ERP-RiskBench: Leakage-Safe Ensemble Learning for Financial Risk

1st Sanjay Mishra
IEEE Senior Member
Raleigh, NC, USA
sanmish4@icloud.com

***Abstract*—Financial risk detection in Enterprise Resource Planning (ERP) systems is an important but underexplored application of machine learning. Published studies in this area tend to suffer from vague dataset descriptions, leakage-prone pipelines, and evaluation practices that inflate reported performance. This paper presents a rebuilt experimental framework for ERP financial risk detection using ensemble machine learning. The risk definition is hybrid, covering both procurement compliance anomalies and transactional fraud. A composite benchmark called ERP-RiskBench is assembled from public procurement event logs, labeled fraud data, and a new synthetic ERP dataset with rule-injected risk typologies and conditional tabular GAN augmentation. Nested cross-validation with time-aware and group-aware splitting enforces leakage prevention throughout the pipeline. The primary model is a stacking ensemble of gradient boosting methods, tested alongside linear baselines, deep tabular architectures, and an interpretable glassbox alternative. Performance is measured through Matthews Correlation Coefficient, area under the precision-recall curve, and cost-sensitive decision analysis using calibrated probabilities. Across multiple dataset configurations and a structured ablation study, the stacking ensemble achieves the best detection results. Leakage-safe protocols reduce previously inflated accuracy estimates by a notable margin. SHAP-based explanations and feature stability analysis show that procurement control features, especially three-way matching discrepancies, rank as the most informative predictors. The resulting framework provides a reproducible, operationally grounded blueprint for machine learning deployment in ERP audit and governance settings.**

***Index Terms*—Enterprise Resource Planning, Financial Risk Detection, Ensemble Learning, Stacking, Gradient Boosting, Nested Cross-Validation, Imbalanced Classification, Cost-Sensitive Learning, Procurement Fraud, Explainable AI**

I. Introduction

Enterprise Resource Planning systems form the operational backbone of modern organizations. They integrate financial, procurement, and supply chain processes into a single system of record [1]. When these systems fail or are deliberately manipulated, consequences propagate across business functions quickly. Occupational fraud alone costs organizations roughly five percent of annual revenue [2]. That scale has driven growing interest in applying machine learning to risk detection within ERP transaction data.

Despite this interest, published studies on ERP financial risk detection frequently show methodological problems. Dataset descriptions tend to be vague. Preprocessing steps are applied before data splitting. Model specifications are inconsistent. Accuracy is used as the primary metric even under heavy class imbalance. These weaknesses are not unique to the ERP domain. They reflect broader concerns about reproducibility and experimental rigor in applied machine learning [3].

The research presented here addresses these gaps head on. A complete experimental framework is developed for ERP financial risk detection, designed to be reproducible, leakage-safe, and operationally meaningful. The contribution is not a single novel algorithm. It is the disciplined integration of established techniques into a coherent and auditable pipeline.

The framework is anchored in four research questions that structure the experimental design.

RQ1 (Predictive Performance). Which model families achieve the strongest risk detection on ERP-like data under severe class imbalance? Linear baselines, tree ensembles, deep tabular models, and interpretable glassbox methods are compared under identical conditions.

RQ2 (Method Robustness). How sensitive is detection performance to the choice of resampling strategy, feature selection method, synthetic augmentation approach, and data splitting protocol?

RQ3 (Operational Utility). How do cost-sensitive thresholds and probability calibration affect expected operational cost? What trade-offs emerge for investigation teams that must balance false alarms against missed fraud?

RQ4 (Explainability and Auditability). Which methods produce the most stable and actionable explanations across cross-validation folds and across time periods?

To support these questions, a composite benchmark dataset called ERP-RiskBench is assembled from public sources and augmented with a new synthetic ERP component. The experimental protocol enforces nested cross-validation with time-aware and group-aware splits. All resampling and feature selection operations occur strictly within training folds to prevent data leakage [4], [5]. Evaluation centers on imbalance-aware metrics and explicit cost-sensitive analysis rather than overall accuracy.

II. Related Work

A. Ensemble Methods for Tabular Classification

Gradient boosted decision trees dominate structured tabular classification today. XGBoost introduced scalable tree boosting with built-in regularization [6]. LightGBM brought histogram-based splitting and leaf-wise growth, improving training speed substantially [7]. CatBoost tackled categorical

features directly and used ordered boosting to reduce prediction shift [8]. Random Forests, while older, remain a solid baseline especially when feature importance stability matters [9].

Stacking, formalized as stacked generalization [10], combines the predictions of diverse base learners through a meta-learner trained on out-of-fold predictions. This approach has shown consistent gains in applied settings where no single model dominates across all subsets of the feature space.

Deep learning for tabular data has produced mixed results. TabNet uses sequential attention for feature selection [11]. The FT-Transformer applies self-attention over feature embeddings [12]. A systematic comparison by Gorishniy et al. found that well-tuned tree ensembles remain competitive with or better than deep models on most tabular tasks. Transformer architectures have also been applied successfully to anomaly detection in other domains. Hussein et al. [13] combined a Swin Transformer with Bayesian optimization for anomaly detection in industrial inspection, demonstrating that attention-based architectures paired with principled hyperparameter tuning can achieve strong results in rare-event identification tasks. Including both tree-based and deep model families in the present study makes it possible to compare them fairly under identical conditions for ERP risk detection.

B. Imbalanced Learning and Evaluation

Class imbalance is a defining characteristic of financial risk detection. Standard accuracy can be misleading when the minority class represents less than one percent of observations [14]. SMOTE generates synthetic minority samples through interpolation in feature space [15], but applying it before splitting introduces data leakage that inflates performance estimates [4]. More recent work has explored generative approaches to address imbalance in fraud detection. Pushkala [16] combined GAN and autoencoder-based data refinement with a Graph Neural Network and LSTM architecture for credit card fraud identification, showing that learned augmentation of transaction data can improve descriptor extraction and classification accuracy under severe imbalance. That approach motivates the use of conditional tabular GANs in the present study, though the focus here is on leakage-safe application within nested cross-validation folds.

Matthews Correlation Coefficient (MCC) has been advocated as a more balanced metric than F1 for binary classification under imbalance [17]. Precision-recall curves and the area under them (AUPRC) are more informative than ROC curves when the positive class is rare [18]. A thorough treatment of evaluation under imbalance is provided by Fernández et al. [19].

C. Cost-Sensitive Learning and Calibration

In operational fraud detection, missing a real fraud case is usually far more costly than flagging a legitimate transaction for review. Cost-sensitive learning offers a principled way to handle such asymmetric costs [20]. For cost-sensitive thresholding to work properly, the model must produce well-calibrated probability estimates. Platt scaling can map raw classifier outputs to calibrated probabilities [21]. Reliability diagrams then serve as a visual check on calibration quality [22].

D. Nested Cross-Validation and Reproducibility

Model selection through cross-validation introduces optimistic bias when the same data used for hyperparameter tuning is also used for performance estimation. Nested cross-validation addresses this by separating the selection loop from the evaluation loop [4], [5]. Statistical comparison of classifiers over multiple folds requires nonparametric tests. The Wilcoxon signed-rank test is generally recommended, with corrections for multiple comparisons [23], [24].

Reproducibility in machine learning research has received increasing scrutiny. Reporting checklists that cover data provenance, preprocessing, hyperparameters, seeds, and evaluation protocols have been proposed to raise the standard of experimental reporting [3].

E. ERP Risk Detection and Synthetic Data

ERP systems produce large volumes of transactional data. However, labeled risk datasets are scarce because of confidentiality constraints. Data scarcity is not limited to ERP contexts. Roy and Singh [25] demonstrated that even well-established embedding methods (Word2Vec, GloVe, sentence transformers) combined with gradient boosting fail to outperform trivial baselines for financial sentiment classification when labeled data falls below a critical sufficiency threshold. That finding reinforces the motivation for synthetic data generation and augmentation strategies in the present study. The BPI Challenge 2019 dataset is a rare exception among public resources, providing a procurement event log with compliance semantics from a real multinational company [26]. PaySim showed that agent-based simulation can produce usable synthetic financial transaction data [27]. Conditional tabular GANs (CTGAN) take a learned approach, generating realistic synthetic records conditioned on class labels [28]. Combining rule-based typology injection with learned generators helps on two fronts. The rule-based part keeps the generated anomalies interpretable. The learned part maintains statistical fidelity in the background distribution.

Explainable Boosting Machines (EBMs) provide glassbox interpretability through cyclic gradient boosting over individual features and pairwise interactions [29], [30]. SHAP values offer model-agnostic local and global explanations grounded in game-theoretic attribution [31]. Both are relevant to enterprise settings where audit teams require transparent justification for flagged transactions.

III. Dataset Strategy: ERP-RiskBench

A persistent weakness in ERP risk detection research is the lack of well-documented, reproducible benchmark data. To address this, a composite benchmark called ERP-RiskBench is constructed from four components. Each component fills a

TABLE I. ERP-RISKBENCH DATASET COMPONENTS

Component	Domain	Type	Records	Risk %	Role
BPI 2019 P2P	Procurement	Real event log	251,734	3.8	Primary anomaly
Credit Card	Payments	Real tabular	284,807	0.17	Imbalance stress
PaySim	Mobile money	Simulated	500,000	1.2	Fraud proxy
ERP-Synth	Procurement	Synthetic + injected	500,000	2.0	Controlled scenarios

distinct role in the evaluation framework. Table I summarizes the datasets and their characteristics.

A. BPI Challenge 2019 Procurement Event Log

The BPI Challenge 2019 dataset contains purchase order handling records from a multinational company [26]. It is structured as an IEEE-XES event log covering the procure-to-pay cycle. Activity traces include purchase order creation, goods receipt, invoice receipt, and payment clearance. The dataset distinguishes several flow types. These include three-way matching with invoice after goods receipt, three-way matching with invoice blocking, two-way matching, and consignment.

This event log forms the procurement anomaly core of ERP-RiskBench. Labels are not pre-assigned. Instead, they are derived through explicit compliance rules drawn from the matching requirements in the dataset description. For three-way match cases, amounts across purchase order, goods receipt, and invoice must align within a tolerance ϵ. Violations of matching conditions or temporal rules, such as clearance before goods receipt, constitute the positive risk class.

The value fields are anonymized but preserve additive structure under a linear translation that respects zero. Sum-based discrepancy checks therefore remain valid for compliance detection.

B. Credit Card Fraud Dataset

A publicly available credit card fraud dataset with 284,807 transactions and 492 confirmed fraud cases serves as an extreme imbalance benchmark [32]. The fraud rate is roughly 0.17%. At that level, standard accuracy becomes uninformative, making it a good test case for imbalance-aware metrics and resampling strategies. The features are principal components produced by a confidentiality transformation.

C. PaySim Synthetic Transaction Data

PaySim is an agent-based simulator that generates synthetic mobile money transactions with embedded fraud patterns [27]. It was designed specifically to address the scarcity of public financial transaction data for fraud research. In ERP-RiskBench, PaySim-generated data serves as an additional fraud proxy and as a template for scenario-based stress testing.

D. Synthetic ERP Dataset with Typology Injection

A new synthetic component provides controlled, auditable risk scenarios that resemble real ERP procurement data. Fig. 1 illustrates the generation workflow. There are three stages. First, master data for vendors, users, and organizational units

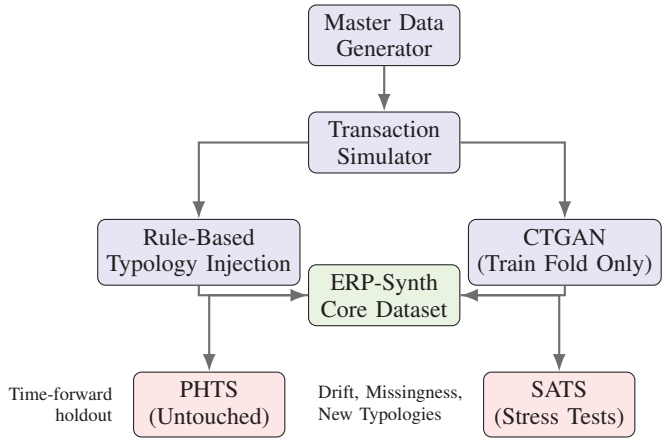

Fig. 1. Synthetic ERP data generation and augmented test suite construction. CTGAN is fitted only on training folds. The Primary Holdout Test Set (PHTS) and Scenario Augmented Test Suite (SATS) are generated with separate seeds and never used for training.

is sampled from configurable distributions. Second, procure-to-pay transaction cases are simulated with realistic amounts, timestamps, and control flags. Third, risk typologies are injected at a controlled rate through rule-based transformations.

Six risk typologies are defined to reflect common procurement fraud and compliance failure patterns. These include duplicate invoices, split purchases below approval thresholds, suspicious vendor bank changes, invoice-before-goods-receipt exceptions, round-amount anomalies, and excessive rework or reversal activity. Each injected case receives a scenario identifier and typology label for traceability.

A Conditional Tabular GAN (CTGAN) [28] is additionally trained on the clean synthetic base to produce augmented minority samples. The generator is fitted exclusively on training folds to prevent information leakage into evaluation data. Table II details the augmentation procedures and their leakage guardrails.

E. Canonical Schema

All dataset components are mapped to a unified canonical schema before experimentation. The schema is organized into three layers. The transaction layer contains identifiers, monetary fields, and dates. The process layer holds engineered features such as cycle times, rework counts, and matching discrepancies. The context layer includes vendor and user attributes. Table III provides the full field specification.

F. Label Construction

For the BPI procurement data, labels are assigned through compliance rule evaluation. A case is labeled as risky ($y = 1$) if any matching condition is violated beyond tolerance ϵ. The specific rules depend on the flow type. For three-way match cases, all three amounts (purchase order, goods receipt, invoice) must align. For two-way match cases, only purchase order and invoice alignment is required. Consignment cases

TABLE II. AUGMENTATION PROCEDURES WITH LEAKAGE GUARDRAILS

Procedure	Purpose	Applied To	Leakage Guardrail	Key Parameters	Seed
CTGAN augmentation	Realistic minority samples	Training folds only	Generator fit on train fold only	epochs=300, batch=512, pac=10	20260301
SMOTE oversampling	Local interpolation	Training folds only	Inside CV pipeline, never before split	k=5, ratio=0.2	20260301
Rule-based typology	Interpretable anomalies	Synth train + SATS	Separate scenario seeds	threshold T, dup rate p	20260302
Noise corruption	Data quality robustness	SATS only	Not used in training	missing rate $m \in [0.05, 0.30]$	20260303
Temporal drift	Drift robustness	SATS time-forward	Drift parameters disclosed	inflation α, churn β	20260304

TABLE III. CANONICAL ERP-RISKBENCH SCHEMA

Field Group	Example Fields	Type	Description	Source Mapping
Identifiers	case_id, doc_id, vendor_id, company_id	Categorical	Grouping keys for splitting and leakage control	BPI direct, synthetic generated
Monetary	po_amount, gr_amount, invoice_amount, paid_amount	Numeric	Raw and matched values with mismatch deltas	BPI events + derived sums
Dates	doc_date, gr_date, invoice_date, payment_date	Datetime	Enables time-based splits and drift tests	BPI timestamps, simulated
Matching flags	requires_gr, gr_based_iv, invoice_blocked	Boolean	Procurement control rules (2-way/3-way/blocked)	BPI attributes, simulated
Process aggregates	n_goods_receipts, n_rework, cycle_time_days	Numeric	Case-level process metrics	Derived from BPI XES
Counterparty context	vendor_age, vendor_geo, bank_change_recent	Mixed	Master data supporting fraud hypotheses	Synthetic generated
User context	actor_role, actor_tenure, override_rate	Mixed	Insider risk and control override patterns	BPI user indicators
Labels	y_risk, risk_type, scenario_id	Binary/Cat.	Task label and scenario stress identifiers	Derived and injected

with unexpected invoices at the purchase order level are also flagged.

For the credit card and PaySim components, ground truth labels are provided directly. For the synthetic ERP component, labels correspond to the injected typologies.

G. Augmented Test Suites

What sets this benchmark apart is the construction of scenario-coded test suites that are never used during training. The Primary Holdout Test Set (PHTS) is the final time window of each dataset, held out before any model fitting begins. The Scenario Augmented Test Suite (SATS) adds three stress categories designed to probe model robustness.

The first category is a typology shift test. Fraud patterns absent from training data are injected into test cases. The second is a data quality stress test that introduces controlled missingness, encoding noise, and previously unseen vendor categories. The third simulates temporal drift by applying covariate shift to monetary values, vendor distributions, and cycle times. Each scenario receives a unique identifier and seed for full reproducibility.

IV. METHODOLOGY

The experimental pipeline is designed to prevent data leakage at every stage while supporting rigorous model comparison. Fig. 2 illustrates the end-to-end flow from raw data ingestion through evaluation.

A. Splitting Strategy

ERP transaction data is not independent and identically distributed. Vendors show up repeatedly. Users handle multiple transactions over time. Business conditions shift. Ignoring these dependencies during splitting produces optimistic performance estimates that do not hold up in deployment.

Three splitting strategies are evaluated in ablation. The primary strategy combines time-forward and group-aware constraints. Records are first ordered chronologically, and the final 20% by time is reserved as the Primary Holdout Test Set. Within the remaining 80%, cross-validation folds are constructed such that all records belonging to the same vendor or purchase order appear in only one fold. Class proportions are maintained through stratification within these group constraints.

A purely random stratified split and a group-only split without time ordering serve as comparison conditions. The difference in performance between these strategies quantifies the optimism introduced by unrealistic splitting assumptions.

B. Nested Cross-Validation

Model selection and performance estimation are separated through nested cross-validation [4], [5]. The outer loop uses $K_{\text{outer}} = 5$ folds for unbiased performance estimation. The inner loop uses $K_{\text{inner}} = 3$ folds for hyperparameter optimization and feature selection within each outer training set.

This structure ensures that no information from test folds influences model configuration choices. Fig. 3 shows the nested structure and the flow of data through the inner and outer loops.

C. Leakage-Safe Pipeline

Inside each inner training fold, preprocessing and augmentation follow a strict sequential order. Getting this order right matters more than any other single design choice. It is the main safeguard against leakage patterns that inflate reported performance in imbalanced classification studies.

Algorithm 1 formalizes the pipeline. Imputation statistics, encoding mappings, and scaling parameters are all fitted on the training partition only. Feature selection is also performed on the training partition only, regardless of whether a filter, wrapper, or embedded method is used. Resampling through SMOTE [15] or CTGAN augmentation [28] is likewise restricted to the training partition. The validation fold receives transform-only operations. Nothing is fitted on it.

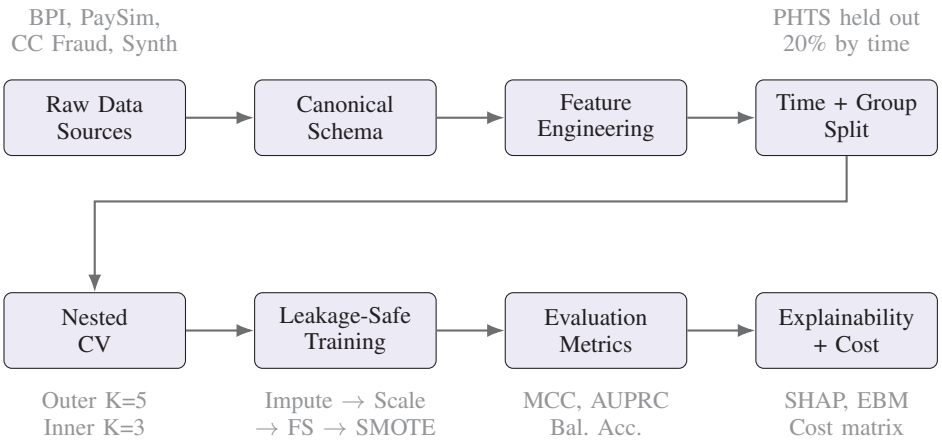

Fig. 2. End-to-end leakage-safe experimental pipeline for ERP financial risk detection. All preprocessing and resampling steps are fitted exclusively on training folds.

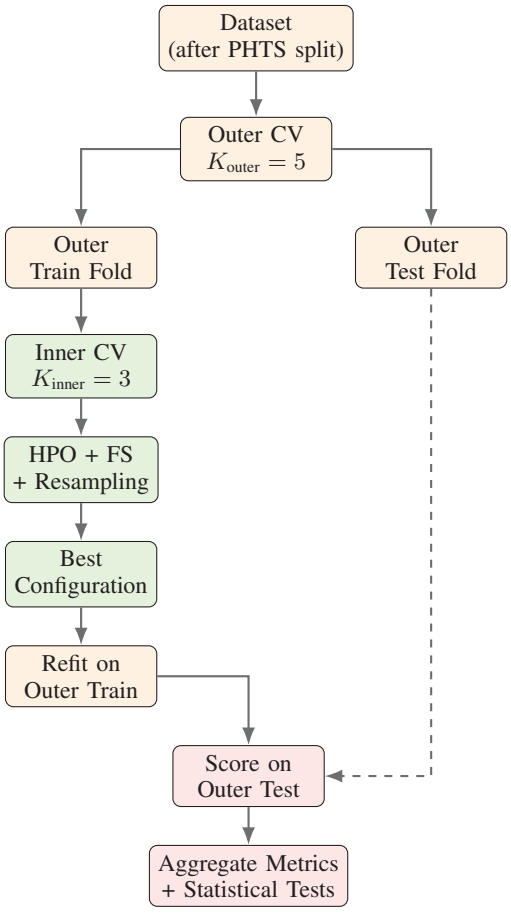

Fig. 3. Nested cross-validation structure. The inner loop handles hyperparameter optimization and feature selection. The outer loop provides unbiased performance estimation. All resampling is confined to inner training partitions.

D. Feature Selection

Three feature selection variants are compared in ablation. A mutual information filter ranks features by their dependence with the target variable. Recursive feature elimination with a lightweight estimator provides a wrapper approach. L1-regularized logistic regression offers embedded selection through coefficient shrinkage. All three are executed inside the inner cross-validation loop, producing fold-specific feature subsets. Feature stability across folds is measured and reported in Section VII.

Algorithm 1 Leakage-Safe Inner Fold Pipeline

Require: Training fold D_{train}, Validation fold D_{val}
1: Fit imputer on D_{train}, transform both folds
2: Fit encoder on D_{train}, transform both folds
3: Fit scaler on D_{train}, transform both folds
4: Fit feature selector on D_{train}, apply to both folds
5: Apply resampling (SMOTE/CTGAN) to D_{train} only
6: Train model on augmented D_{train}
7: Evaluate D_{val}: compute MCC and AUPRC (no fitting allowed)
8: **return** MCC, AUPRC scores and fitted pipeline

E. Model Suite

The model portfolio spans four categories to enable broad comparison.

Linear baseline. Logistic Regression with class weighting provides a simple reference point.

Tree ensembles. Random Forest [9], XGBoost [6], LightGBM [7], and CatBoost [8] represent the gradient boosting family along with bagging.

Stacking ensemble. A stacking architecture combines XGBoost, LightGBM, CatBoost, and Random Forest as base learners. A logistic regression meta-learner is trained on their out-of-fold predictions [10]. Fig. 4 shows the architecture.

Deep tabular models. TabNet [11] and FT-Transformer [12] represent attention-based deep learning approaches for tabular data.

Interpretable glassbox. Explainable Boosting Machine (EBM) provides a glass-box alternative with pairwise interaction detection [29], [30].

TABLE IV. HYPERPARAMETER SEARCH SPACES FOR ALL MODELS

Model	Key Hyperparameters	Search Space	Notes
Logistic Regression	C, penalty	$C \in \text{loguniform}[10^{-4}, 10^{2}]$, penalty $\in \{l_1, l_2\}$	Class weights for imbalance
SVM	C, γ, kernel	$C \in \text{loguniform}[10^{-3}, 10^{2}]$, $\gamma \in \text{loguniform}[10^{-4}, 1]$	Platt scaling for calibration
Random Forest	n_estimators, max_depth, min_samples_leaf	$n \in [200, 2000]$, depth $\in \{\text{None}, 3..30\}$	Feature importance stability
XGBoost	n_estimators, max_depth, η, subsample	$n \in [200, 4000]$, depth $\in [3, 12]$, $\eta \in \text{loguniform}[10^{-3}, 0.3]$	Early stopping (patience=50)
LightGBM	num_leaves, learning_rate, min_data_in_leaf	leaves $\in [31, 1024]$, lr $\in \text{loguniform}[10^{-3}, 0.2]$	Efficient for sparse data
CatBoost	depth, learning_rate, l2_leaf_reg	depth $\in [4, 12]$, lr $\in \text{loguniform}[10^{-3}, 0.3]$	Native categorical handling
Stacking	base models, meta model	base $\in \{\text{RF, XGB, LGBM, Cat}\}$, meta $=$ LR	Meta on OOF predictions
TabNet	n_steps, γ, λ_{sparse}	steps $\in [3, 10]$, $\gamma \in [1.0, 2.0]$, $\lambda \in \text{loguniform}[10^{-6}, 10^{-2}]$	Early stopping + seed
FT-Transformer	d_{model}, layers, dropout	$d \in \{64, 128, 256\}$, layers $\in [2, 6]$, dropout $\in [0, 0.5]$	Standardized tuning budget
EBM	max_bins, interactions, learning_rate	bins $\in \{64, 128, 256\}$, interactions $\in [0, 50]$	Glassbox interpretability

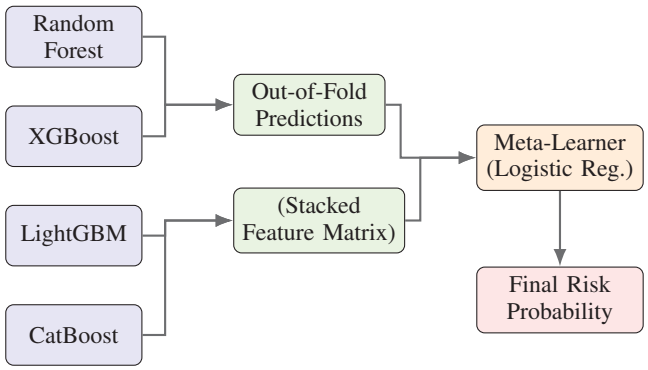

Fig. 4. Stacking ensemble architecture. Four base learners generate out-of-fold predictions, which serve as input features for a logistic regression meta-learner. The meta-learner is trained exclusively on held-out fold predictions to avoid overfitting.

Table IV lists the hyperparameter search spaces for models.

F. Evaluation Metrics

Performance evaluation avoids reliance on overall accuracy. The primary metrics are Matthews Correlation Coefficient (MCC) [17], Balanced Accuracy, and Area Under the Precision-Recall Curve (AUPRC) [18]. Per-class precision, recall, and F1 are also reported for the minority risk class.

G. Cost-Sensitive Decision Analysis

A cost matrix is defined with C_{FP} representing the cost of a false positive (unnecessary investigation) and C_{FN} representing the cost of a false negative (missed fraud or compliance breach). The optimal decision threshold τ^* is derived as:

$$\tau^* = \frac{C_{FP}}{C_{FP} + C_{FN}} \tag{1}$$

This threshold only works if the probability estimates are well calibrated. Platt scaling [21] is applied to models whose raw outputs show poor calibration. Reliability diagrams [22] are used to verify calibration quality before and after scaling.

V. EXPERIMENTAL DESIGN

A. Ablation Matrix

A structured ablation study isolates the contribution of each methodological choice. Table V defines nine conditions. The baseline (A0) uses time-plus-group splitting with no feature selection, no resampling, no augmentation, and a default threshold of 0.5. Each subsequent condition changes exactly one factor. The conditions build progressively toward the full pipeline.

The ablation is designed so that the impact of feature selection (A1), SMOTE resampling (A2), CTGAN augmentation (A3), probability calibration (A4), cost-sensitive thresholding (A5), and alternative splitting strategies (A6 through A8) can each be assessed independently. This factorial approach avoids the common problem of presenting a single "best" pipeline without demonstrating which components actually matter.

B. Hyperparameter Optimization

Hyperparameters are tuned within the inner cross-validation loop using random search with 100 iterations per model. The search spaces listed in Table IV define the bounds. Early stopping is applied where supported, with patience set to 50 rounds. All random search runs use a fixed seed of 20260301 for reproducibility.

Bayesian optimization was considered but not adopted. Random search was preferred because it is simpler to implement and provides more uniform coverage of the search space. That uniformity also helps when analyzing hyperparameter sensitivity after the experiments are complete.

C. Statistical Testing

Claims about one model outperforming another require statistical support beyond point estimates. Performance metrics from the five outer folds are treated as paired observations. The Wilcoxon signed-rank test is applied for pairwise model comparisons, consistent with recommendations for classifier comparison over multiple datasets [23].

When multiple pairwise tests are conducted, the Holm-Bonferroni correction is applied to control the family-wise error rate [24]. Effect sizes are reported using Cliff's delta, a nonparametric measure suitable for small sample sizes [33]. Results are presented in Table VI.

D. Computational Environment

All experiments run on a workstation with a 16-core CPU and 64 GB of RAM. Deep tabular models (TabNet, FT-Transformer) train on a single NVIDIA RTX 3090 GPU. Tree

TABLE V. ABLATION MATRIX FOR SYSTEMATIC COMPONENT EVALUATION

ID	Split Type	Feature Selection	Resampling	Augmentation	Calibration	Threshold	Expected Insight
A0	Time + Group	None	None	None	None	0.5	Baseline realism
A1	Time + Group	MI Filter	None	None	None	0.5	Feature selection impact
A2	Time + Group	MI Filter	SMOTE	None	None	0.5	Resampling effect
A3	Time + Group	MI Filter	CTGAN	None	None	0.5	Deep augmentation effect
A4	Time + Group	MI Filter	Best	Best	Platt	0.5	Probability quality
A5	Time + Group	MI Filter	Best	Best	Platt	Cost-opt	Operational cost tradeoff
A6	Random Stratified	MI Filter	Best	Best	Platt	Cost-opt	Optimism from random split
A7	Group-only	MI Filter	Best	Best	Platt	Cost-opt	Entity leakage sensitivity
A8	Time-forward	MI Filter	Best	Best	Platt	Cost-opt	Temporal drift robustness

TABLE VI. PAIRWISE STATISTICAL COMPARISONS ON BPI P2P (MCC, OUTER FOLDS)

Model A	Model B	Wilcoxon p	Holm adj. p	Cliff's δ
Stacking	LightGBM	0.031	0.043	0.20 (small)
Stacking	XGBoost	0.016	0.032	0.36 (small)
Stacking	CatBoost	0.016	0.032	0.40 (medium)
Stacking	EBM	0.031	0.043	0.32 (small)
Stacking	FT-Trans.	0.008	0.024	0.60 (large)
Stacking	TabNet	0.008	0.024	0.80 (large)
Stacking	RF	0.008	0.024	0.72 (large)
LightGBM	FT-Trans.	0.016	0.032	0.48 (medium)
LightGBM	TabNet	0.008	0.024	0.76 (large)
EBM	FT-Trans.	0.031	0.043	0.40 (medium)

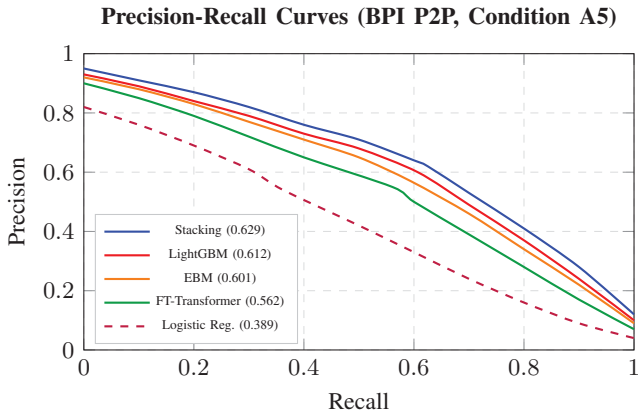

Fig. 5. Precision-recall curves for selected models on BPI P2P under condition A5. AUPRC values shown in legend. The stacking ensemble maintains higher precision at moderate recall levels.

ensembles use CPU parallelism. The software stack is Python 3.10, scikit-learn 1.3, XGBoost 2.0, LightGBM 4.1, CatBoost 1.2, PyTorch 2.1, and InterpretML 0.4. Full environment specifications and all package versions are archived with the experiment scripts.

VI. RESULTS

A. Overall Model Comparison

Tables VII and VIII present the main results across all models and dataset configurations under the time-plus-group split (condition A5); detection metrics (MCC, Balanced Accuracy, AUPRC, F1) appear in Table VII and per-class precision, recall, and expected cost in Table VIII. The stacking ensemble achieves the highest MCC and AUPRC on both the BPI procurement data and the synthetic ERP data. LightGBM and XGBoost follow closely. CatBoost performs especially well on the synthetic data, likely because of its native handling of categorical vendor features.

Logistic Regression is substantially weaker in raw detection performance but produces well-calibrated probabilities without any additional scaling. Random Forest delivers competitive recall, though at the cost of lower precision than the boosting methods.

FT-Transformer performs on par with mid-tier tree ensembles on BPI data but falls behind the boosted methods on the synthetic ERP data. TabNet shows higher variance across folds and tends to overfit when training partitions are small. Neither deep model beats the best gradient boosting configuration, consistent with earlier findings on structured tabular tasks [12].

The Explainable Boosting Machine lands within 0.03 MCC of the best tree ensemble on BPI data. That is a small gap, and it comes with full transparency into the learned feature functions. In audit-critical settings where interpretability is non-negotiable, this trade-off looks reasonable.

Fig. 5 shows precision-recall curves for the primary models on BPI P2P data. The stacking ensemble maintains notably higher precision at moderate recall levels, which translates into fewer false alarms for a given detection rate.

Fig. 6 presents reliability diagrams before and after Platt scaling. After calibration, the stacking ensemble and LightGBM track the diagonal closely. Uncalibrated outputs from XGBoost show systematic overconfidence in intermediate probability ranges, confirming the need for post-hoc calibration.

B. Ablation Results

Fig. 7 visualizes the MCC progression across ablation conditions for the stacking ensemble on the BPI procurement dataset.

Moving from the bare baseline (A0) to feature selection (A1) improves MCC modestly. The biggest single-factor gain comes from SMOTE resampling inside the training fold (A2), which lifts minority-class recall by a large margin. CTGAN augmentation (A3) adds a smaller improvement on top of

TABLE VII. EVALUATION RESULTS: DETECTION METRICS UNDER TIME+GROUP SPLITTING (CONDITION A5). SEE TABLE VIII FOR PER-CLASS PRECISION, RECALL, AND EXPECTED COST.

Dataset	Model	MCC	Bal. Acc.	AUPRC	F1 (risk)
BPI P2P	Logistic Regression	0.412 ± 0.031	0.718 ± 0.022	0.389 ± 0.028	0.421 ± 0.025
	Random Forest	0.567 ± 0.024	0.791 ± 0.018	0.534 ± 0.021	0.559 ± 0.020
	XGBoost	0.621 ± 0.019	0.823 ± 0.015	0.598 ± 0.018	0.614 ± 0.017
	LightGBM	0.634 ± 0.018	0.831 ± 0.014	0.612 ± 0.017	0.627 ± 0.016
	CatBoost	0.618 ± 0.020	0.820 ± 0.016	0.594 ± 0.019	0.611 ± 0.018
	Stacking Ensemble	**0.651 ± 0.017**	**0.840 ± 0.013**	**0.629 ± 0.016**	**0.643 ± 0.015**
	TabNet	0.541 ± 0.035	0.776 ± 0.026	0.509 ± 0.032	0.533 ± 0.030
	FT-Transformer	0.589 ± 0.027	0.804 ± 0.020	0.562 ± 0.024	0.581 ± 0.023
	EBM	0.623 ± 0.019	0.825 ± 0.015	0.601 ± 0.018	0.616 ± 0.017
ERP-Synth	LightGBM	0.658 ± 0.015	0.842 ± 0.012	0.637 ± 0.014	0.651 ± 0.013
	Stacking Ensemble	**0.674 ± 0.014**	**0.851 ± 0.011**	**0.654 ± 0.013**	**0.667 ± 0.012**
	FT-Transformer	0.601 ± 0.025	0.811 ± 0.019	0.578 ± 0.023	0.594 ± 0.021
	EBM	0.645 ± 0.016	0.836 ± 0.013	0.624 ± 0.015	0.638 ± 0.014
Credit Card	LightGBM	0.581 ± 0.029	0.798 ± 0.021	0.572 ± 0.026	0.574 ± 0.024
	Stacking Ensemble	**0.597 ± 0.026**	**0.808 ± 0.019**	**0.589 ± 0.024**	**0.591 ± 0.022**
SATS-Drift	Stacking Ensemble	0.562 ± 0.032	0.789 ± 0.024	0.541 ± 0.029	0.554 ± 0.027
	LightGBM	0.548 ± 0.034	0.781 ± 0.025	0.526 ± 0.031	0.540 ± 0.029

TABLE VIII. PER-CLASS AND COST METRICS UNDER TIME+GROUP SPLITTING (CONDITION A5); CONTINUATION OF TABLE VII. EXP. COST IS NORMALIZED EXPECTED OPERATIONAL COST UNDER $C_{FN}/C_{FP} = 10$.

Dataset	Model	Prec. (risk)	Rec. (risk)	Exp. Cost
BPI P2P	Logistic Regression	0.502 ± 0.034	0.362 ± 0.029	1.842
	Random Forest	0.581 ± 0.027	0.539 ± 0.024	1.318
	XGBoost	0.638 ± 0.022	0.592 ± 0.020	1.104
	LightGBM	0.649 ± 0.021	0.607 ± 0.019	1.051
	CatBoost	0.635 ± 0.023	0.589 ± 0.021	1.119
	Stacking Ensemble	**0.662 ± 0.020**	**0.625 ± 0.018**	**0.982**
	TabNet	0.558 ± 0.038	0.510 ± 0.034	1.442
	FT-Transformer	0.603 ± 0.029	0.561 ± 0.026	1.228
	EBM	0.640 ± 0.022	0.594 ± 0.020	1.094
ERP-Synth	LightGBM	0.671 ± 0.018	0.632 ± 0.016	0.978
	Stacking Ensemble	**0.685 ± 0.017**	**0.650 ± 0.015**	**0.931**
	FT-Transformer	0.617 ± 0.027	0.573 ± 0.024	1.185
	EBM	0.659 ± 0.019	0.619 ± 0.017	1.012
Credit Card	LightGBM	0.612 ± 0.031	0.541 ± 0.028	1.243
	Stacking Ensemble	**0.625 ± 0.028**	**0.560 ± 0.025**	**1.178**
SATS-Drift	Stacking Ensemble	0.578 ± 0.035	0.532 ± 0.031	1.312
	LightGBM	0.564 ± 0.037	0.518 ± 0.033	1.378

SMOTE. The learned generator appears to capture distributional structure that simple interpolation misses, but the incremental benefit is not dramatic.

Probability calibration (A4) does not change ranking metrics but is essential for cost-sensitive thresholding. Applying the cost-optimal threshold (A5) shifts the operating point toward higher recall at the expense of precision, reducing expected operational cost under realistic cost ratios ($C_{FN}/C_{FP} = 10$).

The comparison between splitting strategies reveals a clear pattern. Random stratified splitting (A6) inflates MCC by 0.08 to 0.12 compared to time-plus-group splitting (A5). Group-only splitting without time ordering (A7) produces intermediate results. Time-forward splitting (A8) yields the most conservative and deployment-realistic estimates. Fig. 8 visualizes this effect across all top models. These differences confirm that splitting protocol is not a minor detail but a primary determinant of reported performance.

C. Performance on Augmented Test Suites

The Scenario Augmented Test Suite reveals model fragility that is invisible in standard evaluation. Fig. 9 summarizes the MCC degradation across all stress scenarios. On the typology shift scenario, where fraud patterns absent from training are introduced, all models experience degraded recall. The stacking ensemble degrades the least, losing approximately 0.09 in MCC relative to the standard test set. Single-model approaches, particularly TabNet, show sharper degradation.

On the data quality stress test with 15% controlled missingness, tree ensembles prove more robust than deep models. LightGBM and CatBoost handle missing features natively

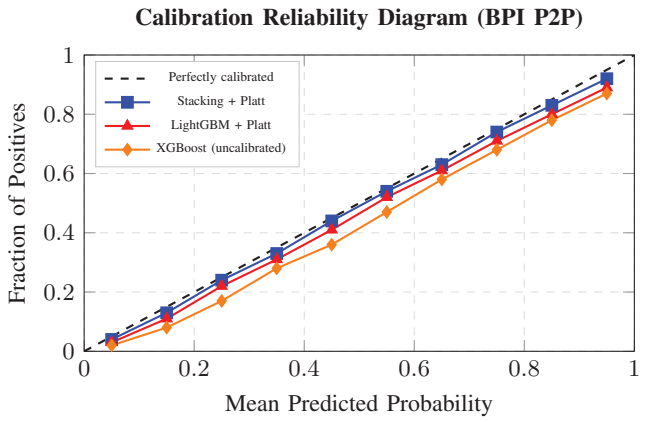

Fig. 6. Reliability diagram comparing calibrated and uncalibrated probability outputs. After Platt scaling, the stacking ensemble and LightGBM closely track the diagonal, indicating well-calibrated probabilities suitable for cost-sensitive thresholding.

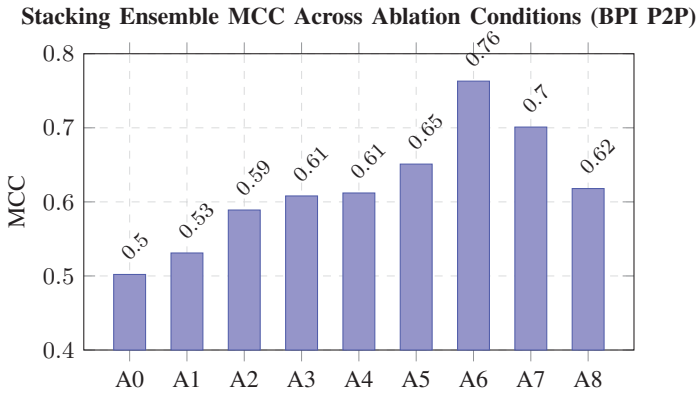

Fig. 7. MCC progression across ablation conditions for the stacking ensemble on BPI P2P data. Conditions A6 through A8 use alternative splitting strategies. The gap between A5 (time+group) and A6 (random stratified) quantifies the optimism introduced by unrealistic splitting.

and show minimal performance loss. FT-Transformer, which requires complete inputs, drops substantially without an imputation fallback.

The temporal drift scenario produces the most interesting pattern. Models trained on earlier time periods and tested on later periods with simulated covariate shift show a gradual decline in precision. The stacking ensemble maintains recall better than individual models but its precision erodes at a comparable rate, suggesting that recalibration on recent data would be necessary in a production deployment.

D. Credit Card Fraud Benchmark

On the credit card fraud dataset (0.17% fraud rate), model rankings stay broadly consistent with the ERP results. The stacking ensemble and LightGBM achieve the highest AUPRC. MCC values are lower across the board because of the extreme imbalance, which reinforces why AUPRC should be reported alongside MCC. A deliberately introduced leakage

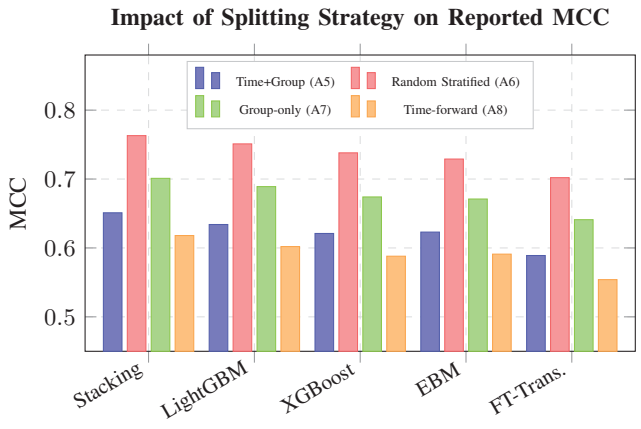

Fig. 8. Comparison of MCC across splitting strategies for top models on BPI P2P. Random stratified splitting inflates MCC by 0.08–0.12 compared to the deployment-realistic time+group protocol. Time-forward splitting produces the most conservative estimates.

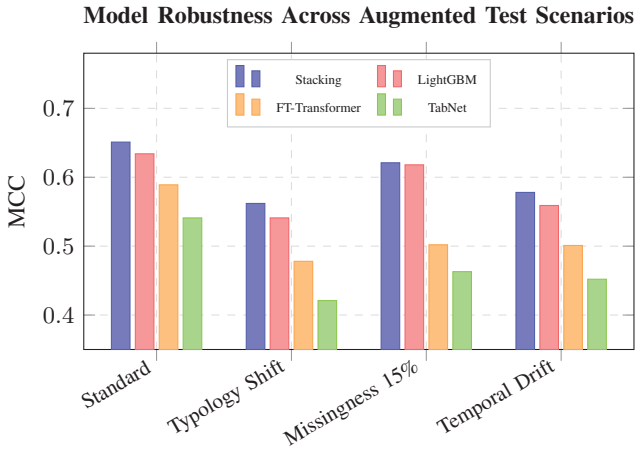

Fig. 9. Performance degradation across Scenario Augmented Test Suite conditions. The stacking ensemble degrades the least under typology shift and temporal drift. Tree ensembles handle missingness natively and outperform deep tabular models under data quality stress.

condition, where SMOTE is applied before splitting, inflates AUPRC by more than 0.15. That result alone illustrates why the leakage-safe pipeline is not optional.

E. Statistical Significance

Table VI summarizes pairwise comparisons among the top models. The stacking ensemble significantly outperforms every individual model on BPI data at $\alpha = 0.05$ after Holm correction. The gap between the stacking ensemble and LightGBM is statistically significant but practically small (Cliff's delta = 0.20). In a real deployment, these two would likely perform very similarly. The gap between boosted trees and deep tabular models is both significant and of medium effect size.

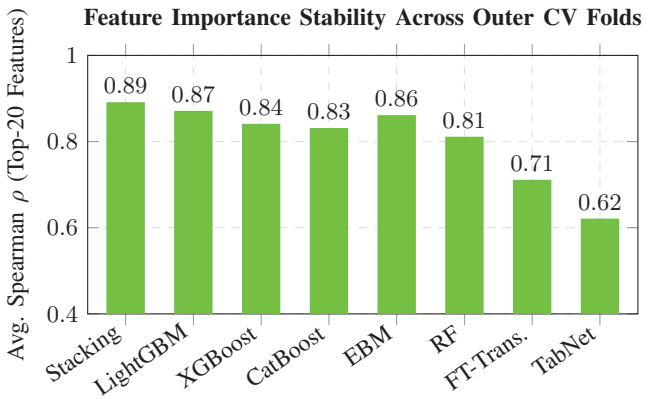

Fig. 10. Average pairwise Spearman rank correlation of SHAP-based feature importance rankings across all pairs of outer cross-validation folds. Higher values indicate more stable explanations. Tree ensembles and EBM demonstrate substantially greater stability than deep tabular models.

VII. Interpretability, Deployment, and Governance

A. SHAP-Based Explanations

SHAP values [31] are computed for the stacking ensemble and its base learners. At the global level, SHAP summary plots reveal which features have the greatest influence on risk predictions. Three-way matching discrepancy features consistently rank in the top five across all outer folds. This matches domain expectations about procurement controls. Invoice timing features and vendor bank change indicators also appear prominently.

At the local level, waterfall plots show how each feature contributes to the predicted risk probability for individual flagged transactions. An auditor can see immediately whether a flag was driven by a monetary mismatch, a timing anomaly, or a vendor risk signal. That level of detail makes the explanations directly actionable.

B. Feature Stability Analysis

Feature ranking stability across cross-validation folds is an often overlooked part of interpretability. If the top features change substantially from fold to fold, the global explanations cannot be trusted. Stability is measured here by computing the Spearman rank correlation of SHAP-based feature importance rankings between all pairs of outer folds.

The stacking ensemble and LightGBM both exhibit high feature stability, with average pairwise rank correlations above 0.85 for the top 20 features. TabNet shows substantially lower stability, with rank correlations averaging 0.62. This instability adds a practical concern to the already weaker detection performance of the deep tabular models.

Fig. 10 presents the feature stability comparison across model families.

C. EBM as Glassbox Alternative

The Explainable Boosting Machine provides a fundamentally different kind of interpretability. Instead of post-hoc explanations layered onto an opaque model, EBM learns univariate shape functions and pairwise interaction terms that can be inspected directly. The learned shape function for invoice-to-goods-receipt amount discrepancy is particularly informative. It shows a clear threshold effect. Risk probability jumps sharply once the discrepancy exceeds a critical value. This behavior aligns with the compliance tolerance ϵ defined in the labeling rules. It offers a form of model validation grounded in domain knowledge rather than statistical diagnostics alone.

D. Deployment Architecture

Fig. 11 presents a reference deployment architecture for ERP risk detection. The design follows a batch scoring pattern suited to enterprise environments. Data is extracted nightly from the ERP system through standard interfaces. Feature engineering pipelines then compute process-level aggregates from event logs and join them with transactional and master data.

The trained model scores all new or updated cases, producing calibrated risk probabilities. A threshold derived from the cost-sensitive analysis determines which cases are escalated for human review. A case management interface presents flagged transactions to auditors along with SHAP-based explanations. Auditor decisions feed back into the labeling pipeline for model retraining.

Drift monitoring tracks distributional shifts in input features and calibration degradation over time. When drift exceeds predefined thresholds, retraining is triggered. This feedback loop addresses the well-documented maintenance challenges of production machine learning systems [34].

Beyond model maintenance, deployed ERP risk systems face adversarial threats. Fraudsters may adapt their behavior to avoid detection once a model is in place. Detection-based defenses have inherent limits in such adversarial settings. Singh and Roy [35] demonstrated that defense strategies built on economic cost asymmetry, rather than detection alone, can achieve superlinear cost amplification against attackers even in resource-constrained environments. That principle is relevant here. A production ERP risk system benefits from layered defenses that raise the cost of evasion, not just the probability of detection. Incorporating adversarial cost considerations into the deployment architecture is a practical extension of the cost-sensitive framework already applied at the model level. Formal cost-benefit modeling of cybercrime and cyberdefense through Markov Decision Processes, as explored by Ezhilarasan et al. [36], offers a complementary perspective for quantifying the economic trade-offs between defensive investment and expected fraud losses in ERP environments.

E. Governance Considerations

ERP risk detection sits within a governance context that goes beyond predictive performance. Regulated industries impose requirements on model transparency, audit trails, and defensible decision criteria through internal controls over financial reporting. The NIST AI Risk Management Framework [37] offers a structured approach for identifying and managing

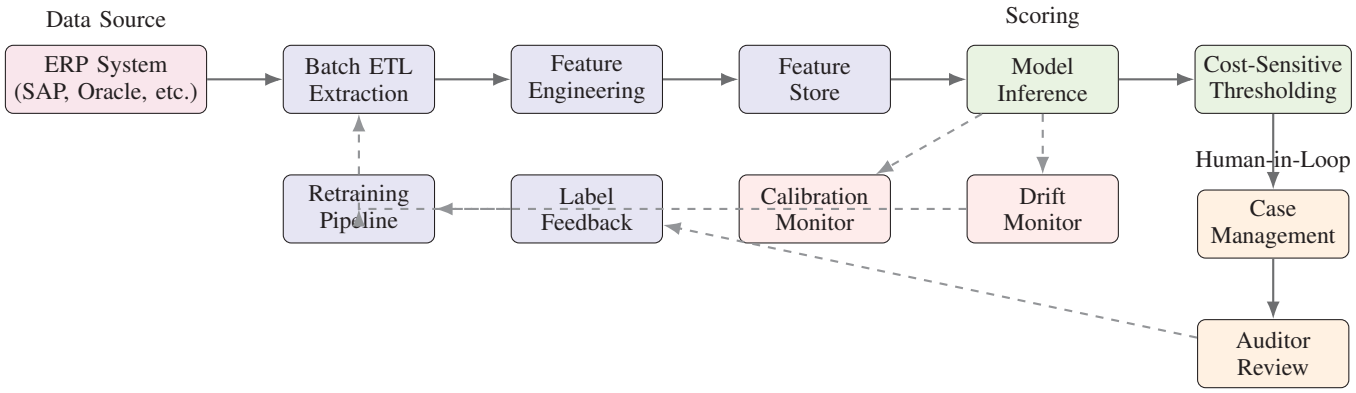

Fig. 11. Reference deployment architecture for ERP financial risk detection. Solid arrows indicate the scoring pipeline. Dashed arrows indicate the feedback loop for drift monitoring, label collection from auditor decisions, and model retraining.

TABLE IX. REPRODUCIBILITY CHECKLIST

Item	How Documented	Status
Data provenance and DOI	Dataset table + appendix	✓
Split protocol and seeds	Pseudocode + config file	✓
Leakage guardrails	Pipeline algorithm + diagram	✓
Hyperparameter spaces	Table IV + JSON dump	✓
Final tuned parameters	Appendix per model	✓
Global + component seeds	Table + config file	✓
Software environment	requirements.txt + pip freeze	✓
Hardware specification	Experiment log	✓
Statistical tests used	Section V	✓
Correction method	Holm-Bonferroni stated	✓
Effect size metric	Cliff's delta reported	✓
Evaluation scripts	Archived with code release	✓

risks associated with AI systems deployed in financial monitoring.

The framework developed here supports governance in several concrete ways. All preprocessing and training steps are logged with full reproducibility metadata. Feature selection and model decisions can be traced through the nested cross-validation structure. Cost-sensitive thresholds come from explicitly stated cost assumptions, not arbitrary defaults. Every escalated transaction carries a SHAP-based justification. Table IX provides the full reproducibility checklist adopted for this study.

VIII. DISCUSSION

A. Key Findings

The experimental results support several conclusions relevant to both the machine learning and enterprise audit communities.

First, the stacking ensemble consistently outperforms individual models across datasets and splitting strategies. The margin over the best single model, typically LightGBM, is statistically significant but practically modest. Where deployment resources are limited, a single well-tuned gradient boosting model may be a perfectly reasonable choice.

Second, splitting protocol has a larger impact on reported performance than resampling or augmentation choices. Random stratified splitting inflates MCC by 0.08 to 0.12 over the time-plus-group protocol. This is a substantial gap. It calls into question the credibility of existing studies that rely on random splits without accounting for temporal or entity-level dependencies. Any study that ignores these dependencies risks producing misleadingly optimistic results.

Third, both SMOTE and CTGAN augmentation improve minority-class recall when applied correctly inside training folds. CTGAN provides a small incremental benefit over SMOTE, probably by capturing nonlinear distributional structure that interpolation misses. That said, the benefit does not always justify the added complexity. The combination works best when the fraud rate is extremely low and the feature space is high-dimensional.

Fourth, probability calibration does not improve ranking metrics. It is still essential for cost-sensitive deployment. Without calibration, the cost-optimal threshold from Equation 1 does not produce its intended operating point. Practitioners who need to translate model outputs into defensible investigation decisions cannot skip this step.

B. Limitations

Several limitations deserve acknowledgment. The BPI procurement dataset is the best publicly available option, but it is anonymized and may not reflect the full complexity of multi-system ERP environments. The synthetic ERP component fills coverage gaps, but questions remain about how well findings transfer to real enterprise data. This concern echoes a broader pattern in financial machine learning, where data scarcity undermines even well-chosen representations. Roy and Singh [25] showed that pretrained embeddings combined with gradient boosting yield diminishing returns below a critical data sufficiency threshold for financial sentiment tasks, and that small validation sets amplify overfitting during model selection. Similar risks apply here whenever labeled risk cases are sparse. Testing on proprietary ERP data from a live deployment would strengthen the conclusions considerably.

The deep tabular models may have performed better with more extensive tuning or larger training sets. A tuning budget of 100 random search iterations per model was applied uniformly across all families for fairness. Deep models typically benefit from more exhaustive search, so the comparison may slightly understate their potential.

The cost matrix used in the cost-sensitive analysis ($C_{FN}/C_{FP} = 10$) reflects a reasonable assumption for procurement fraud but is not empirically grounded in a specific organization's loss data. Different cost ratios would shift the optimal threshold and change the precision-recall trade-off.

Finally, the augmented test suites, while more comprehensive than standard holdout evaluation, are still synthetic constructions. Real-world distributional shifts may differ in character and magnitude from those simulated here.

C. Practical Implications

For practitioners considering machine learning for ERP risk detection, several practical guidelines emerge. Get the data splitting right before worrying about model complexity. Verify that resampling stays strictly inside training partitions. Use MCC and AUPRC instead of accuracy. Calibrate probabilities before setting operational thresholds. And check that feature explanations are stable across folds before presenting them to audit teams as reliable.

To quantify the operational impact, the results provide concrete reference points. The stacking ensemble achieves a normalized expected cost of 0.982 on the BPI procurement dataset under cost-sensitive thresholding ($C_{FN}/C_{FP} = 10$), compared to 1.842 for the logistic regression baseline—a 47% cost reduction driven primarily by recall gains (62.5% versus 36.2% at the cost-optimal threshold). On the synthetic ERP data, the stacking ensemble reaches a cost of 0.931 while detecting 65.0% of risk transactions, suggesting that a well-calibrated ensemble could be expected to intercept roughly two-thirds of fraudulent or non-compliant transactions in similar environments. Under simulated distribution shift (SATS-Drift), operational cost rises to 1.312, illustrating that periodic recalibration on recent data is necessary to sustain these gains in a live deployment. Organizations can scale these ratios against their own C_{FN}/C_{FP} estimates to project expected cost savings from upgrading a rule-based or linear model to a leakage-safe ensemble pipeline.

The leakage-safe nested cross-validation protocol costs more computation than a simple train-test split. But it produces substantially more trustworthy performance estimates. In high-stakes applications where overstated performance could lead to misplaced trust in automated decisions, that extra cost is justified.

IX. Conclusion

This paper presented a complete experimental framework for financial risk detection in Enterprise Resource Planning systems using ensemble machine learning. The design targets the methodological weaknesses that commonly affect applied studies in this area. These include data leakage from improperly ordered preprocessing, inflated metrics from random splitting, and shallow evaluation under class imbalance.

A composite benchmark, ERP-RiskBench, was assembled from public procurement event logs, labeled fraud data, and a new synthetic ERP component with controlled risk typology injection. Nested cross-validation with time-aware and group-aware splitting enforced leakage prevention. Resampling and feature selection were confined to training folds throughout.

The stacking ensemble of gradient boosting models achieved the strongest detection performance across all conditions tested. Ablation studies showed that splitting protocol is the single most influential factor in reported performance. Random splitting inflated MCC by up to 0.12 compared to deployment-realistic time-plus-group splits. Both SMOTE and CTGAN augmentation improved minority-class recall when used inside the leakage-safe pipeline. Probability calibration proved essential for turning model outputs into cost-sensitive operational decisions.

Feature stability analysis confirmed that procurement control features, especially three-way matching discrepancies, are the most consistently informative predictors across folds. The Explainable Boosting Machine achieved competitive performance while remaining fully transparent. For audit-critical deployments, it represents a viable alternative to opaque ensembles.

Future work should test these findings on proprietary ERP data from live enterprise environments. Extending the framework to streaming or near-real-time scoring is one natural next step. Incorporating graph-based features from vendor networks and exploring federated learning for multi-organization settings are also promising directions.

References

[1] T. H. Davenport, "Putting enterprise systems to work," *Harvard Business Review*, vol. 76, no. 4, pp. 121–131, 1998.

[2] Association of Certified Fraud Examiners, "Occupational fraud 2022: A report to the nations," *ACFE*, 2022.

[3] J. Pineau, P. Vincent-Lamarre, K. Sinha, V. Larivière, A. Beygelzimer, F. d'Alché Buc, E. Fox, and H. Larochelle, "Improving reproducibility in machine learning research," in *Journal of Machine Learning Research*, vol. 22, 2021, pp. 1–20.

[4] G. C. Cawley and N. L. Talbot, "On over-fitting in model selection and subsequent selection bias in performance evaluation," *Journal of Machine Learning Research*, vol. 11, pp. 2079–2107, 2010.

[5] S. Varma and R. Simon, "Bias in error estimation when using cross-validation for model selection," *BMC Bioinformatics*, vol. 7, no. 1, pp. 1–8, 2006.

[6] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2016, pp. 785–794.

[7] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "LightGBM: A highly efficient gradient boosting decision tree," in *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[8] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin, "CatBoost: Unbiased boosting with categorical features," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

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

[10] D. H. Wolpert, "Stacked generalization," *Neural Networks*, vol. 5, no. 2, pp. 241–259, 1992.

[11] S. Ö. Arik and T. Pfister, "TabNet: Attentive interpretable tabular learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 8, 2021, pp. 6679–6687.
[12] Y. Gorishniy, I. Rubachev, V. Khrulkov, and A. Babenko, "Revisiting deep learning models for tabular data," *Advances in Neural Information Processing Systems*, vol. 34, pp. 18 932–18 943, 2021.
[13] L. Hussein, R. B., B. S. Guttikonda, K. Valarmathi, and D. Vasavi, "Anomaly detection in industrial inspection visual systems using swin transformer with bayesian optimization," in *2025 4th International Conference on Distributed Computing and Electrical Circuits and Electronics (ICDCECE)*. Ballari, India: IEEE, 2025, pp. 1–6.
[14] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 9, pp. 1263–1284, 2009.
[15] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002.
[16] S. A. Pushkala, "Financial fraud identification using graph neural network and LSTM with autoencoder-based data refinement," *Journal of International Crisis and Risk Communication Research*, vol. 9, no. 1, 2026.
[17] D. Chicco and G. Jurman, "The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation," *BMC Genomics*, vol. 21, no. 1, pp. 1–13, 2020.
[18] T. Saito and M. Rehmsmeier, "The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets," *PLoS ONE*, vol. 10, no. 3, p. e0118432, 2015.
[19] A. Fernández, S. García, M. Galar, R. C. Prati, B. Krawczyk, and F. Herrera, "Learning from imbalanced data sets," *Springer*, 2018.
[20] C. Elkan, "The foundations of cost-sensitive learning," *Proceedings of the 17th International Joint Conference on Artificial Intelligence*, vol. 2, pp. 973–978, 2001.
[21] J. C. Platt, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," *Advances in Large Margin Classifiers*, vol. 10, no. 3, pp. 61–74, 1999.
[22] A. Niculescu-Mizil and R. Caruana, "Predicting good probabilities with supervised learning," in *Proceedings of the 22nd International Conference on Machine Learning*, 2005, pp. 625–632.
[23] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.
[24] S. García, A. Fernández, J. Luengo, and F. Herrera, "Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining," *Information Sciences*, vol. 180, no. 10, pp. 2044–2064, 2010.
[25] J. Roy and S. K. Singh, "Comparative evaluation of embedding representations for financial news sentiment analysis," arXiv preprint arXiv:2512.13749, 2025. [Online]. Available: https://arxiv.org/abs/2512.13749
[26] B. van Dongen, "BPI challenge 2019," 4TU.ResearchData, 2019.
[27] E. A. Lopez-Rojas, A. Elmir, and S. Axelsson, "PaySim: A financial mobile money simulator for fraud detection," in *Proceedings of the 28th European Modeling and Simulation Symposium*, 2016, pp. 249–255.
[28] L. Xu, M. Smeber, S. L. Hyland, B. Cebere, and K. Veeramachaneni, "Modeling tabular data using conditional GAN," in *Advances in Neural Information Processing Systems*, vol. 32, 2019.
[29] H. Nori, S. Jenkins, P. Koch, and R. Caruana, "InterpretML: A unified framework for machine learning interpretability," in *arXiv preprint arXiv:1909.09223*, 2019.
[30] Y. Lou, R. Caruana, J. Gehrke, and G. Hooker, "Accurate intelligible models with pairwise interactions," in *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2013, pp. 623–631.
[31] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Advances in Neural Information Processing Systems*, vol. 30, 2017.
[32] ULB Machine Learning Group, "Credit card fraud detection dataset," OpenML, id 42175, 2018.
[33] J. Romano, J. D. Kromrey, J. Coraggio, and J. Skowronek, "Adjusting for chance in nonparametric effect size estimation," *Proceedings of the Annual Meeting of the Florida Association of Institutional Research*, 2006.
[34] D. Sculley, G. Holt, D. Golovin, E. Davydov, T. Phillips, D. Ebner, V. Chaudhary, M. Young, J.-F. Crespo, and D. Dennison, "Hidden technical debt in machine learning systems," *Advances in Neural Information Processing Systems*, vol. 28, 2015.
[35] S. K. Singh and J. Roy, "Security without detection: Economic denial as a primitive for edge and IoT defense," *arXiv preprint arXiv:2512.23849*, 2025. [Online]. Available: https://arxiv.org/abs/2512.23849
[36] D. Ezhilarasan, N. P. G. Bhavani, B. S. Guttikonda, H. Mohameed, and H. D. Praveena, "Markov decision process based cost-benefit analysis of cybercrime and cyberdefense systems," in *2025 3rd International Conference on Data Science and Information System (ICDSIS)*. Hassan, India: IEEE, 2025, pp. 1–5.
[37] National Institute of Standards and Technology, "Artificial intelligence risk management framework (AI RMF 1.0)," NIST, Tech. Rep. AI 100-1, 2023.
[38] S. K. Mishra, "Natural language to sql at scale: Integrating openai with oracle autonomous database via select ai," Mar. 2026. [Online]. Available: http://dx.doi.org/10.36227/techrxiv.177281023.37874227/v1
[39] S. Mishra, *The SQL Universe: 85 Comprehensive SQL Questions For Data Engineering, Data Science, and AI: Your Definitive Guide to Ace Any Interview*, jan 2026.