

Adopting Drift-Aware Machine Learning for Real-Time Fraud Detection in Financial Institutions: A Simulation-Based Pre-Deployment Evaluation

Zofia Otieno ^{1*}, Lena Bernard ²

¹ Université Paris-Saclay, FRANCE

² Technical University of Munich, GERMANY

Abstract

Background. Financial fraud models degrade as fraud tactics, customer behaviour and fraud prevalence change, and confirmed labels arrive only after investigation. Financial institutions considering drift-aware models must decide before deployment whether such a policy will maintain detection, how it will affect alert workload, and how often it will change the model under governance — but pre-deployment evidence over realistic time horizons is limited. **Objective.** To provide a pre-deployment evaluation, for financial institutions, of a drift-aware monitoring layer implementing the temporal-drift framework of Zuberi *et al.* [1], and to derive a governed adoption roadmap. **Methods.** A 52-week stream of 520,000 transactions with sudden (card testing), gradual (behavioural) and recurring (account takeover and seasonal) drift and a two-week label delay was simulated to represent conditions a financial institution faces. A gradient-boosting scoring model was maintained by five strategies: static, periodic (four-weekly), sliding-window (weekly), performance-triggered and drift-aware monitoring. The drift-aware layer applied PSI, Kolmogorov–Smirnov and Jensen–Shannon tests to SHAP-selected features, a prevalence-shift z-test and F2 threshold tuning, plus a diagnosis step separating real from virtual drift. Strategies were evaluated prequentially over 5 replications. **Results.** The static model fell to an average precision of 0.289. Drift-aware monitoring reached 0.643 ± 0.012 and a recall of 0.653 at a 1% alert budget, 98.9% of the weekly sliding window (0.650), while retraining 4.6 instead of 47 times (90% fewer). It outperformed performance-triggered (0.628) and periodic (0.611) retraining. The prevalence test triggered timely retraining in four of five replications, and the diagnosis step suppressed retraining on harmless seasonal drift. Distributional tests, however, did not detect a card-testing campaign confined to the fraud minority. Scoring latency remained at most 11 μ s per transaction. **Conclusions.** For financial institutions, the results support adopting drift-aware retraining in which supervised and prevalence monitoring decide when to retrain and unsupervised drift tests are used for diagnosis. Because the evidence is simulation-based, adoption should proceed through an offline backtest on the institution’s own data, shadow mode and a pilot portfolio before production; a roadmap with readiness requirements, rollout phases and key risk indicators is provided.

Keywords: Financial Institutions; Concept Drift; Fraud Detection; Model Risk Management; Adaptive Monitoring; Population Stability Index; Real-Time Scoring; Model Retraining; SHAP; Label Delay

Introduction

Payment fraud is an adversarial and non-stationary problem. Fraudsters change tactics when controls catch them, customers change how and when they pay, and the share of fraudulent

transactions moves with seasons, product launches and criminal campaigns [2], [3]. A fraud model that is trained once and left in production therefore loses accuracy over time — a phenomenon known as concept drift [4]–[6]. In card and account fraud the problem is sharpened by two operational facts: fraud is rare, typically well under 1% of transactions, and the true label of a transaction is confirmed only after investigation or a customer dispute, often days or weeks later [7].

Institutions respond to this in three broad ways. Many retrain on a fixed calendar; some retrain continuously on a sliding window of recent data; and a growing number monitor the model and retrain only when monitoring indicates that something has changed [8]–[10]. The third option — drift-aware adaptation — is attractive because it promises the accuracy of frequent retraining without its computational, validation and governance cost: every retrained fraud model must be tested, documented and approved before deployment under model-risk guidance such as SR 11-7 [11]. Yet adoption evidence is thin. Most drift-detection studies evaluate detectors on generic benchmark streams, and most fraud studies evaluate a single model on a single chronological split.

A recent framework by Zuberi *et al.* [1] provides a practical basis for closing this gap. Working on the European credit-card benchmark, it combined strict chronological splitting and leakage control with a battery of temporal-drift measurements — the Population Stability Index (PSI), the Kolmogorov–Smirnov (KS) test and the Jensen–Shannon divergence (JSD) computed on SHAP-selected features, together with prevalence-shift z-tests — and F2-oriented threshold tuning. Because that benchmark spans only 48 hours, the authors explicitly treated their drift results as a methodological lower bound and noted that long-horizon production drift remained to be studied [1].

For a financial institution, adopting a drift-aware model is a decision taken before any production evidence exists. Model-risk management requires that a new monitoring and retraining policy be justified, tested and approved before it can influence live fraud decisions [11], and fraud operations need to know in advance how the policy will affect alert volumes, analyst workload and the frequency of model changes. Institutions therefore need pre-deployment evidence: controlled evaluations that show how a candidate policy behaves under the kinds of drift the institution expects, and a clear path from that evaluation to a governed rollout.

This article provides such a pre-deployment evaluation. We implement the drift-measurement framework of Zuberi *et al.* [1] as the monitoring layer of a real-time fraud-scoring system and evaluate it in a controlled simulation of a 52-week transaction stream containing sudden, gradual and recurring drift and a two-week label delay. The study is not a report of deployment at a particular institution; it is designed to inform the adoption decision of institutions considering drift-aware models, and to show how such a model can be introduced under model-risk governance. Its contributions are:

1. an implementation of the temporal-drift framework of Zuberi *et al.* [1] as a live monitoring and retraining-decision layer, including a drift-diagnosis step that separates real drift (which harms detection) from virtual drift (which does not);
2. a controlled comparison of five adaptation strategies — static, periodic, sliding-window, performance-triggered and drift-aware — over 5 independent replications of a 520,000-transaction stream;

3. evaluation at an operational alert budget (the top 1% of transactions sent to analysts), alongside average precision, F2, false alerts per 1,000 transactions, retraining frequency, training cost and per-transaction scoring latency;
4. an analysis of what the unsupervised drift battery can and cannot see when fraud is a small minority of traffic, and of how model explanations shift as fraud patterns change;
5. an adoption roadmap for financial institutions — readiness requirements, a phased rollout from offline backtesting to production, the roles of the three lines of defence and the key risk indicators to report — derived from these results.

Related Work

Concept drift in fraud detection

Concept drift is a change in the joint distribution of inputs and labels over time [5], [12]. Following Widmer and Kubat [4], it is useful to distinguish *real* drift, in which the relationship between features and fraud changes, from *virtual* drift, in which only the input distribution changes. The distinction matters in fraud: a holiday surge in purchase amounts is virtual drift that inflates feature-level alarms without necessarily harming a model, whereas a new card-testing campaign is real drift that may leave aggregate feature distributions almost untouched while making the model blind to a new attack. Drift may also be sudden, gradual, incremental or recurring [5], [6].

Fraud-specific studies have documented each of these effects. Dal Pozzolo *et al.* [3], [7] showed on several years of real card data that models must be updated to remain accurate, that verification latency means only a small, biased subset of labels is available quickly, and that combining models trained on delayed and on immediate feedback helps. Carcillo *et al.* [10] built a streaming fraud architecture on Spark, while Whitrow *et al.* [13] and Jurgovsky *et al.* [14] showed that temporal aggregation and sequence features carry much of the fraud signal — features whose distributions are themselves sensitive to drift.

Drift detection and adaptation

Supervised detectors monitor the error of a deployed model: the Drift Detection Method (DDM) signals when the error rate rises significantly above its historical minimum [8], ADWIN compares adaptive windows of a performance statistic [9], and CUSUM-type tests such as Page–Hinkley accumulate evidence of a shift [15]. These detectors need labels, which in fraud arrive late. Unsupervised detectors compare input distributions directly and therefore react without labels; PSI, KS and JSD are the statistics most often used in practice [16]–[18]. Adaptation strategies range from full retraining on recent data to ensembles that reweight members as performance changes [19]. Sliding-window retraining is a strong but expensive baseline, because it retrains whether or not anything has changed.

The temporal-drift framework adopted in this study

Zuberi *et al.* [1] proposed a robust machine-learning framework for detecting temporal drift in financial fraud prevention. Its central methodological elements are: (i) chronological partitioning, with every preprocessing and selection step fitted on the training partition only; (ii) SHAP-based feature selection [20] to focus monitoring on the features the model actually uses; (iii) a drift battery of PSI, KS and JSD on those features, complemented by z-tests for a shift in fraud prevalence; and

(iv) decision thresholds tuned to maximise F2, which weights recall above precision as fraud operations require. Across 85 model–sampler configurations the framework reached an MCC of 0.806 and an AUC of 0.970 with a soft-voting ensemble, while measuring only modest drift within the 48-hour benchmark [1]. This study implements elements (i)–(iv) unchanged in a continuously operating system and adds the decision logic needed to turn drift measurements into retraining actions. Table 1 positions the strategies compared here.

Table 1. Adaptation strategies compared in this study

Strategy	When the model is retrained	Training data	Monitoring used
Static	Never (trained on weeks 0–3)	Weeks 0–3	None
Periodic	Every 4 weeks	Last 8 labelled weeks	None
Sliding window	Every week	Last 4 labelled weeks	None
Performance-triggered	When weekly average precision falls below the reference mean by max(0.06, 3 SD)	Last 4 labelled weeks	Supervised (DDM-style) [8]
Drift-aware monitoring	As performance-triggered, plus prevalence z-test $ z > 3$, plus unsupervised alarm confirmed by a smaller performance drop	Last 4 labelled weeks	Framework of Zuberi <i>et al.</i> [1] + diagnosis

Materials and Methods

Simulated transaction stream

Real production data with documented, long-horizon drift are not publicly available, and the public card benchmark spans only two days [1], [7]. We therefore generated a synthetic stream in the tradition of agent- and distribution-based fraud simulators such as PaySim [21], which allows the type, timing and size of each drift to be controlled and known exactly. Each of 52 weeks contains 10,000 transactions described by ten features: log amount, hour of day, a card-not-present flag, log distance from the customer’s usual location, 24-hour transaction velocity, log time since the previous transaction, a new-device flag, log account age, the ratio of the amount to the customer’s average, and a merchant-category risk score. The fraud rate is 0.6% before week 18 and 0.9% afterwards, giving about 3,900 fraud cases per replication. The drift schedule is summarised in Table 2.

Table 2. Drift schedule of the simulated 52-week transaction stream

Weeks	Event	Drift type	What changes
0–17	Baseline; pattern C (account takeover) in weeks 6–10	Recurring (real)	Fraud is high-value card-not-present (pattern A); C adds new-device, long-distance fraud

Weeks	Event	Drift type	What changes
18	Card-testing campaign (pattern B) begins	Sudden (real)	Low amounts, high velocity, short gaps; 60% of fraud; prevalence rises from 0.6% to 0.9%
26–36	Behavioural shift	Gradual (virtual and real)	Legitimate card-not-present share 35%→60%, night activity 8%→22%; pattern A amounts fall
44–49	Holiday season and return of pattern C	Recurring (virtual and real)	Legitimate amounts and velocity rise; account-takeover fraud returns

Scoring model and real-time constraints

Every strategy uses the same scoring model, a histogram-based gradient-boosting classifier (200 iterations, learning rate 0.08, 31 leaves) of the LightGBM family [22], implemented in scikit-learn [23]. For training, legitimate transactions are undersampled to 20% and reweighted by a factor of five, a standard response to extreme class imbalance [7]. Scoring latency was measured on every transaction to confirm that adaptation does not compromise real-time use.

Implementation of the adopted drift framework

Figure 1 shows the monitoring architecture. Following Zuberi *et al.* [1], all fitting steps use only data available at the time: each retraining uses the four most recent weeks for which labels have arrived, and labels arrive two weeks after a transaction. After each retraining the six features with the largest mean absolute SHAP value [20], [24] on a 3,000-transaction training sample are selected for monitoring, the drift references are reset to the two most recent labelled weeks, and the alert threshold is re-tuned to maximise F2 [1].

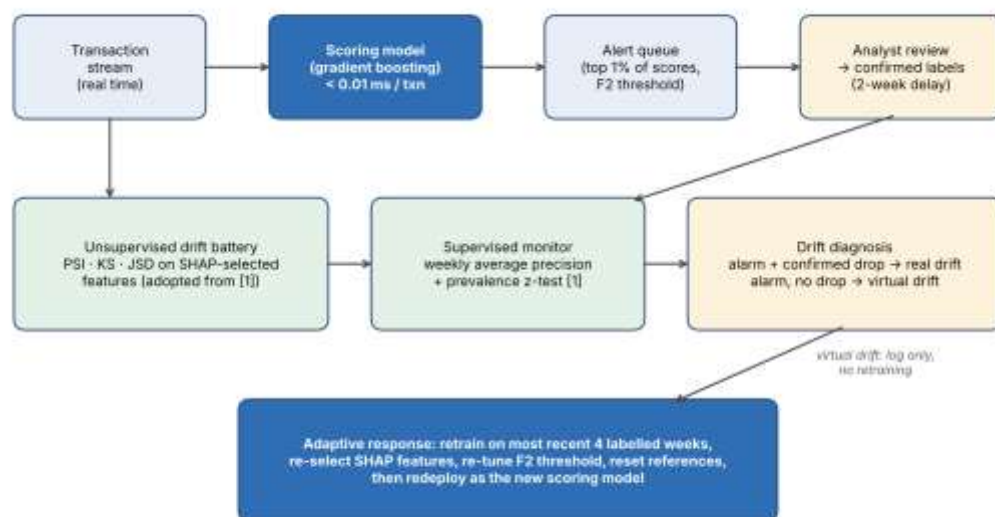


Figure 1. Architecture of the drift-aware monitoring system. The unsupervised battery and the prevalence test implement the temporal-drift framework of Zuberi et al. [1]; the diagnosis step decides whether a detected change warrants retraining.

Each week, the unsupervised battery compares the newest unlabelled week with the reference on each SHAP-selected feature using PSI, the two-sample KS test and JSD [1], [16]–[18]. A feature is flagged when $\text{PSI} \geq 0.10$ with a KS p-value below 0.001, or when $\text{PSI} \geq 0.25$; an alarm is raised when at least two features are flagged. For discrete features PSI is computed over observed values rather than quantile bins. When labels for a week arrive, the supervised monitor computes that week’s average precision and a two-proportion z-test of fraud prevalence against the training reference [1].

The diagnosis step then applies three rules. A large performance drop — below the reference mean by $\max(0.06, 3 \text{ SD})$ — triggers retraining, as in the performance-triggered strategy. A prevalence shift with $|z| > 3$ triggers retraining even without a performance drop, because the class balance the model was trained on has changed. An unsupervised alarm is checked against the labels of the alarmed week when they arrive: a smaller drop of $\max(0.03, 1.5 \text{ SD})$ confirms *real* drift and triggers retraining; otherwise the alarm is recorded as *virtual* drift and no retraining occurs. Retraining is suppressed for two weeks after a previous retraining.

Evaluation protocol

Strategies were evaluated prequentially: each week from week 4 to week 51 is scored by the model in force at that time, before any of its labels are used. The primary metrics are average precision (AP), which is more informative than ROC AUC under heavy imbalance [25], and recall at a 1% alert budget — the share of fraud caught when analysts review the top 100 transactions of each 10,000. We also report precision at that budget, false alerts per 1,000 transactions, F2 at the tuned threshold [1], the number of retraining events, total training time and mean scoring latency. Every stream and strategy was run with 5 independent random seeds; results are reported as mean $\pm$ standard deviation across seeds.

Results

Overall detection performance

Table 3. Detection performance and adaptation cost over weeks 4–51 (mean $\pm$ SD over 5 seeds)

Strategy	Average precision	Recall @ 1%	Prec. @ 1%	False alerts per 1,000	F2	Re-trains	Train time (s)
Static	0.289 $\pm$ 0.018	0.355 $\pm$ 0.012	0.261	7.39	0.262	0.0	1.1
Periodic	0.611 $\pm$ 0.015	0.638 $\pm$ 0.011	0.515	4.85	0.588	11.0	10.2
Sliding window	0.650 $\pm$ 0.005	0.658 $\pm$ 0.006	0.532	4.68	0.530	47.0	46.2
Performance-triggered	0.628 $\pm$ 0.021	0.638 $\pm$ 0.013	0.514	4.86	0.500	4.0	4.9
Drift-aware monitoring	0.643 $\pm$ 0.012	0.653 $\pm$ 0.009	0.527	4.73	0.513	4.6	5.4

Note. Recall, precision and false alerts are computed at a fixed budget of 1% of transactions per week. Mean scoring latency was between 3 and 11 μ s per transaction for every strategy.

Table 3 summarises the results. Without adaptation, the static model collapsed: its average precision fell to 0.289 and it caught only 35.5% of fraud at the 1% budget, against 65.5% before the first major drift. Every adaptive strategy recovered most of that loss. The sliding window, which retrains every week, reached the highest average precision (0.650) and recall (0.658). Drift-aware monitoring came within 0.007 of it in average precision (0.643, 98.9% of the sliding-window value) and within 0.005 in recall, while retraining 4.6 times instead of 47 — 90% fewer retraining events and 88% less training time.

Drift-aware monitoring also improved on the purely performance-triggered strategy (average precision 0.643 vs 0.628; recall 0.653 vs 0.638) and on periodic retraining (average precision 0.611), with less run-to-run variability than the performance trigger. Periodic retraining achieved the highest F2 at its tuned threshold (0.588), reflecting its larger eight-week training window, which gives the F2 threshold more fraud examples to tune on. Scoring latency stayed between 3 and 11 μ s per transaction for all strategies, so drift awareness carries no real-time penalty. Figure 2 shows the weekly trajectories.

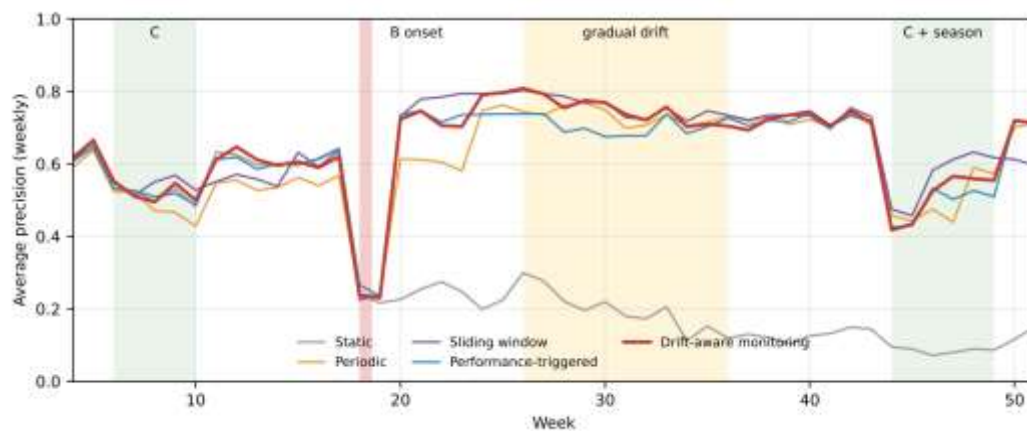


Figure 2. Weekly average precision of the five strategies (mean of 5 seeds). Shading marks the account-takeover pattern C (green), the onset of the card-testing pattern B (red), and the gradual behavioural drift (yellow).

Performance by drift phase

The phase-level view (Figure 3) shows where the strategies differ. After the sudden card-testing campaign (weeks 18–25), recall fell to 0.283 for the static model, while the sliding window (0.629), drift-aware monitoring (0.613) and the performance trigger (0.606) all recovered; periodic retraining lagged (0.572) because its schedule could not respond to the attack and its eight-week training window diluted the new pattern with older data. Every adaptive strategy dipped in weeks 18–19 (Figure 2): with a two-week label delay, no supervised method can respond before the first labels of a new pattern arrive.

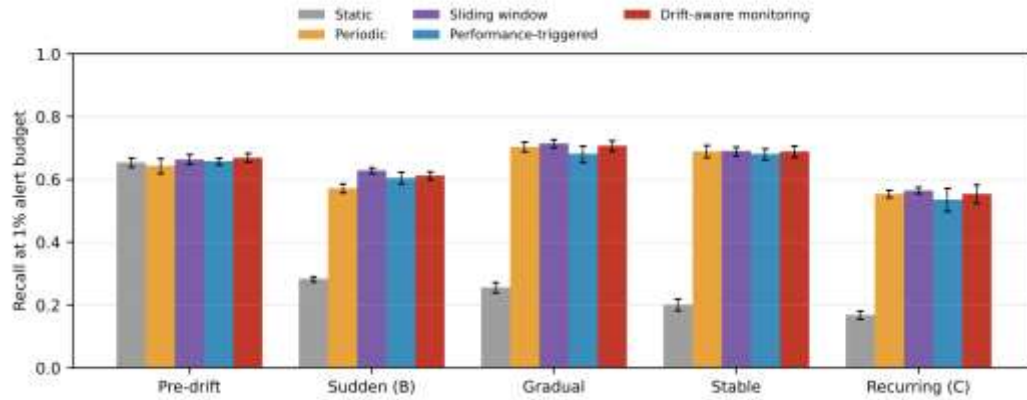


Figure 3. Recall at the 1% alert budget by drift phase. Error bars are one standard deviation across seeds.

During the gradual behavioural drift (weeks 26–36), drift-aware monitoring reached a recall of 0.707, essentially equal to the sliding window (0.715) and higher than the performance trigger (0.681). The difference came from the prevalence test adopted from Zuberi *et al.* [1]: in 4 of 5 replications the z-test detected the rise in fraud prevalence (in week 22 in 3 replications) and triggered a retraining that the performance rule alone did not request, so the drift-aware model entered the gradual phase already adapted to the new fraud mix. In the recurring phase (weeks 44–51) recall was 0.554 for drift-aware monitoring and 0.566 for the sliding window. Table 4 lists every retraining decision.

Table 4. Retraining and monitoring events by replication (week of the labelled data that triggered the event)

Seed	Performance-triggered retraining	Drift-aware retraining	Prevalence z-test	Unsupervised alarms	Diagnosed virtual
1	10, 18, 22, 32, 44	10, 18, 22, 32, 44	—	44, 45	—
2	18, 34, 44	18, 22, 34, 44	22	44, 45	—
3	18, 31, 37, 43	18, 27, 31, 43	27	44, 45, 46, 47, 48, 49	47, 48, 49
4	18, 29, 44	18, 22, 32, 44	22	44, 45	—
5	9, 18, 30, 36, 41	7, 18, 22, 36, 41, 45	22	44, 45, 46	—

Note. A retraining triggered by week w uses labels that arrive in week $w + 2$, so the new model scores transactions from week $w + 2$ onwards.

What the drift battery saw

Figure 4 traces the unsupervised battery for one replication. Three findings stand out. First, the battery was blind to the card-testing campaign: because fraud made up less than 1% of traffic, the arrival of pattern B left the distributions of the monitored features unchanged (maximum PSI 0.003 in weeks 18–25 in every replication). This campaign was detected entirely by the supervised monitor, at the earliest point allowed by the label delay (week-18 labels, in all replications). Second, the gradual behavioural drift was visible but moderate: the largest single-feature PSI, always on the

card-not-present flag, reached 0.08–0.26 by weeks 32–36, but only one feature crossed the flagging rule at a time, so no alarm was raised before a retraining reset the reference.

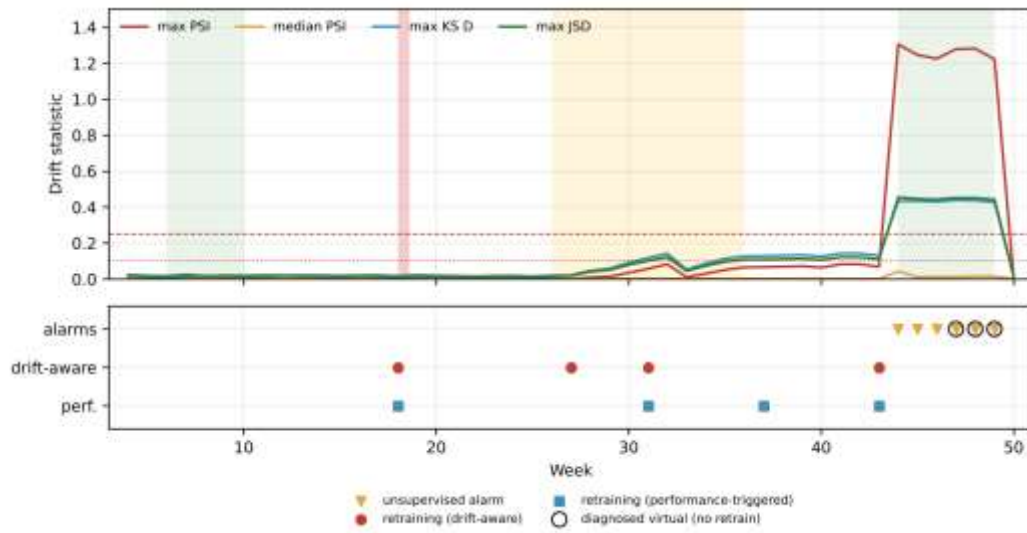


Figure 4. Unsupervised drift battery (PSI, KS and JSD on SHAP-selected features, after Zuberi et al. [1]) and resulting decisions for replication 3. Dashed and dotted lines mark PSI thresholds of 0.25 and 0.10.

Third, the holiday season produced the strongest signal in the stream: the maximum PSI exceeded 1.2, driven by 24-hour velocity and amount, and alarms were raised in every replication. Here the diagnosis step earned its place. In replication 3, three successive alarms (weeks 47–49) were followed by labelled weeks with no meaningful loss of average precision; they were classified as virtual drift and the model was not retrained. In replication 5 the season coincided with a confirmed drop and was classified as real drift. In this way the diagnosis step prevented retraining in response to changes in legitimate spending that did not affect detection — precisely the distinction between real and virtual drift that a raw PSI threshold cannot make [4], [12].

Shifts in model explanations

SHAP explanations [20], [24] showed how the model's reasoning changed as fraud evolved (Figure 5). Before the card-testing campaign, card-not-present status (0.60), log amount (0.52) and merchant risk (0.46) dominated. After pattern B appeared, 24-hour velocity rose from 0.10 to 0.51 and time since the previous transaction from 0.12 to 0.38, while the importance of amount fell. After the gradual drift made card-not-present purchases common among legitimate customers, that flag's importance fell from 0.81 to 0.49. Re-selecting the monitored features after each retraining, as the adopted framework does through SHAP [1], keeps the drift battery focused on the features the current model actually uses.

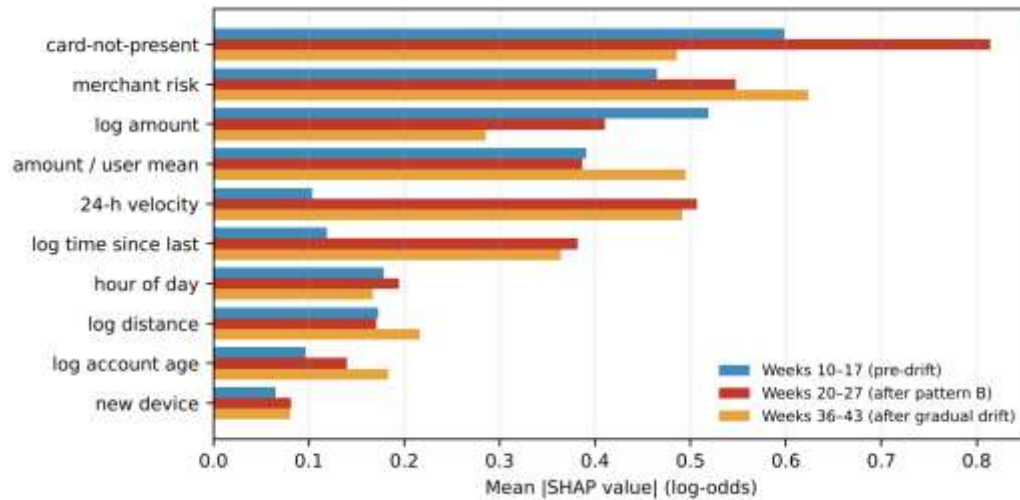


Figure 5. Mean absolute SHAP value of each feature for models trained before the card-testing campaign, after it, and after the gradual behavioural drift (replication 1).

Cost of adaptation

Figure 6 places each strategy on the trade-off between detection and retraining effort. The sliding window buys its small accuracy advantage with 47 retraining events, each of which in a regulated institution would require validation and sign-off [11]. Drift-aware monitoring and the performance trigger occupy the efficient corner — high recall with roughly four to five retraining events a year — and drift-aware monitoring sits higher in that corner.

Expressed in operational terms for an institution processing one million transactions a week with a 1% alert budget, drift-aware monitoring would generate about 4,731 false alerts a week against 7,388 for an unmaintained model, while catching 65% rather than 35% of fraud at the same analyst capacity — roughly 298 additional fraud cases intercepted for every 1,000 attempted. Over the 48 evaluated weeks it would require about 5 model changes for validation and approval, compared with 47 for weekly retraining and 11 for four-weekly retraining — some 42 fewer validation cycles a year than weekly retraining.

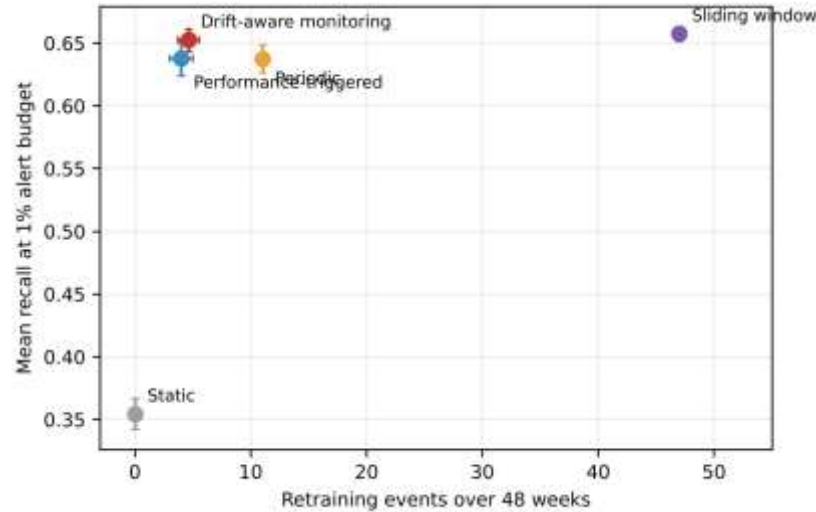


Figure 6. Trade-off between recall at the 1% alert budget and the number of retraining events (mean $\pm$ SD across seeds).

Discussion

Drift awareness matters more than retraining frequency

The central finding is that a fraud model does not need to be retrained often to stay accurate; it needs to be retrained at the right time. Drift-aware monitoring delivered 99% of the sliding window's average precision with about one tenth of its retraining events, and it outperformed a fixed four-weekly schedule that retrained more than twice as often. For an institution, each avoided retraining is a validation cycle, a deployment and a governance record avoided [11]. These results extend the conclusion of Zuberi *et al.* [1] — that drift must be measured under strict temporal discipline — from a two-day benchmark to a year-long operating horizon, and show that their measurement battery can be turned into a retraining policy.

Unsupervised monitoring has a structural blind spot

The second finding is a caution. Feature-distribution tests watch the whole population, and in fraud the whole population is almost entirely legitimate. A new attack that affects a fraction of 1% of transactions can therefore pass through PSI, KS and JSD unnoticed, as the card-testing campaign did here, while large but harmless shifts in legitimate behaviour trigger strong alarms. Unsupervised monitoring is therefore best treated as one input to a diagnosis rather than as a retraining trigger in its own right. The prevalence z-test, which is computed on labels, was the component of the adopted framework that added most to detection, and the confirmation step was the component that prevented unnecessary retraining. Zuberi *et al.* [1] anticipated this by pairing distributional tests with prevalence-shift tests; the present results suggest that the pairing is essential rather than optional.

These findings shape how a financial institution should introduce drift-aware fraud models; the next section sets out a roadmap.

Adoption Roadmap for Financial Institutions

A simulation can establish how a monitoring policy behaves under known drift; it cannot establish how it will behave on a particular institution's portfolio. The roadmap below therefore treats the results of this study as the first stage of a governed adoption process, in which each subsequent stage generates the institution's own evidence before the policy is allowed to change live fraud decisions. It follows the expectations of model-risk guidance for development, validation and ongoing monitoring [11].

Readiness requirements

Table 5. Readiness requirements for adopting drift-aware fraud models

Area	Requirement	Why it matters (evidence in this study)
Label pipeline	Confirmed fraud and non-fraud outcomes linked back to transactions, with a known and measured delay	Label delay sets the earliest possible detection; every adaptive strategy dipped for two weeks after a new attack
Prevalence data	Weekly fraud counts and transaction volumes per portfolio	The prevalence z-test [1] triggered timely retraining in four of five replications
Alert capacity	A defined analyst alert budget (e.g. top 1% of transactions)	All results were evaluated at a fixed budget; recall depends on it
Feature monitoring	Stable feature definitions and a store of reference distributions	PSI, KS and JSD require consistent features; SHAP-based selection keeps monitoring focused
Retraining pipeline	Reproducible training on the most recent labelled weeks, with automated validation tests	Drift-aware retraining needed about four to five model changes a year
Governance	Model inventory entry, documented thresholds, named owners and approval route	Every retraining is a model change subject to validation and approval [11]

Phased rollout

Table 6. Phased rollout of drift-aware monitoring in a financial institution

Phase	Activity	Exit criterion
1. Offline backtest	Replay 12–24 months of the institution's own labelled transactions through the five strategies compared in this study, with the same prequential protocol	Drift-aware policy matches or exceeds current practice at the institution's alert budget
2. Shadow mode	Run the drift battery, prevalence test and retraining logic alongside the production model; log alarms and candidate models without acting on them	False-alarm rate on stable periods and agreement with analyst judgement are acceptable to the first line

Phase	Activity	Exit criterion
3. Pilot portfolio	Allow drift-aware retraining to update the model for one product or region, with champion–challenger comparison against the incumbent	Challenger performs at least as well over a full seasonal cycle; independent validation sign-off
4. Production	Extend to all portfolios; automate retraining with a validation gate; report key risk indicators monthly	Ongoing: indicators remain within agreed limits
5. Periodic review	Review thresholds, alarms, virtual-drift diagnoses and retraining outcomes at least annually	Model-risk committee approval of any changes

Roles and key risk indicators

Responsibilities follow the three-lines-of-defence model used by most banks. The first line — fraud operations and the model owner — runs the monitors, reviews alarms and requests retraining. The second line — model-risk management — validates the monitoring policy before the pilot, approves each retrained model, and confirms that virtual-drift diagnoses are reasonable. Internal audit, as the third line, periodically tests that the policy operates as documented. The results of this study suggest five indicators for monthly reporting to the risk committee: recall at the alert budget on labelled data; false alerts per 1,000 transactions; the fraud prevalence z-statistic; the number of drift alarms, with their diagnosis as real or virtual; and the number of retraining events, with validation outcomes. Because the unsupervised battery did not detect a fraud pattern confined to the fraud minority, the first two, label-based indicators should carry the most weight.

Limitations

This study has four main limitations. First, it is a pre-deployment evaluation based on a simulated stream, not a report of deployment at a financial institution. Simulation makes the type and timing of every drift known, which is what allows strategies to be compared fairly, but real fraud is more varied, and the absolute values reported here should not be read as estimates of production performance [21]; the roadmap above is designed to generate that institution-specific evidence. Second, the label delay is fixed at two weeks and labels are assumed complete; in practice, labels arrive at variable delays and unreported fraud is never labelled [7]. Third, only one model family and one set of detector thresholds was evaluated; other thresholds would trade earlier detection against more false alarms. Fourth, the drift-aware strategy reacts after labels arrive, so like every supervised method it cannot prevent losses in the first weeks of a new attack.

Future Work

Three extensions follow. The most important is an institutional study: applying the first two phases of the roadmap — an offline backtest and a shadow-mode trial — on a financial institution’s own long-horizon data, ideally with documented fraud campaigns, to test whether the advantages observed here hold. Monitoring of the alerted, high-score population and of analyst feedback could be added to the drift battery to reduce its blind spot for minority-class drift. Finally, cost-sensitive retraining policies that weigh the expected fraud losses avoided against the cost of validation and deployment would turn the qualitative trade-off in Figure 6 into an explicit decision rule.

Conclusion

For financial institutions deciding whether to adopt drift-aware fraud models, this pre-deployment evaluation provides three conclusions. First, a drift-aware monitoring layer built on the temporal-drift framework of Zuberi *et al.* [1] can be expected to match the accuracy of weekly retraining — within 0.007 in average precision in this simulation while requiring 90% fewer model changes, the quantity that drives validation and governance workload. Second, label-based monitors (performance and prevalence) should drive retraining, because feature-distribution tests could not see a new attack confined to the fraud minority, while being valuable for diagnosing harmless seasonal change. Third, adoption should proceed through a governed sequence offline backtest on the institution's own data, shadow mode, pilot portfolio and production with key risk indicators so that the evidence presented here is confirmed on each institution's own portfolio before the policy influences live fraud decisions.

References

- [1] Zuberi, N., Rahman, M. M., Rashid, S. U., Ahsan, M. N., Rahman, M. S., & Mollah, M. H. O. R. (2026). A robust machine learning framework for detecting temporal drift in financial fraud prevention. *Scientific Reports*, 16, Article 27776.
- [2] Bolton, R. J., & Hand, D. J. (2002). Statistical fraud detection: A review. *Statistical Science*, 17(3), 235–255.
- [3] Dal Pozzolo, A., Caelen, O., Le Borgne, Y.-A., Waterschoot, S., & Bontempi, G. (2014). Learned lessons in credit card fraud detection from a practitioner perspective. *Expert Systems with Applications*, 41(10), 4915–4928.
- [4] Widmer, G., & Kubat, M. (1996). Learning in the presence of concept drift and hidden contexts. *Machine Learning*, 23(1), 69–101.
- [5] Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., & Bouchachia, A. (2014). A survey on concept drift adaptation. *ACM Computing Surveys*, 46(4), Article 44.
- [6] Lu, J., Liu, A., Dong, F., Gu, F., Gama, J., & Zhang, G. (2019). Learning under concept drift: A review. *IEEE Transactions on Knowledge and Data Engineering*, 31(12), 2346–2363.
- [7] 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.
- [8] Gama, J., Medas, P., Castillo, G., & Rodrigues, P. (2004). Learning with drift detection. In *Advances in Artificial Intelligence – SBIA 2004* (Lecture Notes in Computer Science, Vol. 3171, pp. 286–295). Springer.
- [9] 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). Society for Industrial and Applied Mathematics.
- [10] Carcillo, F., Dal Pozzolo, A., Le Borgne, Y.-A., Caelen, O., Mazzer, Y., & Bontempi, G. (2018). SCARFF: A scalable framework for streaming credit card fraud detection with Spark. *Information Fusion*, 41, 182–194.

- [11] Board of Governors of the Federal Reserve System & Office of the Comptroller of the Currency. (2011). *Supervisory guidance on model risk management* (SR Letter 11-7).
- [12] Webb, G. I., Hyde, R., Cao, H., Nguyen, H. L., & Petitjean, F. (2016). Characterizing concept drift. *Data Mining and Knowledge Discovery*, 30(4), 964–994.
- [13] Whitrow, C., Hand, D. J., Juszczak, P., Weston, D., & Adams, N. M. (2009). Transaction aggregation as a strategy for credit card fraud detection. *Data Mining and Knowledge Discovery*, 18(1), 30–55.
- [14] Jurgovsky, J., Granitzer, M., Ziegler, K., Calabretto, S., Portier, P.-E., He-Guelton, L., & Caelen, O. (2018). Sequence classification for credit-card fraud detection. *Expert Systems with Applications*, 100, 234–245.
- [15] Page, E. S. (1954). Continuous inspection schemes. *Biometrika*, 41(1/2), 100–115.
- [16] Yurdakul, B. (2018). *Statistical properties of population stability index* [Doctoral dissertation, Western Michigan University]. ScholarWorks at WMU.
- [17] Massey, F. J., Jr. (1951). The Kolmogorov–Smirnov test for goodness of fit. *Journal of the American Statistical Association*, 46(253), 68–78.
- [18] Lin, J. (1991). Divergence measures based on the Shannon entropy. *IEEE Transactions on Information Theory*, 37(1), 145–151.
- [19] Kolter, J. Z., & Maloof, M. A. (2007). Dynamic weighted majority: An ensemble method for drifting concepts. *Journal of Machine Learning Research*, 8, 2755–2790.
- [20] Lundberg, S. M., & Lee, S.-I. (2017). A unified approach to interpreting model predictions. *Advances in Neural Information Processing Systems*, 30, 4765–4774.
- [21] Lopez-Rojas, E. A., Elmir, A., & Axelsson, S. (2016). PaySim: A financial mobile money simulator for fraud detection. In *Proceedings of the 28th European Modeling and Simulation Symposium* (pp. 249–255).
- [22] 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. *Advances in Neural Information Processing Systems*, 30, 3146–3154.
- [23] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.
- [24] Lundberg, S. M., Erion, G., Chen, H., DeGrave, A., Prutkin, J. M., Nair, B., Katz, R., Himmelfarb, J., Bansal, N., & Lee, S.-I. (2020). From local explanations to global understanding with explainable AI for trees. *Nature Machine Intelligence*, 2(1), 56–67.
- [25] 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.