THE METASCIENCE **Volume 6, Issue 4 (Oct-Dec)-2026**
(An International Multidisciplinary Research Journal)

Integrating Transaction Fraud Detection and SME Credit-Risk Prediction Through Drift-Aware Machine Learning

[1*]Andreea Tóth, [2] Radu Popescu
[1] Université Félix Houphouët-Boigny, CÔTE D'IVOIRE
[2] University of Zambia, ZAMBIA

Keywords

Transaction Fraud; SME Credit Risk; Concept Drift; Integrated Risk Architecture; Early Warning; Governance; Privacy; Temporal Validation

ABSTRACT

Transaction fraud detection and small and medium-sized enterprise (SME) loan-distress prediction are managed, in most institutions, as entirely separate operational functions, yet the two problems share a common methodological core: both involve rare adverse outcomes, delayed and slowly maturing labels, a predictor-outcome relationship that changes over time, and asymmetric costs attached to a false alarm versus a missed event. At the same time, the two problems differ in ways that make naively combining them into a single pooled model a genuine risk rather than a convenience: fraud is typically assessed at the level of an individual transaction with near-real-time latency requirements, while SME distress is assessed at the level of a borrower or facility over a materially longer horizon, and a transaction can be fraudulent without implying anything about the originating borrower's underlying solvency. This article proposes a task-aware integration architecture that shares data-quality monitoring, drift reporting, model-version tracking, and governance infrastructure across the two functions while deliberately retaining separate prediction services, separate outcome definitions, and separate evaluation criteria for fraud and for credit distress. The architecture draws on a recently proposed multi-institution validation design for drift-resilient early-warning models spanning both SME loan distress and transaction fraud, and on a leakage-safe, chronologically evaluated approach to fraud-specific drift detection under severe class imbalance, synthesizing both into a four-layer structure comprising task-specific prediction services, a shared monitoring layer, a risk-orchestration layer, and a governance and audit layer. The proposed evaluation methodology specifies chronological validation within each task, task-specific performance measures that are never averaged across fraud and credit risk, cross-institution testing, alert-routing rules, privacy and data-minimization controls, and an integration-level evaluation that treats the operational benefit of sharing infrastructure as an empirical question rather than an assumption. The article argues explicitly against pooling fraud and credit-distress labels into one undifferentiated score, on the grounds that doing so would discard exactly the distinctions that make each task's evaluation meaningful. We implement the architecture on two public datasets, the European credit-card fraud stream and 375,261 SBA small-business loans, with separate task services and one shared monitoring layer running identical code. A single universal alert threshold proved unworkable: a population stability index of 0.25 fired in 100 percent of no-shift comparisons for fraud and 0 percent for distress, and missed every cross-sector shift in the distress task, while task-calibrated thresholds behaved sensibly. The shared data-integrity screen raised no false alarms in either task and caught all four injected defects. Separate task models performed as well as a pooled joint model. Cross-task signals could not be tested, because no public dataset links a firm's transactions to its loans, and the architecture has not been validated in a live deployment.

Introduction

Transaction fraud and SME loan distress are, in most financial institutions, handled by different teams, governed by different policies, and modeled with entirely separate pipelines. Yet both problems share a

cluster of methodological difficulties that recur across financial risk prediction generally: adverse outcomes are rare relative to the overall population under observation, confirmatory labels arrive with meaningful delay rather than instantly, the relationship between observable predictors and the eventual outcome changes as behavior, policy, and economic conditions change, and the institutional cost of a false alarm is not interchangeable with the cost of a missed event. Fraud models typically classify individual transactions at high frequency and under tight latency constraints, flagging a specific payment for review within seconds; SME distress models typically estimate borrower- or facility-level risk over a considerably longer horizon, informing a credit review that may unfold over weeks. Integrating the infrastructure behind these two functions is an appealing prospect for an institution seeking to reduce duplication in its monitoring and governance tooling, but it also introduces a substantial methodological risk if integration is pursued carelessly: the prediction units genuinely differ, labels mature on different timescales, and the costs of the two kinds of error are not comparable on any natural common scale.

Rashid and Sattiraju[1] propose a multi-institution architecture for drift-resilient early-warning models that explicitly covers both SME loan distress and transaction fraud, piloted across a sample of U.S. community financial institutions, which establishes that the two problems are already being considered together at the level of validation methodology even where they remain organizationally and technically separate. Zuberi *et al.*[2] provide a detailed, fraud-specific demonstration of leakage-safe temporal evaluation under extreme class imbalance, offering a concrete chronological-splitting and validation-tuned thresholding procedure for exactly the kind of rare-event, non-stationary problem that both fraud and SME distress present. This article synthesizes these two directions into a single layered architecture that shares governance and drift-monitoring infrastructure while explicitly preserving task-specific modeling and evaluation. It does not assume, and argues directly against assuming, that transaction fraud and credit distress should ever be combined into one target variable or one undifferentiated risk score, however tempting that simplification might appear from an engineering standpoint.

Contribution

This article contributes a task-aware integration framework that separates shared infrastructure from task-specific prediction; an explicit architectural boundary specifying which components of a combined fraud-and-credit-risk system may legitimately be shared and which must remain task-specific; and a proposed evaluation design for assessing cross-task alert quality and the operational benefit of integration itself, rather than assuming that benefit follows automatically from combining more data sources. It also implements the shared monitoring layer and the separate prediction services on public fraud and SME-loan data, so that two of its three propositions are tested empirically. Its central organizing principle is that shared monitoring infrastructure can be integrated productively across tasks, but outcome definitions, decision thresholds, and model validation must remain specific to the particular risk being predicted, because collapsing that distinction risks producing a system that reports confidently on a quantity no one actually asked about.

Literature and Conceptual Distinction

Different Prediction Units and Horizons

Transaction fraud is typically assessed at the level of a single transaction, under near-real-time latency requirements that leave little room for an extended review process before a decision is required. SME distress, by contrast, is generally assessed at the level of a borrower, a loan, or a credit facility, over a horizon defined in months rather than seconds. A transaction can be fraudulent without implying anything about the solvency of the account or borrower it touches, and a financially distressed SME can continue to generate entirely legitimate transactions right up until, and often well past, the point at which distress becomes visible in its account record. These two tasks should therefore not be treated as interchangeable labels standing in for a single underlying construct called financial risk; their integration, where it is pursued at all, should occur

through a coordinated architecture that respects this distinction rather than through a single pooled classifier that implicitly denies it.

Common Methodological Challenges

Both tasks face genuine non-stationarity and meaningful label delay, though the specific mechanisms differ in detail. Fraud labels may require customer confirmation, a chargeback event, or a completed investigation before they can be treated as confirmed, a verification-latency problem that Dal Pozzolo and colleagues document directly from a practitioner's vantage point on deployed credit-card fraud systems, where the delay between a transaction and its confirmed label becomes part of the learning problem itself rather than an incidental detail to abstract away[3]. Distress labels, in turn, may depend on missed payments, a formal restructuring, or a defined default event, and can take months to mature into a form an institution is willing to treat as ground truth. The concept-drift literature more broadly establishes that the relationship between features and outcomes can change over time independently of any change in the features' own marginal distribution[4], which means a shared monitoring system serving both tasks must accommodate different label-maturity windows and different, task-specific definitions of what counts as material deterioration, rather than applying one maturity assumption uniformly across both.

A further methodological challenge specific to the integrated setting, rather than to either task in isolation, is the risk of inadvertent feature leakage across the shared infrastructure layer. Once a common feature-engineering or data-quality pipeline serves both the fraud and the distress service, a change made for one task's benefit, such as a new derived feature or an adjusted imputation rule, can propagate silently into the other task's feature set if the two services are not kept behind a genuinely enforced boundary. This differs from the classical temporal leakage problem that chronological splitting is designed to prevent, in that the leakage here runs across tasks rather than across time, and it requires its own safeguard: each service's feature-availability policy should be versioned and reviewed independently, so that a change intended to improve fraud detection cannot silently alter the SME distress model's inputs, or vice versa, without an explicit, documented decision to permit it.

Rare-Event Evaluation

Ordinary accuracy is close to useless when the adverse outcome being predicted is rare, since a classifier that never raises an alert can still achieve an impressively high accuracy figure by construction. Precision-recall analysis offers a considerably more informative view of minority-event detection under this kind of imbalance[5], and the Matthews correlation coefficient usefully incorporates all four confusion-matrix cells into a single balanced summary that is markedly less distorted by class skew than accuracy or even F1[6]. Fawcett's foundational treatment of receiver operating characteristic analysis cautions, in a closely related vein, that the apparent robustness of an ROC curve to class imbalance can be misleading, since a large population of negative cases can make a false-positive rate look reassuringly small in relative terms even when the implied absolute alert volume is operationally unmanageable[7], and Hand's critique of the area under that curve as a single summary statistic adds that the measure implicitly averages over a range of relative misclassification costs that may have no relationship to the costs either a fraud team or a credit team actually faces[8]. These metrics should be reported separately for each task rather than blended into a single combined figure, because a false positive in transaction fraud triggers a payment-authorization review with one operational cost structure, while a false positive in SME distress triggers a relationship-manager contact or an additional credit review with a materially different one; combining these costs into a single unexamined metric can obscure exactly the institutional priorities that a well-designed evaluation is supposed to surface. Where severe imbalance makes model training itself difficult, resampling techniques such as the Synthetic Minority Over-sampling Technique[9] are a standard, task-level response, generating synthetic minority-class examples by interpolation in feature space, though such a resampling step must be revisited whenever

the underlying rare-event distribution drifts, in either task, since a resampling strategy calibrated once at training time can otherwise become a hidden source of staleness.

Cross-Institution Evidence

The multi-institution architecture proposed by Rashid and Sattiraju[1] highlights a danger directly relevant here: the risk of mistaking a site-specific pattern for genuinely generalizable model behavior. The same issue applies, in a slightly different form, when a shared system integrates transaction-fraud and credit-risk signals across institutions, because data definitions, institutional policy, customer populations, and reporting practices can all differ from one participating institution to the next. A common monitoring layer serving both tasks should therefore standardize metadata, alert formats, and drift-reporting conventions without ever assuming that the underlying risk models themselves, or the populations they are scored against, are identical across sites.

Proposed Integrated Architecture

The architecture proposed here consists of four layers: task-specific prediction services, a shared data-quality and drift-monitoring layer, a risk-orchestration layer, and a governance and audit layer. The prediction services remain separate for transaction fraud and SME distress at every stage, from feature engineering through final decision threshold. The shared layer standardizes data lineage, timestamp handling, schema checks, drift reporting, and model-version metadata across both services, so that an institution gains the efficiency of common tooling without being forced to also share the substantive modeling decisions that tooling supports. The orchestration layer routes alerts to the relevant business process without merging incompatible target probabilities into a single combined score, preserving the principle, established above, that shared infrastructure and shared inference are not the same thing.

This four-layer separation is deliberately asymmetric in how much change it permits at each layer. The prediction-service layer is expected to change frequently and independently for each task, as either service's underlying model is recalibrated, retrained, or replaced in response to its own monitoring evidence. The shared monitoring, orchestration, and governance layers, by contrast, are expected to change far less often, precisely because they are the layer other parts of the organization come to rely on as a stable interface; a shared layer that itself drifts as readily as the prediction services it monitors would undermine the very consistency it exists to provide. This asymmetry should be reflected directly in how each layer is versioned and change-managed: prediction-service changes follow each task's own model-governance cycle, while shared-layer changes follow a slower, more deliberately reviewed change-management process of the kind described in the Governance and Traceability section below.

Task-Specific Prediction Services

The fraud service produces a transaction-level score together with an associated uncertainty estimate or reason code explaining the basis for the score. The SME service produces a borrower- or facility-level risk estimate over a specified prediction horizon. Each service should maintain its own target definition, its own feature-availability policy, its own calibration procedure, its own decision threshold, and its own evaluation metrics, rather than inheriting any of these from the other service by default. Where one service genuinely benefits from a signal produced by the other, for instance a fraud alert informing a credit review or vice versa, that cross-task signal must be available at prediction time in both services and must not leak information about an outcome that would not actually have been known at the moment the prediction was made.

The two services' decision thresholds should also be reviewed on independent cadences, reflecting the fact that the fraud service's operating environment can change within days, as a new attack pattern emerges, while the SME service's operating environment typically changes on a slower cycle tied to broader credit conditions. A shared review calendar that forces both thresholds to be revisited at the same interval risks

leaving the fraud threshold stale during a fast-moving attack campaign, or forcing unnecessary SME threshold churn in response to short-lived volatility that would not, on its own, justify a credit-policy change. Each service's threshold-review cadence should therefore be set according to that task's own rate of meaningful change, documented in the shared governance layer, rather than inherited from whichever task's cadence happens to be reviewed first.

Shared Monitoring Infrastructure

The shared layer monitors schema changes, missingness, feature distributions, score distributions, and operational latency across both services using a common technical vocabulary. It can apply consistent definitions for the population stability index, the Kolmogorov-Smirnov statistic, and Jensen-Shannon divergence wherever those measures are suitable for the data type in question, but it should not impose a single universal alert threshold across every feature and every task, since different event rates and different label-delay profiles require task-specific thresholds and task-specific response rules. An adaptive-windowing approach, in which the effective monitoring window is automatically reduced once a change-detection statistic crosses a calibrated threshold, offers a mechanism the shared layer can apply to either task without presupposing that the two tasks' drift dynamics behave identically, since the window-shrinking behavior is driven by each task's own detected change rather than by a schedule imposed from outside[10].

The shared layer's dashboards should also distinguish, for any given alert, whether the underlying signal is a change in the input distribution, a change in the model's own score distribution, or a change in confirmed outcome performance, because these three layers of evidence answer different questions for either task and a monitoring interface that collapses them into one undifferentiated alert stream makes diagnosis harder rather than easier. An input-distribution shift flagged for the fraud service and an input-distribution shift flagged for the SME service should be visually and operationally distinguishable at a glance, even where both are computed using the same underlying statistical test, so that a reviewer is never left guessing which task, or which layer of evidence, a given alert actually concerns.

Risk Orchestration

The orchestration layer determines how alerts from either service enter existing institutional workflows. A high-risk transaction alert may require an authorization-review step, whereas an elevated SME distress score may trigger a credit analyst's review, and these two workflows need not share a queue, a reviewer pool, or a service-level expectation. A cross-task alert, combining weaker evidence from both services, may be appropriate when separately validated evidence from each task suggests a broader relationship worth a human's attention, but the orchestration layer should never infer, and should be explicitly designed not to imply, that a fraud score is a credit-default probability or that a distress score is a fraud probability; the two remain distinct constructs even when a combined alert is operationally useful.

Designing the rule that decides when a cross-task alert is warranted is itself a nontrivial governance question rather than a purely technical one. A cross-task alert should require that both services' own task-specific evidence independently clears a documented threshold before a combined signal is raised, rather than allowing a strong signal from one service to compensate for a weak one from the other; otherwise the orchestration layer would effectively be constructing an informal pooled score through the back door, reintroducing exactly the risk the architecture's separation of prediction services was designed to avoid. The specific combination rule used, and the evidence required from each side before a cross-task alert fires, should be documented and auditable in the same way as either service's own decision threshold, so that a reviewer encountering a cross-task alert can trace precisely why it was raised.

Governance and Traceability

Every alert generated by either service should retain the originating model version, the score timestamp, the feature schema in force at scoring time, the decision threshold applied, and the subsequent confirmed outcome once it becomes available. Shared governance across the two services should define who owns each model, who holds approval authority for an intervention, and how an incident is escalated, mirroring the kind of documented approval record and defined alert ownership that a credible drift-governance process requires in the single-task case. Institutions should be able to trace any individual decision back to the specific model version and the specific data available at the time that decision was made, not merely to whatever model happens to be current when the trace is requested, since a governance structure that can only explain today's model cannot actually account for a decision made under yesterday's.

Governance responsibilities should also specify, in advance rather than reactively, how a disagreement between the two services' owners is resolved when a proposed change to the shared layer would benefit one task but complicate the other. Because the shared infrastructure layer by definition serves two separately accountable business functions, a change-approval process limited to a single owner is structurally insufficient; the governance and audit layer should require sign-off from both services' accountable owners before a shared-layer change is deployed, with an explicit, documented escalation path for cases where the two owners cannot agree, so that neither task's monitoring integrity is ever quietly subordinated to the other's convenience.

Proposed Evaluation Methodology

Temporal Design

Each task should use a chronological training-validation-test structure, written here as $D_{train,k} \prec D_{validation,k} \prec D_{test,k}$ for task k drawn from $\{fraud, distress\}$, where the ordering symbol denotes strict chronological precedence. Feature engineering and any learned preprocessing step should be fitted on training data only, consistent with the leakage-safe practice that Zuberi *et al.* demonstrate for fraud-specific drift evaluation[2]; the validation period should be used for model selection and threshold choice, while the test period remains genuinely untouched until final evaluation. Repeated forward windows, advancing the training and test boundary through the available history rather than fixing it at a single split, are preferred whenever each task's history is long enough to support them, since a single chronological split tests generalization across only one boundary rather than repeatedly across several.

Task-Specific Metrics

For transaction fraud, the evaluation should report precision, recall, the area under the precision-recall curve, the Matthews correlation coefficient, calibration quality, alert latency, and alert volume. For SME distress, it should report precision, recall, calibration, lead time ahead of the defined distress event, and performance disaggregated by borrower or facility segment. In both cases, confidence intervals and event prevalence should be disclosed alongside the point estimates, and decision thresholds should be set to reflect each task's own operational costs of a false alarm against a missed event rather than a threshold convention imported from the other task. Where risk scores are interpreted as probabilities or used to set an automatic threshold, a Dirichlet calibration procedure of the kind Kull and colleagues propose offers a principled way to recover well-calibrated estimates when the relationship between a model's raw output and the true outcome probability is more complex than a single scalar rescaling can capture, a realistic concern for either task once a model is evaluated outside the exact conditions it was originally fitted under[11].

Integration-Level Evaluation

Integration itself should be evaluated against operational outcomes rather than simply assumed to be beneficial on the grounds that more signals are now available to the combined system. Relevant measures include the duplicate-alert rate across the two services, analyst workload, time to resolution, the proportion

of alerts that result in a confirmed adverse outcome, and the frequency with which the two services produce contradictory signals about the same account or customer. If cross-task information is introduced into either model's feature set, an ablation study should test directly whether that information adds predictive value beyond the task-specific baseline, rather than assuming that adding a plausible-sounding cross-task feature necessarily helps.

An institution running this evaluation should also report what this article calls the integration overhead: the incremental analyst and engineering effort required to maintain the shared layer itself, set against whatever duplication it eliminates relative to two fully separate stacks. A positive result on the duplicate-alert rate or time-to-resolution measures above is only genuinely informative once this overhead is netted out, since a shared layer that reduces duplicate alerts but requires disproportionately more ongoing maintenance than two independent systems combined would represent, in the terms this article has used throughout, a case where integration was pursued because it was available rather than because it was shown to be worth its cost.

Cross-Institution Tests

Where multiple institutions participate in a shared evaluation, the evaluation should report site-specific metrics and explicit transfer experiments rather than a single pooled average, since a pooled figure can hide meaningful underperformance at an institution with a different customer base or a different internal data process. Local recalibration should be tested on a designated calibration subset, held separate from the sample used to fit the underlying model, and evaluated on a later, genuinely untouched period, following the same discipline recommended for within-task calibration above.

Risk, Fairness, Privacy, and Operational Controls

An integrated risk architecture can increase the consequences of a data error relative to two fully separate systems, because a shared feature-engineering step or a shared monitoring defect can propagate into both services at once rather than being contained within one. Data lineage, schema versioning, access control, and independent service-level checks are therefore not optional refinements but essential components of the architecture, precisely because a shared platform should never be allowed to become a single point of silent failure for two otherwise independent business functions.

Human Oversight

Human review should be proportionate to the consequence and uncertainty of the decision a given alert supports. Transaction alerts typically demand a rapid decision within a tightly bounded window, while SME distress alerts more often support a longer, relationship-based intervention that unfolds over an extended period. The interface presenting either kind of alert to a human reviewer should explain plainly what the underlying score does and does not mean, and what evidence actually supports it; a SHAP-based feature attribution can usefully show a reviewer which inputs are driving a particular score, but, consistent with the broader explainability literature's own caution, such an explanation should never be presented to a reviewer as causal proof of why the underlying event is occurring, since the explanation describes the model's behavior rather than the real-world mechanism generating the outcome[12].

Privacy and Data Minimization

Combining data across two previously separate risk functions can materially increase an institution's privacy exposure even when no single new data element is introduced, simply because a combined dataset makes re-identification and cross-task inference easier than either dataset would allow on its own. Institutions should define the minimum data genuinely required for each task, establish explicit permitted purposes for each data element, and restrict cross-task reuse to cases where that reuse has been separately justified rather than permitted by default because the data happens to be available in a shared store. Where multi-institution validation is involved, data-sharing arrangements should address security, access control, retention limits,

and auditability directly, and privacy-preserving computation techniques may help reduce exposure in some settings, but they do not remove the underlying need for governance of the kind described throughout this article.

Failure Containment

Because fraud and credit-risk models serve different workflows with different consequences, they should be independently deployable and independently reversible. A change made to one service should never silently alter the other service's decision thresholds or alert volume, and monitoring dashboards should clearly distinguish service-specific alerts from incidents in the shared infrastructure itself, so that an operator investigating a problem can tell quickly whether it is contained to one task or affects both. A validated fallback model and a documented rollback procedure for each service, maintained independently of the other, reduce the risk that a single infrastructure incident cascades into a broader disruption across both risk functions at once.

Empirical Implementation on Public Fraud and SME Loan Data

Services and Shared Monitoring Layer

The fraud service followed the leakage-safe protocol of Zuberi *et al.*[2] on the European credit-card dataset[13]: a LightGBM model[15] trained on the first 80 percent of day one, with 15 inputs selected by SHAP on training data only, scored eight three-hour windows of day two. The distress service used the SBA loan file[14]: a LightGBM model on origination-time features trained on retail-trade loans and applied to the seven other large industry sectors. These sector portfolios stand in for the participating institutions of Rashid and Sattiraju's design[1], since the public file identifies no lender; the full site-level evaluation is reported in a companion study. The two services shared nothing but a monitoring layer. That layer ran identical code for both tasks: a data-integrity screen (a field suddenly dominated by one value, or a continuous field whose median scale changed more than fivefold), maximum feature PSI, Jensen-Shannon divergence and a Bonferroni-corrected Kolmogorov-Smirnov test, all on windows of 2,000 records.

One Threshold or Two

To test the claim in the Shared Monitoring Infrastructure section that one universal threshold should not be imposed across tasks, each statistic was computed 30 times on pairs of samples where no shift was expected (adjacent periods of day one for fraud; two random halves of the reference sector for distress). This gave each task's false-alarm rate under the conventional thresholds and a task-calibrated threshold at the 95th percentile of its own null distribution. Table 1 reports the results.

Table 1: Shared Monitoring Layer Under a Universal Versus a Task-Calibrated Threshold

Task	False alarms, PSI >= 0.25 (no shift)	False alarms, KS p < 0.05 (no shift)	Task-calibrated PSI threshold	Windows alarmed, universal PSI	Windows alarmed, calibrated PSI	Integrity-screen false alarms
Fraud	100%	100%	3.47	8 of 8	8 of 8	0%
SME distress	0%	3%	0.016	0 of 7	7 of 7	0%

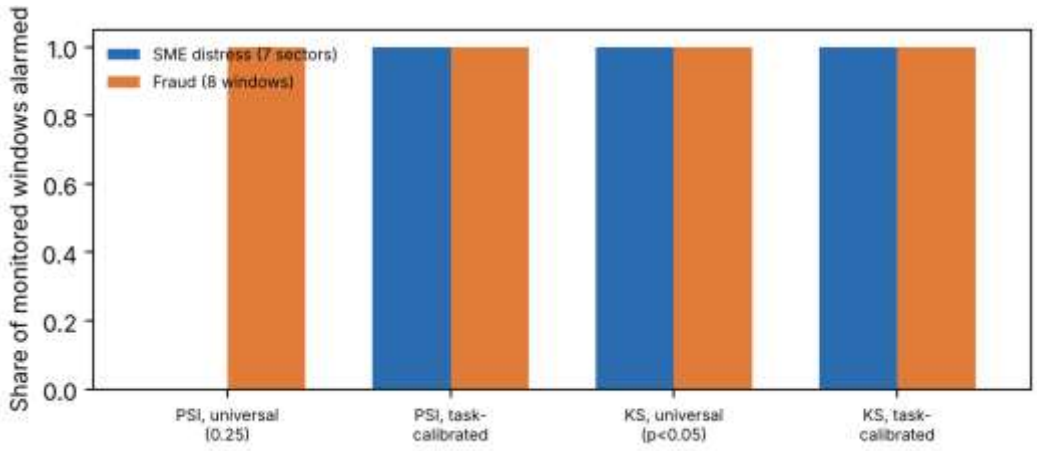


Figure 1. Share of monitored windows alarmed under universal and task-calibrated thresholds, by task [authors' analysis].

The same statistic and threshold meant entirely different things in the two tasks. For fraud, the hour-of-day composition of transactions differs between any two periods, so PSI exceeded 0.25 in every no-shift comparison and the task-calibrated threshold was 3.47; even at that level every day-two window alarmed, so for this task the statistic carries information only against a seasonally matched reference. For distress, PSI never reached 0.25 in a no-shift comparison, but it also stayed below 0.25 for all seven new sectors (maximum 0.19), although transferring the model to those sectors mis-stated their charge-off rates by up to 8.4 percentage points in the companion analysis. A threshold calibrated to the distress task's own null (0.016) flagged all seven. The KS test, with 2,000 records per window, flagged nearly everything in both tasks. A shared layer is therefore workable only if it shares the code and the reporting format but holds thresholds, and the choice of reference window, separately for each task.

Data-Quality Defects and the Shared Screen

Proposition P1 concerns data-quality problems. Two defect types were injected into one window of each task: a null-fill, writing the most important feature as zero for every record (V14 for fraud, the SBA-guaranteed share for distress), and a currency-unit error multiplying the amount field by 100. The shared integrity screen, applied unchanged to both tasks, flagged all four defects in the window where they occurred, naming the affected field each time, and it raised no false alarms in the 60 no-shift comparisons. Because the screen tests a defect's signature (a collapsed or rescaled field) rather than a distributional distance, it did not need the task-specific thresholds that the drift statistics required. This supports the part of P1 that public data can test, namely that one consistently applied screen detects the same defect type in both services without delay. Whether it detects defects faster than two independently maintained stacks would require observing such stacks in an institution.

Separate Versus Joint Models

For P2, the two separate services were compared with a single joint LightGBM model trained on both tasks' training data, using a task indicator and the union of both feature sets. Separation cost nothing. On the fraud test windows, PR-AUC was 0.784 for the separate model and 0.772 for the joint model; on held-out retail loans, AUC was 0.738 and 0.739, and Brier scores were 0.1557 and 0.1554. The joint model gained nothing from pooling, as expected when the two tasks share no features, while combining two outcome definitions into one model. P3, that cross-task signals add value only when available at decision time, could not be tested: no public dataset links an SME's card transactions to its loans.

Analytical Synthesis and Propositions

The architecture proposed in this article separates common infrastructure from task-specific inference as a deliberate design choice rather than an incidental one. This separation permits institutions to share practices for data quality, monitoring, documentation, and incident management across the fraud and credit-risk functions, while fully preserving the different meanings, horizons, and operational responses attached to transaction fraud on one hand and SME distress on the other. It also, and just as importantly, enables an institution to evaluate integration itself as a measurable intervention with its own costs and benefits, rather than treating it as a presumed improvement that follows automatically from having more data available in one place.

Table 2 summarizes which components of the proposed architecture are genuinely shared across the two tasks and which must remain task-specific, since the central design discipline of this framework is knowing which is which for every component rather than defaulting either toward full integration or full separation.

Table 2: Shared infrastructure and task-specific controls in an integrated risk architecture.

Component	Shared across tasks?	Task-specific requirements
Data lineage and schema checks	Yes	Feature availability and definitions
Drift statistics	Methods may be shared	Baselines and thresholds differ
Outcome label	No	Fraud confirmation vs. distress/default definition
Decision threshold	No	Transaction authorization vs. credit review
Performance reporting	Structure may be shared	Metrics and prediction horizon differ
Governance audit trail	Yes	Separate owners, approvals, and rollback paths

Testable Propositions

Three propositions follow from this architecture. The first, P1, is that shared monitoring infrastructure reduces the time required to detect a data-quality problem across risk services relative to two fully independent monitoring stacks, because a single, consistently applied set of schema and distributional checks should surface a given defect faster than two separately maintained checks would. The implementation supports the testable part of P1: the shared integrity screen caught all four injected defects, in both tasks, in the window they occurred, with no false alarms; the comparison with independent stacks needs institutional data. It also shows a limit: the shared distributional statistics were usable only with task-specific thresholds. The second, P2, is that keeping outcome models separate while sharing governance infrastructure improves traceability without degrading task-specific predictive performance. The implementation supports P2: separate services matched a joint model on both tasks. The third, P3, is that cross-task signals add predictive value to either service only when they are genuinely available at decision time and validated against that service's own task-specific baseline through an ablation study. P3 remains untested because no public data link the two tasks at the level of the firm.

Discussion

The central design principle running through this article is selective integration. Combining monitoring infrastructure across the fraud and SME-distress functions can improve operational consistency and reduce duplicated engineering effort, but combining the labels and scores themselves, without preserving what each

one actually means, risks producing invalid inferences that are difficult to detect precisely because the combined system still reports a plausible-looking number. Fraud detection and SME distress prediction differ in their unit of analysis, their event definition, their horizon, and the intervention each one triggers, and the architecture proposed here is built specifically to preserve these distinctions even where the two tasks share data-quality services or governance tooling underneath.

The framework also treats operational benefit as an empirical question rather than a premise. A shared platform may reduce duplication in monitoring and governance work, but it can simultaneously increase alert complexity for reviewers, raise privacy exposure through data combination, and introduce a new dependency on common infrastructure that neither task previously had to rely on. The evaluation methodology proposed above is designed specifically to compare the integrated architecture against separate baseline workflows on both technical and operational grounds, so that an institution adopting this architecture can determine, rather than assume, whether the integration was actually worth its cost.

Implications for Practice

For an institution considering this architecture, the most actionable starting point is not the shared monitoring layer itself but the explicit inventory of which components are genuinely shareable and which must remain task-specific, along the lines set out in Table 2. Institutions that already operate separate fraud and credit-risk functions typically have partially overlapping, inconsistently documented monitoring practices rather than no monitoring at all; the practical value of this framework lies in making the boundary between shared and task-specific components explicit enough that an institution can consolidate the former without being tempted to also consolidate the latter. A second practical implication is that the governance and audit layer should be established, and the dual sign-off requirement for shared-layer changes agreed between both services' owners, before any technical consolidation work begins, rather than retrofitted once two previously separate monitoring stacks have already been merged; experience with similar infrastructure consolidations elsewhere suggests that governance retrofitted after the fact is consistently harder to enforce than governance designed in from the start.

Future Research

Future empirical work should prioritize testing the three propositions advanced above on institutions that genuinely operate both a transaction-fraud and an SME-credit function, since a synthetic or single-task evaluation cannot speak to the cross-task claims this framework actually makes. A particularly informative study design would compare detection-time and response-time outcomes before and after a shared monitoring layer is introduced at an institution that previously ran two fully independent stacks, using the integration-level measures proposed in the Integration-Level Evaluation section rather than relying on either service's own task-specific metrics alone. Future work should also examine whether the dual-ownership governance model proposed here scales past two tasks, since an institution that eventually wishes to add a third risk function, such as anti-money-laundering monitoring, to the same shared infrastructure layer would need a governance structure that generalizes beyond the pairwise sign-off arrangement described above.

Conclusion

Transaction fraud and SME loan distress share a cluster of genuine methodological challenges, non-stationarity, rare outcomes, delayed labels, and asymmetric error costs, but they remain distinct prediction tasks rather than two views of a single underlying risk. A robust integrated architecture should share data-quality, monitoring, documentation, and governance infrastructure across the two functions while retaining task-specific labels, models, thresholds, and evaluation criteria for each, because collapsing that distinction in pursuit of engineering convenience risks producing a system that cannot be trusted to mean what it appears to say. This article proposes such an architecture, together with an evaluation protocol built on chronological validation, task-specific metrics, cross-institution tests, and operational outcome measures. Implemented on

public fraud and SME-loan data, the shared layer detected injected data defects in both tasks with one screen, but its drift statistics needed task-specific thresholds, and separate task models lost nothing against a joint model. Its full operational benefits remain to be established empirically rather than assumed from its design alone. Future work should test whether integration of this kind genuinely improves monitoring consistency and response time without a corresponding increase in false alerts, privacy exposure, or cross-task confusion, using the same leakage-safe, chronologically disciplined evaluation approach recommended throughout this article[2].

References

[1] Rashid, S. U., & Sattiraju, S. R. (2025). Multi-Institution Validation of Drift-Resilient Early-Warning Models for SME Loan Distress and Transaction Fraud: Evidence from US Community Financial Institution Pilots. *International Journal of Modern Computing*, *8*(1), 40-54.

[2] Zuberi, N., Rahman, M. M., Rashid, S. U., Ahsan, M. N., Rahman, M. D., & Mollah, M. H. O. R. (2026). A robust machine learning framework for detecting temporal drift in financial fraud prevention. *Scientific Reports*.

[3] Dal Pozzolo, A., Boracchi, G., Caelen, O., Alippi, C., & Bontempi, G. (2018). Credit card fraud detection: A realistic modeling and a novel learning strategy. IEEE Transactions on Neural Networks and Learning Systems, 29(8), 3784-3797.

[4] Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., & Bouchachia, A. (2014). A survey on concept drift adaptation. ACM Computing Surveys, 46(4), Article 44.

[5] Saito, T., & Rehmsmeier, M. (2015). The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets. PLOS ONE, 10(3), Article e0118432.

[6] Chicco, D., & Jurman, G. (2020). The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. BMC Genomics, 21, Article 6.

[7] Fawcett, T. (2006). An introduction to ROC analysis. Pattern Recognition Letters, 27(8), 861-874.

[8] Hand, D. J. (2009). Measuring classifier performance: A coherent alternative to the area under the ROC curve. Machine Learning, 77, 103-123.

[9] Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic minority over-sampling technique. Journal of Artificial Intelligence Research, 16, 321-357.

[10] Bifet, A., & Gavaldà, R. (2007). Learning from time-changing data with adaptive windowing. In Proceedings of the 2007 SIAM International Conference on Data Mining (pp. 443-448).

[11] Kull, M., Silva Filho, T. M., & Flach, P. (2017). Beyond temperature scaling: Obtaining well-calibrated multiclass probabilities with Dirichlet calibration. In Advances in Neural Information Processing Systems (Vol. 30). Curran Associates, Inc.

[12] Lundberg, S. M., & Lee, S.-I. (2017). A unified approach to interpreting model predictions. In Advances in Neural Information Processing Systems (Vol. 30, pp. 4765-4774). Curran Associates, Inc.

[13] Dal Pozzolo, A., Caelen, O., Johnson, R. A., & Bontempi, G. (2015). Calibrating probability with undersampling for unbalanced classification. In *2015 IEEE Symposium Series on Computational Intelligence* (pp. 159-166). IEEE.

[14] Li, M., Mickel, A., & Taylor, S. (2018). "Should this loan be approved or denied?": A large dataset with class assignment guidelines. *Journal of Statistics Education*, 26(1), 55-66.

[15] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T.-Y. (2017). LightGBM: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems 30* (pp. 3146-3154).