empirical loss of (7).
8. Emit fraud probabilities for every training and validation observation, as in (8).
9. Store the sampler name, classifier name, fitted parameters, class distribution and probability vectors.
10. Verify that 35 configurations have been produced, as in (6).
11. Return the repository and the associated probability outputs.

TABLE IV. FIXED HYPERPARAMETER CONFIGURATION OF THE FIVE ENSEMBLE CLASSIFIERS

Classifier	Ensemble family	Configured hyperparameters
Random Forest	Bagging	trees = 250; max depth = 20; class weight = balanced; n_jobs = -1; random_state = 42
Extra Trees	Bagging (extremely randomized)	trees = 250; max depth = 20; class weight = balanced; n_jobs = -1; random_state = 42
XGBoost	Gradient boosting	estimators = 350; max depth = 7; learning rate = 0.05; subsample = 0.85; colsample_bytree = 0.85; eval metric = logloss; n_jobs = -1; random_state = 42
LightGBM	Histogram gradient boosting	estimators = 350; max depth = 8; learning rate = 0.05; class weight = balanced; random_state = 42
CatBoost	Ordered gradient boosting	iterations = 350; depth = 8; learning rate = 0.05; loss = Logloss; verbose = 0; random_state = 42

Hyperparameters are held constant across all seven resampling operators so that any difference in performance is attributable to the sampler, the learner, or their interaction rather than to tuning.

G. Validation-Driven Threshold Optimization

Rather than binarizing at 0.50, each configuration receives its own threshold. For every candidate τ in $\mathcal{T} = \{0.01, 0.02, \dots, 0.99\}$ the validation probabilities are binarized,

$$\hat{y}_{val,i}(\tau) = 1 \text{ if } \hat{p}^{(k,l)}_{val,i} \geq \tau, \text{ else } 0 \quad (9)$$

and the confusion counts yield

$$P_{val}(\tau) = TP/(TP+FP), \quad R_{val}(\tau) = TP/(TP+FN) \quad (10)$$

$$F_{1,val}(\tau) = 2 P_{val}(\tau) R_{val}(\tau) / (P_{val}(\tau) + R_{val}(\tau)) \quad (11)$$

The selected threshold is

$$\tau^*_{k,l} = \operatorname{argmax}_{\tau \in \mathcal{T}} F^{(k,l)}_{1,val}(\tau) \quad (12)$$

and it is transferred unchanged to the test probabilities,

$$\hat{y}_{te,i} = 1 \text{ if } \hat{p}^{(k,l)}_{te,i} \geq \tau^*_{k,l}, \text{ else } 0 \quad (13)$$

Because the search is conducted entirely on validation data, the test labels never influence the operating point. Algorithm 3 states the procedure.

H. Evaluation Protocol and Model Selection

All metrics are computed on the untouched test subset from the confusion counts. Accuracy is $(TP+TN)/N$, and the three rate metrics are

$$P = TP/(TP+FP), \quad R = TP/(TP+FN) \\ F_1 = 2PR/(P+R) \quad (14)$$

together with the two threshold-free areas

$$ROC-AUC = \int TPR \, d(FPR) \quad (15)$$

$$PR-AUC = \int P \, dR \quad (16)$$

where the integrals run over $[0, 1]$. Under a 1.99% positive rate the accuracy of (14) is dominated by the majority class and ROC-AUC is optimistic, because the false-positive rate is normalised by a very large negative count. PR-AUC and F_1 are therefore the primary criteria. The final detector is

$$C^* = \operatorname{argmax}_{C_{k,l} \in \mathcal{C}} F^{(k,l)}_{1,te} \quad (17)$$

with PR-AUC, ROC-AUC, recall and precision used in that order to break ties. Fig. 5 summarises the complete experimental and model-selection design.

Algorithm 3 Validation-driven threshold optimization and model selection

Input: For every trained configuration $C_{k,l}$: y_{val} , $\hat{p}^{(k,l)}_{val}$, X_{te} , y_{te} ; candidate set $\mathcal{T} = \{0.01, 0.02, \dots, 0.99\}$.

Output: Optimal thresholds, test predictions, test metrics, a ranked list of the 35 configurations and the selected detector C^* .

1. Initialise an empty results table and threshold repository.
2. For each configuration $C_{k,l}$:
3. For each $\tau \in \mathcal{T}$: binarize the validation probabilities by (9), form the confusion counts, and compute $P_{val}(\tau)$, $R_{val}(\tau)$ and $F_{1,val}(\tau)$ by (10) and (11).
4. Select $\tau^*_{k,l}$ by (12) and store it.
5. Score the untouched test subset and threshold the probabilities at $\tau^*_{k,l}$ by (13).
6. Form the test confusion matrix and compute accuracy, precision, recall and F_1 by (14), ROC-AUC by (15) and PR-AUC by (16).
7. Store the threshold and all test metrics.
8. Rank the configurations by test F_1 , breaking ties by PR-AUC, then ROC-AUC, then recall, then precision.
9. Select C^* by (17) and return the ranked table, the thresholds and the selected detector.

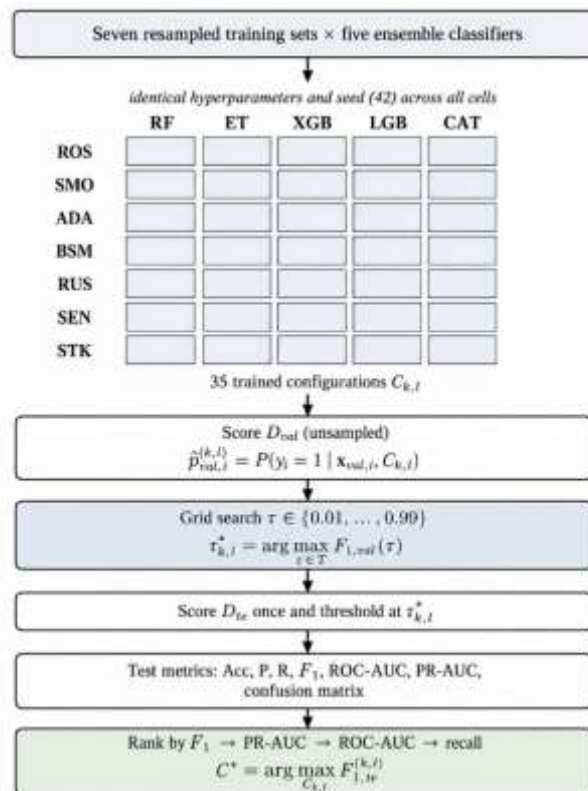


Fig. 5. Experimental and model-selection design. Every cell of the 7×5 grid is trained with identical hyperparameters and seed, scored on the validation subset to fix τ^* , and then evaluated once on the test subset.

IV. IMPLEMENTATION AND EXPERIMENTAL SETUP

The framework was implemented in Python using Scikit-learn for preprocessing and bagging ensembles, Imbalanced-learn for the resampling operators, and the XGBoost, LightGBM and CatBoost libraries for the boosting ensembles. Preprocessing, resampling, model fitting, threshold selection and final testing were executed as strictly separated stages so that the leakage barrier of Fig. 3 could be enforced and audited.

A. Experimental Environment

Experiments were executed in Google Colab on a high-RAM runtime with a v6e-1 Tensor Processing Unit accelerator. Parallel execution was enabled for the learners that support it. All 35 configurations were run under the same runtime, the same partitions, the same random seed and the same preprocessing pipeline, so that the comparison between them is controlled.

B. Training and Execution Protocol

The experiment is a controlled factorial design. Preprocessing transformations were fitted on the training subset and applied unchanged to the validation and test subsets. Each of the seven resampling operators was then applied to the processed training data, giving seven distinct training distributions. Random Forest, Extra Trees, XGBoost, LightGBM and CatBoost were fitted independently on each, producing 35 sampler-classifier combinations. The validation and test subsets retained their original class proportions throughout. For every trained configuration, class probabilities were generated on the training, validation and test subsets and stored for threshold optimization and final evaluation.

C. Reproducibility Controls

A fixed random seed of 42 was used for every stochastic operation that supports one, including the stratified split, all seven samplers and all five learners. The fitted imputation medians, categorical encoders, scaling parameters, feature ordering, sampler settings, classifier settings, validation probabilities, selected thresholds and final test metrics were all persisted, so that the transformation applied to a future transaction is identical to the one applied during training. The same partition, the same metric definitions and the same selection rule were enforced across all configurations. The test subset was excluded from parameter estimation, resampling, model fitting, hyperparameter specification and threshold optimization; it was used only after τ^* had been fixed on validation data. The implementation and experimental workflow are publicly available to support replication.

D. Computational Considerations

Alongside predictive quality, the resampled training-set size, model-fitting time, probability-generation time and inference time were recorded for each configuration. The oversamplers and hybrids expand the training subset roughly ninetyfold relative to the original minority count, from 168,000 to between 326,621 and 329,385 records, whereas Random Under-Sampling reduces it to 6,694 records. The hybrids incur additional preprocessing overhead because synthesis is followed by a neighbourhood-cleaning pass. A configuration whose predictive scores are marginally higher is not automatically the correct deployment choice if its training, memory or inference cost is disproportionate; the discussion in Section V-G therefore reads the ranking alongside these costs.

V. RESULTS AND DISCUSSION

This section reports the outcome of the 35 configurations. Because fraud constitutes 1.99% of the corpus, recall, F_1 -score and PR-AUC are prioritised over accuracy throughout. The analysis proceeds from the effect of resampling on the training distribution, through the aggregate and metric-wise comparisons, to the ROC and confusion-matrix evidence, and closes with the limitations of the study.

A. Effect of Resampling on the Training Distribution

Fig. 6 shows the class distribution produced by each operator, with the original imbalanced training subset included for reference. Random Over-Sampling, SMOTE, Borderline-SMOTE and SMOTETomek all produce exact parity at 164,653 records per class. ADASYN generates slightly more minority records than majority records, 164,732 against 164,653, because it allocates synthesis in proportion to local difficulty; the resulting fraud rate is 50.0120%. SMOTEENN removes 2,685 majority records during its edited-nearest-neighbour pass, leaving 161,968 against 164,653 and a fraud rate of 50.4115%. Random Under-Sampling reaches parity by discarding majority records, and it does so at a cost: the training subset falls from 168,000 records to 6,694, a reduction of 96%. The four oversamplers and two hybrids therefore trade computation for retained information, while the undersampler makes the opposite trade.

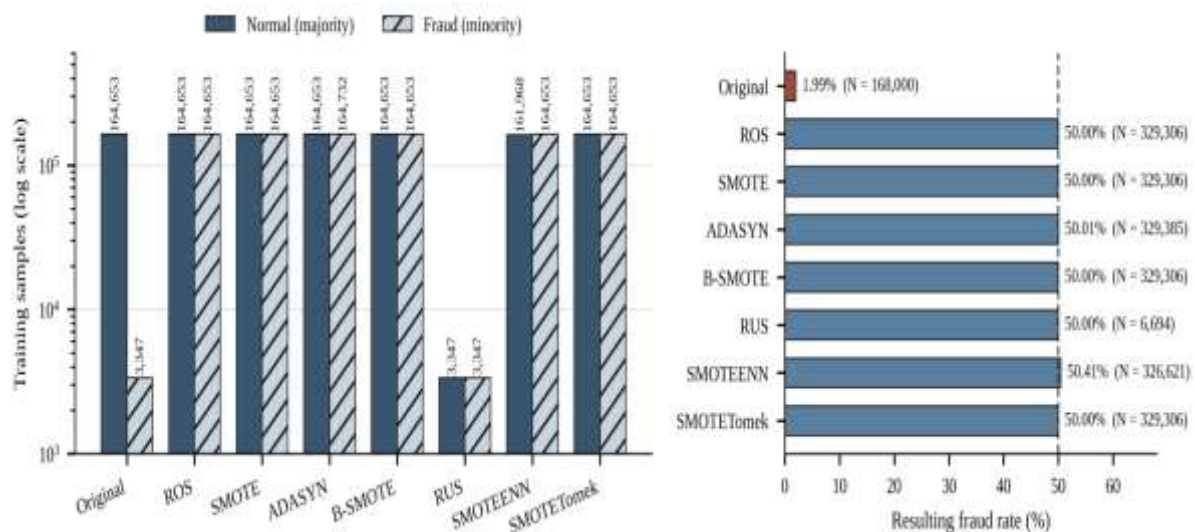


Fig. 6. Effect of the seven resampling operators on the training class distribution. Left: majority and minority counts on a logarithmic scale, with the original imbalanced subset shown for reference. Right: the resulting fraud rate and total training size for each operator.

B. Overall Comparison of the 35 Configurations

Fig. 7 presents the test-set results as a heatmap, with shading normalised within each metric column so that differences in the fourth and fifth decimal place remain visible. Random Forest attains a test F_1 -score of exactly 1.0000 under four of the seven operators ROS, ADASYN, Borderline-SMOTE and SMOTETomek and remains above 0.99978 under the other three. Extra Trees is the most consistent learner overall, returning 0.99993 under six of the seven operators and 0.99969 under SMOTEENN, and achieving perfect recall in every case. The three boosting learners are close behind but systematically lower: XGBoost ranges from 0.99936 to 0.99645, LightGBM from 0.99851 to 0.99845, and CatBoost from 0.99885 to 0.99874. The ordering is stable across operators, which indicates that the learner, rather than the balancing strategy, drives the residual differences at this level of performance.

C. Metric-Wise Analysis

Figs. 8–13 disaggregate the comparison by metric. In each figure the vertical axis is truncated to the observed range, because all 35 configurations lie above 0.9969 and an axis anchored at zero would render the bars indistinguishable.

Test accuracy (Fig. 8) spans only 0.998444 to 1.000000. This narrow band is itself the argument against accuracy as a selection criterion: a trivial majority-class predictor would already score 0.980083 on this partition, so the entire useful range of the metric is compressed into the final two percentage points.

The F_1 -score (Fig. 9) spans 0.998447 to 1.000000 and separates the learners more usefully, because it responds to false positives and false negatives simultaneously. Random Forest leads under most operators, Extra Trees is marginally behind but far more uniform, and LightGBM is consistently lowest.

PR-AUC (Fig. 10) ranges from 0.999908 to 1.000000. Because it summarises precision across the full range of achievable recall without reference to the large negative count, it is the most informative threshold-free measure under this prevalence. Random Forest and Extra Trees reach 1.000000 under several operators; the boosting learners settle between 0.99991 and 0.99997.

Precision (Fig. 11) shows the widest spread, from 0.996987 to 1.000000, and is the metric on which the learners genuinely differ. Random Forest attains perfect precision under every operator, meaning that no legitimate transaction was flagged. LightGBM is lowest at 0.996987 under four operators, which at this test size corresponds to on the order of two additional false alerts per 36,000 transactions small in absolute terms but a direct multiplier on manual review workload at production volume.

Recall (Fig. 12) ranges from 0.999556 to 1.000000. Extra Trees and XGBoost achieve perfect recall under every operator; Random Forest drops to 0.999556 under SMOTE and to 0.999778 under RUS, corresponding to one and two missed fraudulent transactions respectively. In a financial setting a false negative is a realised loss, so this asymmetry matters more than the precision differences noted above.

ROC-AUC (Fig. 13) is compressed into 0.999912 to 1.000000 and is nearly uninformative here. Because the false-positive rate is normalised by 35,283 negative instances, even a substantial number of false alerts moves the curve very little. This is the concrete form of the optimism that motivated the emphasis on PR-AUC in Section III-H.

D. ROC-Based Analysis

Fig. 14 overlays the ROC curves of all 35 configurations. Almost every curve is pinned to the upper-left corner, with true-positive rates approaching 1 at false-positive rates approaching 0, and the reported areas cluster between 0.9999 and 1.0000. The visual overlap is so complete that the figure cannot discriminate between configurations; it establishes only that every configuration separates the classes well. Fig. 15 restricts the plot to the ten highest-ranked configurations and confirms the same picture at finer resolution: the curves are effectively coincident, and all ten lie far above the random-classifier diagonal. Meaningful discrimination between these ten therefore has to come from the precision–recall evidence of Fig. 10 and the error counts of Section V-E rather than from ROC geometry.

E. Confusion-Matrix and Error Analysis

Figs. 16–18 report the confusion matrices of the ADASYN–Extra Trees configuration, whose validation-selected threshold is 0.6983. On the test subset (Fig. 16) the model recovers all 717 fraudulent transactions and classifies all 35,283 legitimate transactions correctly: zero false negatives and zero false positives. On the resampled training subset (Fig. 17) it separates all 164,653 majority and 164,732 minority records without error, and on the validation subset (Fig. 18) it recovers all 718 fraudulent transactions with no false positives among the 35,282 legitimate records.

The absence of false negatives is the operationally important result, because an undetected fraudulent transaction is a realised monetary loss whereas a false alert costs only investigator time. The absence of false positives at the selected threshold indicates that the validation-driven operating point of 0.6983 substantially above the conventional 0.50 suppresses spurious alerts without sacrificing sensitivity, which is exactly the behaviour the threshold search of Section III-G was designed to obtain. The perfect training-set separation should nonetheless be read with caution: it indicates that the adaptively balanced training distribution is highly separable for a randomized tree ensemble, not that the decision function will remain error-free on data drawn from a later period or a different institution.

F. Complete Results Across All Configurations

Table V reports the full profile of all 35 configurations: the validation-selected threshold and accuracy, precision, recall, F_1 -score, ROC-AUC and PR-AUC on the training, validation and test subsets. Two patterns are worth noting. First, the selected threshold varies systematically with the learner rather than with the sampler: Random Forest selects thresholds near 0.98, Extra Trees near 0.70, XGBoost near 0.89, and LightGBM and CatBoost between 0.75 and 0.80. A single fixed threshold applied across the grid would therefore have been badly mismatched for at least three of the five learners, which is direct evidence for the configuration-specific approach of (12). Second, the sampler has almost no influence within a learner: the ROS, SMOTE, ADASYN, Borderline-SMOTE and SMOTETomek rows are numerically identical for

XGBoost, LightGBM and CatBoost. This indicates that once the training distribution has been balanced, the particular mechanism used to balance it is largely immaterial for these boosting learners on this corpus.

G. Discussion, Limitations and Threats to Validity

Three findings emerge. Bagging ensembles outperform boosting ensembles consistently on this corpus, and the margin, although small, is stable across all seven operators. The choice of resampling operator matters far less than the literature reviewed in Section II-C would suggest, at least once leakage has been eliminated and the threshold has been tuned; the practical differences between ROS, SMOTE, ADASYN, Borderline-SMOTE and SMOTETomek are negligible, while SMOTEENN is marginally worse and RUS achieves comparable quality at a fraction of the training cost. Threshold selection, by contrast, matters a great deal, and the dispersion of τ^* across learners in Table V quantifies that.

Several limitations qualify these conclusions. The performance levels reported here are unusually high for fraud detection, and the most plausible explanation is that this corpus contains fraud signatures that are more cleanly separable than those encountered in production streams; the results should not be read as an estimate of field performance. The evaluation uses a single stratified split rather than repeated splits or cross-validation, so no variance estimate accompanies the point estimates, and differences in the fourth decimal place should not be over-interpreted. The partition is random rather than temporal, so the protocol does not measure resistance to concept drift, which the drift-oriented work reviewed in Section II-B identifies as a first-order concern. No adversarial evaluation was performed, so the robustness question raised in [9] remains open. Finally, RUS discards 96% of the training records yet performs comparably, which is itself evidence that the discriminative signal in this corpus is concentrated and redundant — a property that may not transfer to other institutions.

		Precision	Recall	F_1	ROC-AUC	PR-AUC
ROS	RF	1.00000	1.00000	1.00000	1.00000	1.00000
	ET	0.99987	1.00000	0.99993	1.00000	1.00000
	XGB	0.99911	1.00000	0.99956	0.99997	0.99997
	LGBM	0.99699	0.99991	0.99845	0.99994	0.99994
	CatB	0.99783	0.99969	0.99876	0.99995	0.99994
SMOTE	RF	1.00000	0.99956	0.99978	1.00000	1.00000
	ET	0.99987	1.00000	0.99993	1.00000	1.00000
	XGB	0.99911	1.00000	0.99956	0.99997	0.99997
	LGBM	0.99699	0.99991	0.99845	0.99994	0.99994
	CatB	0.99783	0.99969	0.99876	0.99995	0.99994
ADASYN	RF	1.00000	1.00000	1.00000	1.00000	1.00000
	ET	0.99987	1.00000	0.99993	1.00000	1.00000
	XGB	0.99911	1.00000	0.99956	0.99997	0.99997
	LGBM	0.99699	0.99991	0.99845	0.99994	0.99994
	CatB	0.99783	0.99969	0.99876	0.99995	0.99994
B-SMOTE	RF	1.00000	1.00000	1.00000	1.00000	1.00000
	ET	0.99987	1.00000	0.99993	1.00000	1.00000
	XGB	0.99911	1.00000	0.99956	0.99997	0.99997
	LGBM	0.99699	0.99991	0.99845	0.99994	0.99994
	CatB	0.99783	0.99969	0.99876	0.99995	0.99994
RUS	RF	1.00000	0.99978	0.99989	1.00000	1.00000
	ET	0.99987	1.00000	0.99993	1.00000	1.00000
	XGB	0.99929	1.00000	0.99965	0.99997	0.99997
	LGBM	0.99747	0.99991	0.99869	0.99994	0.99993
	CatB	0.99800	0.99969	0.99884	0.99995	0.99995
SMOTEENN	RF	1.00000	0.99987	0.99993	1.00000	1.00000
	ET	0.99947	0.99991	0.99969	1.00000	1.00000
	XGB	0.99871	1.00000	0.99936	0.99994	0.99994
	LGBM	0.99703	1.00000	0.99851	0.99991	0.99991
	CatB	0.99778	0.99969	0.99874	0.99992	0.99992
SMOTETomek	RF	1.00000	1.00000	1.00000	1.00000	1.00000
	ET	0.99987	1.00000	0.99993	1.00000	1.00000
	XGB	0.99911	1.00000	0.99956	0.99997	0.99997
	LGBM	0.99699	0.99991	0.99845	0.99994	0.99994
	CatB	0.99783	0.99969	0.99876	0.99995	0.99994

Fig. 7. Test-set performance of all 35 sampling-classifier configurations. Shading is normalised within each metric column so that differences in the fourth and fifth decimal place remain visible; the maximum value in each column is shown in bold.

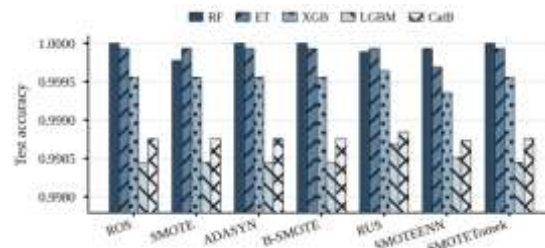


Fig. 8. Test accuracy by sampling operator and classifier. The vertical axis is truncated to the observed range; a majority-class predictor would already score 0.9801.

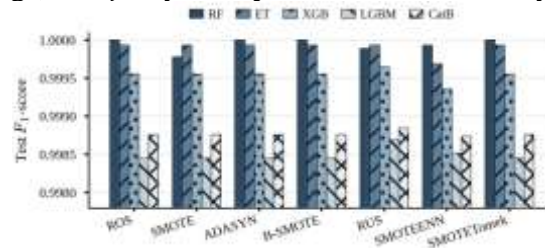


Fig. 9. Test F_1 -score by sampling operator and classifier.

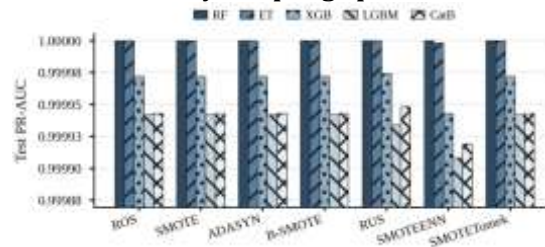


Fig. 10. Test PR-AUC by sampling operator and classifier. PR-AUC is the most informative threshold-free measure at a 1.99% positive rate.

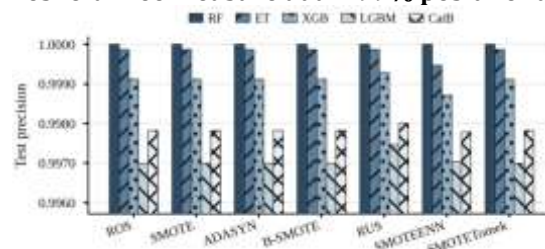


Fig. 11. Test precision by sampling operator and classifier. This is the metric on which the learners differ most.

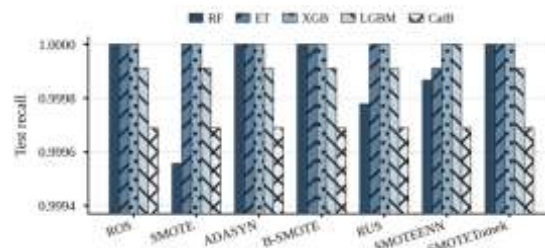


Fig. 12. Test recall by sampling operator and classifier. Extra Trees and XGBoost attain perfect recall under every operator.

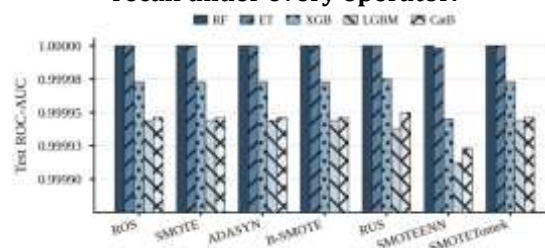


Fig. 13. Test ROC-AUC by sampling operator and classifier. The range spans less than 10^{-4} , illustrating the optimism of ROC-AUC under extreme skew.

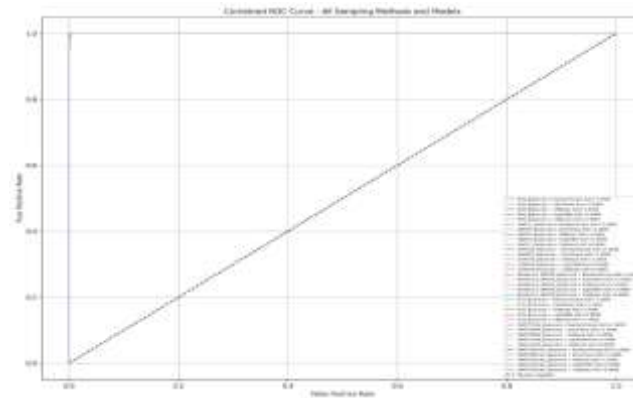


Fig. 14. Combined ROC curves of all 35 sampling-classifier configurations. The curves are almost coincident at the upper-left corner, so the figure establishes uniformly strong separation but cannot discriminate between configurations.

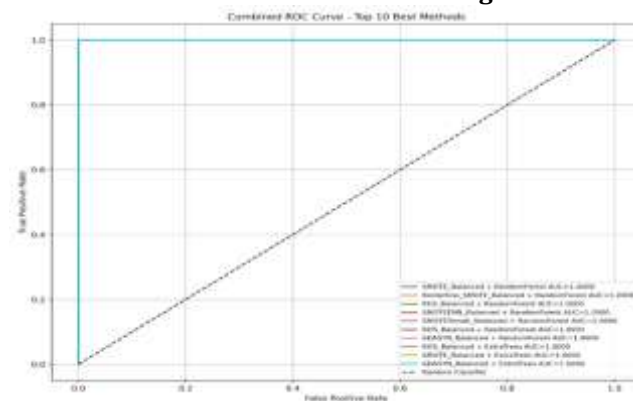


Fig. 15. Combined ROC curves of the ten highest-ranked configurations. The curves remain effectively coincident, confirming that discrimination must come from precision-recall and confusion-matrix evidence.

ADASYN – Extra Trees, test set ($\tau^* = 0.6983$)

	Predicted normal	Predicted fraud
Actual normal	35,283	0
Actual fraud	0	717

Fig. 16. Test confusion matrix of the ADASYN-Extra Trees configuration at $\tau^* = 0.6983$: all 717 fraudulent transactions recovered with no false positives among 35,283 legitimate transactions.

ADASYN – Extra Trees, resampled training set

	Predicted normal	Predicted fraud
Actual normal	164,653	0
Actual fraud	0	164,732

Fig. 17. Confusion matrix of the ADASYN-Extra Trees configuration on the resampled training subset.

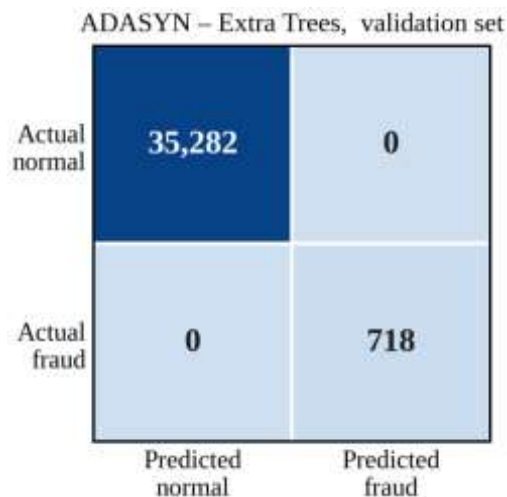


Fig. 18. Validation confusion matrix of the ADASYN-Extra Trees configuration: all 718 fraudulent transactions recovered with no false positives.

TABLE V. TRAINING, VALIDATION AND TEST PERFORMANCE OF ALL 35 SAMPLING-CLASSIFIER CONFIGURATIONS

Configuration		τ^*	Training subset (resampled)							Validation subset							Test subset					
Sampler	Model		Acc	Pre _c	Rec	F1	RO _C	PR	Acc	Pre _c	Rec	F1	RO _C	PR	Acc	Pre _c	Rec	F1	RO _C	PR		
ROS	RF	0.979 2	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000		
	ET	0.698 3	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0	0.9 998	0.9 995	1.0 000	0.9 998	1.0 000	1.0	0.9 999	0.9 999	1.0 000	0.9 999	1.0 000	1.0		
	XGB	0.888 9	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0	0.9 993	0.9 986	1.0 000	0.9 993	0.9 999	0.9 999	0.9 996	0.9 991	1.0 000	0.9 996	1.0 000	1.0		
	LGBM	0.749 3	0.9 997	0.9 995	0.9 998	0.9 997	1.0 000	1.0	0.9 982	0.9 965	0.9 999	0.9 982	0.9 998	0.9 998	0.9 984	0.9 970	0.9 999	0.9 984	0.9 999	0.9		
	CatB	0.746 7	0.9 994	0.9 990	0.9 999	0.9 994	1.0 000	1.0	0.9 984	0.9 972	0.9 995	0.9 984	0.9 998	0.9 998	0.9 988	0.9 978	0.9 997	0.9 988	0.9 999	0.9		
SMOTE	RF	0.979 2	0.9 999	1.0 000	0.9 998	0.9 999	1.0 000	1.0	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0	0.9 998	1.0 000	0.9 996	0.9 998	1.0 000	1.0		
	ET	0.698 3	0.9 999	1.0 000	0.9 998	0.9 999	1.0 000	1.0	0.9 998	0.9 995	1.0 000	0.9 998	1.0 000	1.0	0.9 999	0.9 999	1.0 000	0.9 999	1.0 000	1.0		
	XGB	0.888 9	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0	0.9 993	0.9 986	1.0 000	0.9 993	0.9 999	0.9 999	0.9 996	0.9 991	1.0 000	0.9 996	1.0 000	1.0		
	LGBM	0.749 3	0.9 997	0.9 995	0.9 998	0.9 997	1.0 000	1.0	0.9 982	0.9 965	0.9 999	0.9 982	0.9 998	0.9 998	0.9 984	0.9 970	0.9 999	0.9 984	0.9 999	0.9		
	CatB	0.746 7	0.9 994	0.9 990	0.9 999	0.9 994	1.0 000	1.0	0.9 984	0.9 972	0.9 995	0.9 984	0.9 998	0.9 998	0.9 988	0.9 978	0.9 997	0.9 988	0.9 999	0.9		
ADASYN	RF	0.979 2	0.9 999	1.0 000	0.9 998	0.9 999	1.0 000	1.0	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0		
	ET	0.698 3	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0	0.9 998	0.9 995	1.0 000	0.9 998	1.0 000	1.0	0.9 999	0.9 999	1.0 000	0.9 999	1.0 000	1.0		
	XGB	0.888 9	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0	0.9 993	0.9 986	1.0 000	0.9 993	0.9 999	0.9 999	0.9 996	0.9 991	1.0 000	0.9 996	1.0 000	1.0		
	LGBM	0.749 3	0.9 997	0.9 995	0.9 998	0.9 997	1.0 000	1.0	0.9 982	0.9 965	0.9 999	0.9 982	0.9 998	0.9 998	0.9 984	0.9 970	0.9 999	0.9 984	0.9 999	0.9		
	CatB	0.746 7	0.9 994	0.9 990	0.9 999	0.9 994	1.0 000	1.0	0.9 984	0.9 972	0.9 995	0.9 984	0.9 998	0.9 998	0.9 988	0.9 978	0.9 997	0.9 988	0.9 999	0.9		
B-SMOTE	RF	0.979 2	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0		
	ET	0.698 3	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0	0.9 998	0.9 995	1.0 000	0.9 998	1.0 000	1.0	0.9 999	0.9 999	1.0 000	0.9 999	1.0 000	1.0		

	XGB	0.888 9	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	0.9 993	0.9 986	1.0 000	0.9 993	0.9 999	0.9 999	0.9 996	0.9 991	1.0 000	0.9 996	1.0 000	1.0 000
	LGB M	0.749 3	0.9 997	0.9 995	0.9 998	0.9 997	1.0 000	1.0 000	0.9 982	0.9 965	0.9 999	0.9 982	0.9 998	0.9 998	0.9 984	0.9 970	0.9 999	0.9 984	0.9 999	0.9 999
	CatB	0.746 7	0.9 994	0.9 990	0.9 999	0.9 994	1.0 000	1.0 000	0.9 984	0.9 972	0.9 995	0.9 984	0.9 998	0.9 998	0.9 988	0.9 978	0.9 997	0.9 988	0.9 999	0.9 999
RUS	RF	0.977 9	0.9 999	1.0 000	0.9 998	0.9 999	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	0.9 999	1.0 000	0.9 998	0.9 999	1.0 000	1.0 000
	ET	0.698 3	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	0.9 998	0.9 995	1.0 000	0.9 998	1.0 000	1.0 000	0.9 999	0.9 999	1.0 000	0.9 999	1.0 000	1.0 000
	XGB	0.889 8	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	0.9 992	0.9 984	1.0 000	0.9 992	0.9 999	0.9 999	0.9 996	0.9 993	1.0 000	0.9 996	1.0 000	1.0 000
	LGB M	0.769 0	0.9 997	0.9 995	0.9 998	0.9 997	1.0 000	1.0 000	0.9 984	0.9 969	0.9 999	0.9 984	0.9 998	0.9 998	0.9 987	0.9 975	0.9 999	0.9 987	0.9 999	0.9 999
	CatB	0.764 4	0.9 995	0.9 991	0.9 999	0.9 995	1.0 000	1.0 000	0.9 984	0.9 974	0.9 995	0.9 984	0.9 998	0.9 998	0.9 988	0.9 980	0.9 997	0.9 988	1.0 000	0.9 999
SMOTEEN N	RF	0.980 7	0.9 999	1.0 000	0.9 997	0.9 999	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	0.9 999	1.0 000	0.9 999	0.9 999	1.0 000	1.0 000
	ET	0.710 5	0.9 996	0.9 995	0.9 996	0.9 996	1.0 000	1.0 000	0.9 995	0.9 992	0.9 998	0.9 995	1.0 000	1.0 000	0.9 997	0.9 995	0.9 999	0.9 997	1.0 000	1.0 000
	XGB	0.899 1	0.9 995	0.9 991	1.0 000	0.9 995	0.9 999	0.9 999	0.9 990	0.9 980	1.0 000	0.9 990	0.9 998	0.9 998	0.9 994	0.9 987	1.0 000	0.9 994	0.9 999	0.9 999
	LGB M	0.798 4	0.9 993	0.9 986	1.0 000	0.9 993	0.9 999	0.9 999	0.9 982	0.9 964	1.0 000	0.9 982	0.9 998	0.9 997	0.9 985	0.9 970	1.0 000	0.9 985	0.9 999	0.9 999
	CatB	0.786 1	0.9 991	0.9 983	0.9 999	0.9 991	0.9 999	0.9 999	0.9 982	0.9 969	0.9 995	0.9 982	0.9 998	0.9 998	0.9 987	0.9 978	0.9 997	0.9 987	0.9 999	0.9 999
SMOTETo mek	RF	0.979 2	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000
	ET	0.698 3	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	0.9 998	0.9 995	1.0 000	0.9 998	1.0 000	1.0 000	0.9 999	0.9 999	1.0 000	0.9 999	1.0 000	1.0 000
	XGB	0.888 9	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	1.0 000	0.9 993	0.9 986	1.0 000	0.9 993	0.9 999	0.9 999	0.9 996	0.9 991	1.0 000	0.9 996	1.0 000	1.0 000
	LGB M	0.749 3	0.9 997	0.9 995	0.9 998	0.9 997	1.0 000	1.0 000	0.9 982	0.9 965	0.9 999	0.9 982	0.9 998	0.9 998	0.9 984	0.9 970	0.9 999	0.9 984	0.9 999	0.9 999
	CatB	0.746 7	0.9 994	0.9 990	0.9 999	0.9 994	1.0 000	1.0 000	0.9 984	0.9 972	0.9 995	0.9 984	0.9 998	0.9 998	0.9 988	0.9 978	0.9 997	0.9 988	0.9 999	0.9 999

Each threshold τ^* is the value that maximises the validation F_1 -score for that configuration and is applied unchanged to the test subset. Values are rounded to four decimal places; the configurations attaining the maximum test F_1 are shown in bold. Horizontal rules separate the seven resampling operators.

VI. CONCLUSION AND FUTURE WORK

This paper presented a leakage-aware, imbalance-sensitive framework for banking transaction fraud detection that crosses seven resampling operators with five ensemble classifiers and selects a configuration-specific decision threshold on validation data. All preprocessing statistics were estimated on the training subset alone, resampling was confined to the training subset, and the test subset was scored once, after every configuration choice had been fixed. Across the 35 configurations, Random Forest and Extra Trees were consistently the strongest and most stable learners. The ADASYN-Extra Trees configuration, operating at a validation-selected threshold of 0.6983, recovered all 717 fraudulent transactions in the test subset with no false positives among 35,283 legitimate transactions. The dispersion of selected thresholds across learners, from 0.6983 to 0.9807, shows that a fixed cut-off of 0.50 would have been inappropriate for most of the grid and supports the central methodological claim of the paper.

Future work will extend the protocol in four directions: temporal rather than random partitioning, so that resistance to concept drift can be measured directly; repeated splits or nested cross-validation, so that the point estimates acquire confidence intervals; explainability analysis, so that individual alerts can be justified to investigators and regulators; and evaluation on disjoint real-world banking corpora, so that the transferability of the observed ranking can be tested.

REFERENCES

- [1] M. Adil, Z. Yinjun, M. M. Jamjoom, and Z. Ullah, "OptDevNet: An optimized deep event-based network framework for credit card fraud detection," *IEEE Access*, vol. 12, pp. 132421–132433, 2024.
- [2] L. Ni, J. Li, H. Xu, X. Wang, and J. Zhang, "Fraud feature boosting mechanism and spiral oversampling balancing technique for credit card fraud detection," *IEEE Trans. Comput. Social Syst.*, vol. 11, no. 2, pp. 1615–1630, Apr. 2024.
- [3] F. K. Alarfaj and S. Shahzadi, "Enhancing fraud detection in banking with deep learning: Graph neural networks and autoencoders for real-time credit card fraud prevention," *IEEE Access*, vol. 13, pp. 20633–20646, 2025.
- [4] K. Zhu, N. Zhang, W. Ding, and C. Jiang, "An adaptive heterogeneous credit card fraud detection model based on deep reinforcement training subset selection," *IEEE Trans. Artif. Intell.*, vol. 5, no. 8, pp. 4026–4041, Aug. 2024.
- [5] T.-T.-H. Le, Y. Hwang, H. Kang, and H. Kim, "Robust credit card fraud detection based on efficient Kolmogorov–Arnold network models," *IEEE Access*, vol. 12, pp. 157006–157020, 2024.
- [6] Y. Xie, M. Zhou, G. Liu, L. Wei, H. Zhu, and P. De Meo, "A transactional-behavior-based hierarchical gated network for credit card fraud detection," *IEEE/CAA J. Autom. Sinica*, vol. 12, no. 7, pp. 1489–1503, Jul. 2025.
- [7] H. M. R. A. Lawati et al., "An integrated preprocessing and drift detection approach with adaptive windowing for fraud detection in payment systems," *IEEE Access*, vol. 13, pp. 92036–92056, 2025.
- [8] Z. Xie and X. Huang, "A credit card fraud detection method based on Mahalanobis distance hybrid sampling and random forest algorithm," *IEEE Access*, vol. 12, pp. 162788–162798, 2024.
- [9] D. Lunghi, Y. Molinghen, A. Simitsis, T. Lenaerts, and G. Bontempi, "Fraud-RLA: A reinforcement learning adversarial attack against credit card fraud detection," *IEEE Trans. Dependable Secure Comput.*, vol. 23, no. 3, pp. 7468–7481, May–Jun. 2026.
- [10] E. Ileberi and Y. Sun, "Advancing model performance with ADASYN and recurrent feature elimination and cross-validation in machine learning-assisted credit card fraud detection: A comparative analysis," *IEEE Access*, vol. 12, pp. 133315–133327, 2024.
- [11] I. Akour, N. Mohamed, and S. Salloum, "Hybrid CNN-LSTM with attention mechanism for robust credit card fraud detection," *IEEE Access*, vol. 13, pp. 114056–114068, 2025.
- [12] I. Almubark, "Advanced credit card fraud detection: An ensemble learning using random under sampling and two-stage thresholding," *IEEE Access*, vol. 12, pp. 192079–192089, 2024.
- [13] Y. Xie et al., "A spatial-temporal gated network for credit card fraud detection by learning transactional representations," *IEEE Trans. Autom. Sci. Eng.*, vol. 21, no. 4, pp. 6978–6991, Oct. 2024.
- [14] Y. Xie, G. Liu, C. Yan, C. Jiang, M. Zhou, and M. Li, "Learning transactional behavioral representations for credit card fraud detection," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 4, pp. 5735–5748, Apr. 2024.
- [15] N. F. Aurna, M. D. Hossain, L. Khan, Y. Taenaka, and Y. Kadobayashi, "FedFusion: Adaptive model fusion for addressing feature discrepancies in federated credit card fraud detection," *IEEE Access*, vol. 12, pp. 136962–136978, 2024.
- [16] J. Wu, K. Zhu, N. Zhang, R. Cui, Z. Liu, and C. Jiang, "HFDFM: A heterogeneous credit card fraud detection model based on federated learning with membership privacy," *IEEE Trans. Comput. Social Syst.*, early access, 2026.
- [17] E. Ileberi and Y. Sun, "Performance analysis of cost-sensitive oversampling mechanisms for credit card fraud detection," *IEEE Access*, vol. 13, pp. 202655–202664, 2025.
- [18] I. D. Mienye and N. Jere, "Deep learning for credit card fraud detection: A review of algorithms, challenges, and solutions," *IEEE Access*, vol. 12, pp. 96893–96910, 2024.
- [19] S. N. Kalid, K.-C. Khor, K.-H. Ng, and G.-K. Tong, "Detecting frauds and payment defaults on credit card data inherited with imbalanced class distribution and overlapping class problems: A systematic review," *IEEE Access*, vol. 12, pp. 23636–23652, 2024.
- [20] H. H. Sulimani, "AdaBoost-LSTM: A hybrid ensemble framework for sequential credit card fraud detection," *IEEE Access*, vol. 13, pp. 212504–212516, 2025.
- [21] H. Zhu, M. Zhou, G. Liu, Y. Xie, S. Liu, and C. Guo, "NUS: Noisy-sample-removed undersampling scheme for imbalanced classification and application to credit card fraud detection," *IEEE Trans. Comput. Social Syst.*, vol. 11, no. 2, pp. 1793–1804, Apr. 2024.
- [22] K. G. Dastidar, O. Caelen, and M. Granitzer, "Machine learning methods for credit card fraud detection: A survey," *IEEE Access*, vol. 12, pp. 158939–158965, 2024.
- [23] X. Zhao, Y. Liu, and Q. Zhao, "Improved LightGBM for extremely imbalanced data and application to credit card fraud detection," *IEEE Access*, vol. 12, pp. 159316–159335, 2024.

- [24] J. G. Almaraz-Rivera, J. A. Cantoral-Ceballos, J. F. Botero, F. J. Muñoz, and B. D. Martinez, "Hyphatia: A card-not-present fraud detection system based on self-supervised tabular learning," *IEEE Open J. Comput. Soc.*, vol. 6, pp. 812–821, 2025.
- [25] C. S. Nama and K. S. Banu, "Credit card fraud detection using deep learning techniques and handling unbalanced class distributions with AGSS," *IEEE Access*, vol. 14, pp. 1847–1864, 2026.
- [26] J. Arora and S. Bhardwaj, "Fraud detection in credit card transactions: Techniques and emerging challenges," *Int. J. Syst. Assur. Eng. Manag.*, 2026.
- [27] N. Gamal, E. M. G. Younis, and W. M. Makram, "Enhancing credit card fraud detection with a hybrid approach using machine and deep learning," *Sci. Rep.*, vol. 16, art. no. 10944, 2026.
- [28] C. Chekira, R. Menaoui, M. Ibtissam et al., "Evaluating ensemble combination rules for credit card fraud detection: An empirical study," *SN Comput. Sci.*, vol. 7, art. no. 517, 2026.
- [29] M. A. Ghalwash, S. M. Abdelrazek, N. H. Eladawi et al., "Enhancing credit card fraud detection using DBSCAN-augmented disjunctive voting ensemble," *Sci. Rep.*, vol. 15, art. no. 39754, 2025.
- [30] H. Abbasimehr and H. Fanai, "Credit card fraud detection using a Bayesian-optimized deep supervised autoencoder with blending ensemble learning," *Iran J. Comput. Sci.*, vol. 9, art. no. 10, 2026.
- [31] M. M. Abualhaj, R. Kumar, S. Hota et al., "Credit card fraud detection using multi-objective Harris hawks optimization and K-means SMOTE," *J. Data Inf. Manag.*, 2026.
- [32] D. Singh, S. Singh, and N. Kumar, "Benchmarking credit card fraud detection models: A comprehensive evaluation across diverse datasets," *Comput. Econ.*, 2025.
- [33] L. Ni, X. Li, Y. Zhou et al., "HMOA-GNN: Adaptive adversarial GraphSAGE with hierarchical hybrid sampling and metric-optimized graph construction for credit card fraud detection," *Sci. Rep.*, vol. 15, art. no. 43005, 2025.
- [34] M. A. Alrasheedi, "Enhancing fraud detection in credit card transactions: A comparative study of machine learning models," *Comput. Econ.*, vol. 68, pp. 779–805, 2026.
- [35] B. K. Padhi, B. Naik, R. K. Sahu et al., "A nature-inspired osprey optimization-based feature selection framework with stacking ensemble model for imbalance-aware credit card fraud detection," *Sci. Rep.*, vol. 16, art. no. 21283, 2026.
- [36] S. S. Awad, A. A. Hamza, M. A. Sobh et al., "Applying explainable artificial intelligence to interpret supervised ensemble learning models for robust credit card fraud detection," *Sci. Rep.*, vol. 16, art. no. 15220, 2026.
- [37] T. Manonmani, N. Umakanth, S. N. Kumar et al., "SecureFed+: A federated learning framework with adaptive gradient clipping and encrypted aggregation for credit card fraud detection," *Int. J. Mach. Learn. Cybern.*, vol. 17, art. no. 253, 2026.
- [38] M. Zavvar, M. Jafari, N. M. Pour et al., "A hybrid deep learning framework using synthetic oversampling, autoencoder, convolutional neural networks, and an attention mechanism for credit card fraud detection," *J. Big Data*, vol. 13, art. no. 21, 2026.
- [39] L. T. Rugeles Diaz, M. Á. Echarte Fernández, J. Jorge-Vázquez et al., "Data analytics to prevent retail credit card fraud: Empirical evidence from Latin America," *Financ. Innov.*, vol. 11, art. no. 137, 2025.
- [40] M. El Alami, N. Innan, M. Shafique et al., "Comparative performance analysis of quantum machine learning architectures for credit card fraud detection," *Appl. Intell.*, vol. 56, art. no. 83, 2026.
- [41] E. Z. Çavdar and A. Bozanta, "AI-based credit card fraud detection: A machine learning approach with model explainability on real-world data," *Knowl. Inf. Syst.*, vol. 68, art. no. 161, 2026.
- [42] S. I. Goudar, K. K. Girish, and B. Bhowmik, "Adaptive bagging strategies to address concept drift and retain knowledge in fraud detection systems," *Digit. Finance*, vol. 8, art. no. 13, 2026.