International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18(23s) 2026 pp. 1649-1663
https://doi.org/10.70917/ijcisim-2026-5720
Article

Real-Time Credit Risk Decisioning at Checkout for Short-Term Installment Products: Feature Availability Constraints, Adverse Action Explainability, and Portfolio Loss Monitoring

Vamshi Krishna Dassaraju

Independent researcher, India. Email: vamshidasarraj@gmail.com

Abstract: Extending credit at the moment of purchase compresses an underwriting decision that traditionally takes minutes or days into a window measured in milliseconds, and every part of the decision has to change to fit. This paper sets out an architecture for checkout-time credit decisioning on short-term installment products, organised around three constraints that shape it. The first is feature availability: a highly predictive signal is worth nothing at checkout if it cannot be obtained, validated, and consumed inside the time budget, so availability, freshness, and latency must be treated as first-class properties of a feature rather than as engineering details discovered late. The second is adverse action explainability, which is a legal obligation rather than a product preference; the paper argues that decline reasons must be generated programmatically from the same policy and model outputs that produced the decision, through a pre-approved mapping, so that the explanation is deterministic, reproducible, and derived from the decision rather than reconstructed after it. The third is portfolio loss monitoring, where the governing view is cohort-based by policy version, because aggregate portfolio health can remain stable while a specific approval cohort deteriorates. The work draws on checkout-time decisioning across consumer payments products between February and July 2022, covering decisioning requirements, the data needed at decision time, integration with decisioning systems, and the latency envelope. No approval rates, latency percentiles, delinquency rates, loss curves, or policy-change impacts are reported, because measurements sufficient to substantiate them were not retained. The paper is therefore presented as a decisioning architecture with a stated evaluation design, and Section 12 defines the measurements that would establish it.

Keywords: Credit decisioning, real-time underwriting, buy now pay later, installment credit, feature availability, feature store, latency budget, adverse action, reason codes, explainability, fair lending, portfolio monitoring, vintage analysis, model risk management, champion challenger

1. INTRODUCTION

1.1 A decision inside a purchase

A traditional credit decision is an event the applicant waits for. They apply, the institution assesses, and an answer arrives minutes, hours, or days later. The applicant's expectation matches the process.

A credit decision at checkout is not an event the customer waits for. It is a step inside a purchase the customer is already trying to complete, and it must return in milliseconds to a few seconds. That single change of setting propagates into everything: which data can be consulted, how the policy can be expressed, how a decline must be explained, and how the resulting portfolio has to be watched.

The consequences of exceeding the budget are not confined to a slow page. Checkout abandonment, transaction timeouts, customer friction, and lost conversion follow directly, and a decision path that intermittently exceeds its

budget produces inconsistent decisioning experiences, where the same customer receives different treatment depending on system conditions rather than on credit merit.

1.2 Scope of the work

The positions in this paper derive from work on checkout-time credit decisioning across consumer payments products between February and July 2022. That work covered product strategy, decisioning requirements, the customer experience around the decision, the risk considerations bearing on it, and the technology capabilities required to support it. In practice this meant defining eligibility and decisioning requirements, identifying which customer and transaction data had to be present at decision time, integrating those inputs with decisioning systems, and ensuring the decision returned within a latency envelope compatible with a real-time checkout.

The central design tension throughout was ensuring the decision had the right data at the point of decision while balancing approval and conversion objectives against credit and risk controls. The resulting decision then entered a longer product lifecycle, including monitoring of repayment behaviour and portfolio performance, which is why Section 8 treats monitoring as part of the decisioning system rather than as a downstream reporting function.

1.3 Contribution and scope

This paper contributes:

1. A treatment of feature availability as a first-class constraint, with the practical consequence that feature selection at checkout optimises a different objective than feature selection in batch underwriting.

2. A policy change and release model designed so that a change cannot silently shift approval rates.

3. An adverse action architecture in which reason codes are generated from the decision itself through a pre-approved mapping, producing an auditable chain from input data to notice.

4. A cohort-based portfolio monitoring design, with the specific degradation signals that indicate policy deterioration rather than normal seasoning.

5. An evaluation design specifying what must be measured to establish that the architecture performs, since this paper does not report those measurements.

What this paper does not do. It reports no measured results. Approval rates, latency percentiles, delinquency rates, loss curves, and policy-change impacts are not attributed to the work described, because documented measurements sufficient to defend specific figures were not retained. Presenting estimates as results would be worse than presenting none. Section 12 states the position in full and Section 15 records the limitations that follow.

Section 2 positions the work. Section 3 develops what checkout changes. Sections 4 to 8 treat feature availability, policy change, adverse action, portfolio monitoring, and fairness. Section 9 covers failure and fallback. Sections 10 and 11 give the evaluation design and evidence position. Sections 12 to 14 cover limitations, future work, and conclusions.

2. BACKGROUND AND RELATED WORK

Adverse action obligations. Regulation B requires that a statement of reasons for adverse action be specific and indicate the principal reasons for the action taken [1]. The Fair Credit Reporting Act imposes parallel disclosure duties where a consumer report contributed to the decision [2]. Supervisory guidance has since addressed the specific case of decisions produced by complex algorithms, taking the position that the complexity of the underlying model does not relieve the creditor of the obligation to provide specific reasons [3]. These sources establish what must be produced. They do not prescribe how a reason is derived from a model output, which is the engineering question Section 6 addresses.

Model explanation methods. Additive attribution methods provide a unified framework for assigning contribution to individual features in a model prediction [4], and local surrogate approaches offer an alternative route to per-decision explanation [5]. These methods can identify which features drove a particular score. What they do not do, and are not intended to do, is produce a legally sufficient reason: an attribution is a numeric contribution, and a reason code is an approved statement about the applicant's circumstances. The mapping between them is where the compliance obligation actually sits, and it is a design artifact rather than an output of the explanation method.

Feature infrastructure. The requirement that features be pre-computed, cached, and served within a bounded latency has produced a body of practice around feature stores and online serving layers. That practice addresses the serving problem well. What it does not address, and what Section 4 treats, is the selection problem: which features belong in the online path at all, given that including one carries a latency cost and a fallback obligation on every request whether or not the feature is present.

Model risk governance. Supervisory expectations for model risk management establish the development, validation, and ongoing monitoring disciplines that a credit model must sit within [6]. This framework is what makes the policy release process in Section 5 and the monitoring design in Section 7 obligations rather than good practice.

The gap this paper addresses is the composition. The regulatory sources define the duties, the explanation literature supplies attribution mechanics, and the governance framework defines the control environment. None of them addresses what changes when the decision has milliseconds rather than days, which is the constraint that makes the composition difficult.

3. What the Checkout Setting Changes

Three properties distinguish a checkout decision from a conventional underwriting decision, and each has architectural consequences.

The time budget is hard and externally imposed. The decision must return within milliseconds to a few seconds. This is not a service-level target that can be negotiated against decision quality, because the constraint originates in customer behaviour rather than in system design. A decision that arrives late has failed regardless of how well-judged it was.

The data set is fixed at decision time. Only signals already present can be used. There is no opportunity to request a document, wait for a bureau response that exceeds the budget, or defer to manual review without abandoning the real-time premise. This is the constraint developed in Section 4.

The decision is embedded in a commercial transaction. The customer is trying to buy something, and a decline is not a neutral outcome delivered privately but an interruption of a purchase in progress. This raises the stakes on explanation quality and on consistency, because an applicant who is declined at checkout experiences it immediately and in context.

Figure 1 shows the path and the budget it runs inside.

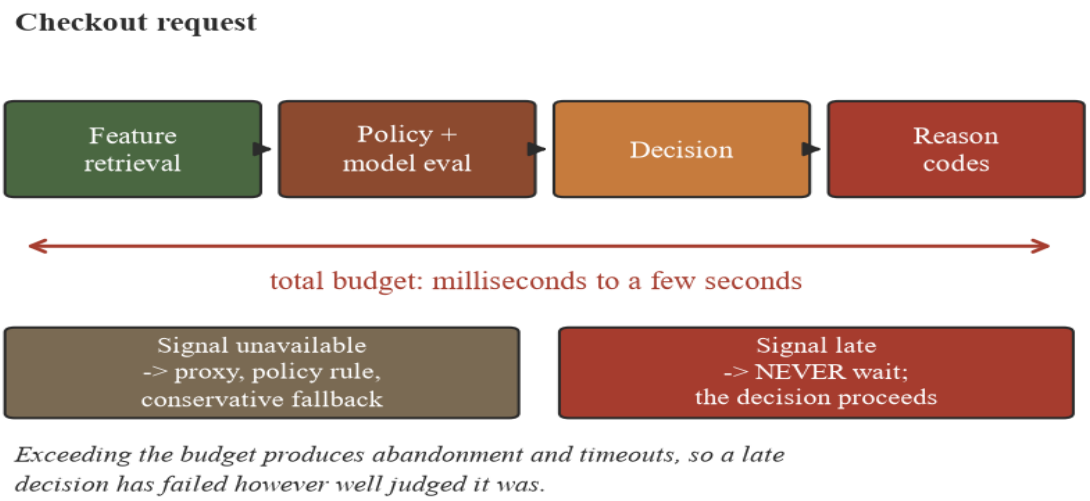


Figure 1. The decision path within the checkout budget. A signal that has not arrived is handled by proxy, rule, or conservative fallback; the decision never waits for it.

The combined effect is that the decision must be fast, accurate, explainable, and risk-controlled at once, and the design problem is that these pull against each other. Accuracy invites more data, which costs latency. Explainability constrains model form. Risk control argues for conservatism, which suppresses conversion. No single objective can be optimised without cost to the others, and the architecture's job is to make those trades explicit rather than to resolve them.

4. Feature Availability as a First-Class Constraint

4.1 What is available and what is not

The signals usable at checkout are those generated before the current credit event. Table 1 separates them from the ones that are not.

Table 1. Signal availability at decision time.

Signal class	**Available**	**Reason**
Customer identity and tenure	Yes	Established before the transaction
Historical transaction behaviour	Yes	Accumulated prior activity
Prior repayment performance	Yes	Observed on earlier obligations
Existing credit exposure	Yes	Current state of the relationship
Account activity	Yes	Pre-existing
Transaction amount, merchant, purchase characteristics	Yes	Present in the current request
Payment instrument	Yes	Present in the current request
Device and session information	Yes	Present in the current session
Established fraud and risk indicators	Yes	Pre-computed
Whether this customer will repay	No	Depends on future behaviour

Whether this customer will become delinquent	No	Depends on future behaviour
Behaviour change after receiving additional credit	No	Depends on the decision being made
External bureau, income, or behavioural data	Sometimes	May not return inside the latency window, or may not be sufficiently fresh

The distinction between the last row and the rows above it is the operationally interesting one. Future-dependent signals are unavailable in principle and no engineering removes that. External data is unavailable in practice, contingently, and that contingency is what a feature availability strategy has to manage.

4.2 The practical approach

Four practices follow. High-value features are pre-computed and cached rather than derived at request time. A reliable real-time feature layer serves them within the budget. Signals are prioritised on a combination of predictive value, freshness, availability, and latency, rather than on predictive value alone. And where a preferred signal is unavailable, the framework uses validated proxy variables, existing customer history, policy rules, and conservative fallback rather than allowing a late-arriving signal to delay the decision.

The last point is the load-bearing one and deserves stating as a principle. The decision never waits for a signal. A design that permits waiting has converted a credit architecture into an availability risk, because the slowest dependency now determines whether the checkout completes.

4.3 Fallback is a policy, not an error path

Where a preferred signal is unavailable, three responses are available and they are not equivalent. A validated proxy substitutes a correlated signal whose relationship to the missing one has been established. A policy rule applies a deterministic decision that does not depend on the missing signal. A conservative fallback declines or reduces the offer rather than approving under uncertainty.

The important design commitment is that the choice among these is made in advance and recorded, not left to whatever the code happens to do when a value is null. An unspecified fallback is still a policy; it is simply an unreviewed one, and it executes precisely when conditions are abnormal and least well understood.

This is why Section 11 treats fallback utilisation as a first-class measurement. The rate at which the platform decides by a path other than the reviewed one is a governance quantity, and it is invisible in approval-rate reporting because a fallback decision is still an approval or a decline.

4.4 Why this reorders feature selection

The consequence is that feature selection at checkout optimises a different objective than feature selection in batch underwriting. In a batch setting, a feature earns its place by predictive contribution, and availability is an implementation concern addressed afterwards. At checkout, a highly predictive feature that cannot be reliably obtained, validated, and consumed within the time budget provides little value, and a moderately predictive feature that is always present may be worth more.

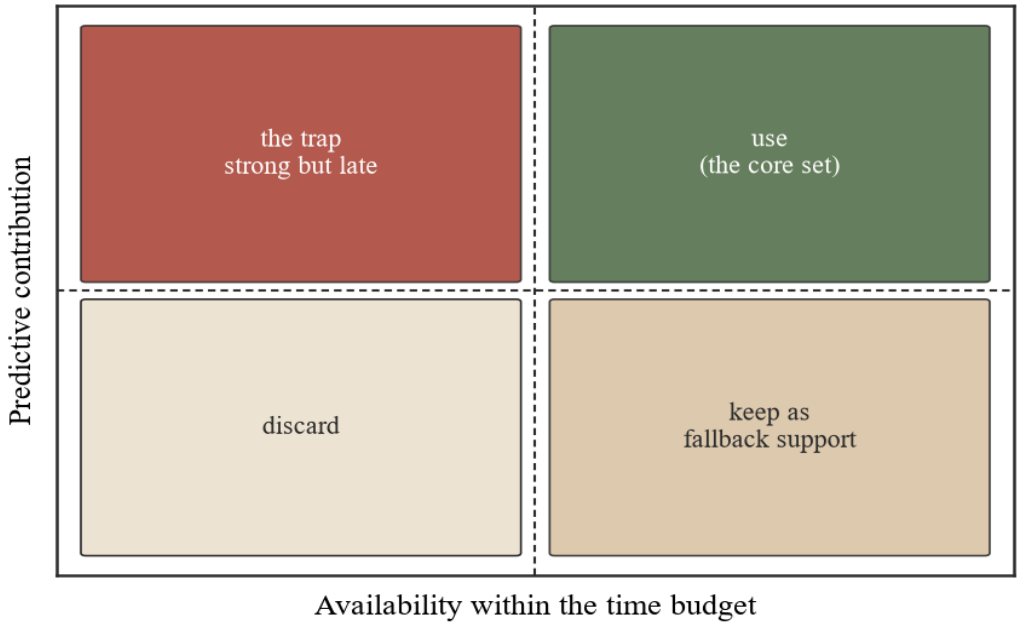


Figure 2. Feature selection at checkout. Predictive contribution alone does not determine value, because a strong signal that arrives late contributes nothing to the decision it was meant to inform.

This is not a compromise on model quality. It is a recognition that a model's realised performance is a property of the model and its inputs together, and that a feature which is absent for some fraction of requests contributes its predictive power only on the complement of that fraction, while contributing fallback complexity on all of it.

5. Expressing and Changing the Decision Policy

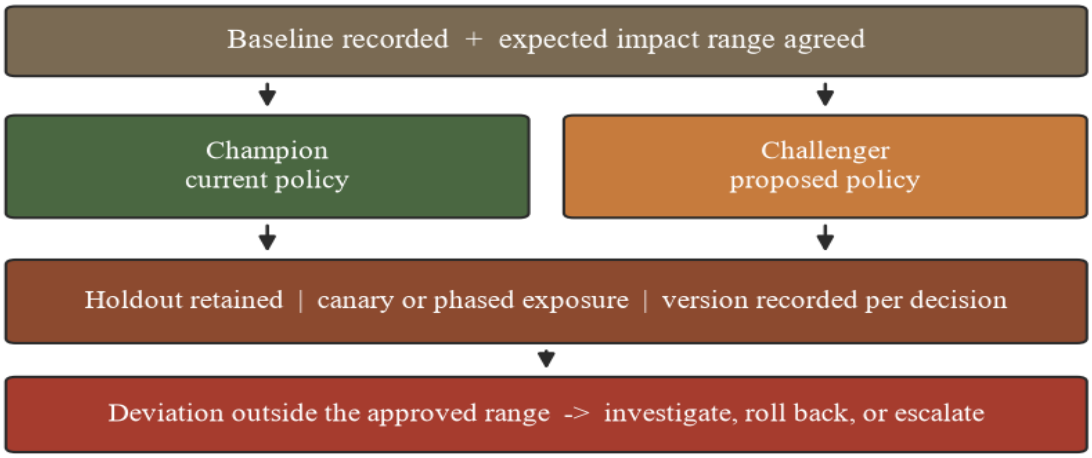
The policy is expressed through a combination of credit risk models, eligibility rules, policy thresholds, score cutoffs, pricing and limit rules, and exception or fallback logic. The engineering discipline that matters is that changes are managed as controlled policy releases rather than as direct production modifications.

Table 2. Policy change lifecycle.

Stage	**Purpose**
Business and risk requirements	Establishes intent and the expected impact range
Policy and model configuration	Expresses the change in the governed artifact rather than in code
Development and testing	Verifies the change behaves as configured
Historical and back-testing analysis	Estimates effect on approval rate, credit mix, and expected loss before exposure
Controlled deployment	Versioning, champion and challenger, canary or phased release, holdout populations
Production monitoring	Compares realised outcomes against the pre-change baseline

Testing evaluates more than model performance. It examines approval rates, credit mix, expected losses, customer segments, adverse action outcomes, and operational impact, because a change that improves discrimination while shifting the decline reason distribution has a compliance consequence that a performance metric will not surface. Approval is required from credit risk, product, compliance, and technology governance in proportion to the materiality of the change.

Preventing a silent shift in approval rate is the specific control objective. Every production change carries a documented baseline and an expected impact range. After release, approval rates and risk outcomes are monitored against that baseline, and any material deviation outside the approved range triggers investigation, rollback, or escalation.



agreed before exposure, so a deviation is detectable as a deviation rather than absorbed as the new normal.

The organising principle is that every policy change should be observable, attributable, reversible, and measurable. A change that cannot be attributed to a version, or reversed once released, is not a governed change regardless of how carefully it was reviewed beforehand.

6. Adverse Action Explainability

6.1 The requirement and the failure mode

When an applicant is declined, the reasons must be specific and must indicate the principal reasons for the action [1]. The failure mode this section guards against is an explanation that is plausible, human-authored, and not actually derived from the decision. Such an explanation may be defensible in the individual case and is indefensible as a system, because nothing connects it to what the decision logic did.

6.2 Reasons generated from the decision

Reason codes are generated programmatically from the same decision policy and model outputs that produced the decision. Each policy rule and model outcome carries a corresponding pre-approved reason-code mapping. When the decision executes, the system records the decision version, the applicable rules, the model and its version, the key contributing factors, and the resulting reason codes.

For model-based decisions, an explainability layer translates the model's contributing factors [4] into approved reason codes through a validated reason-generation methodology. The output is deterministic and reproducible for the same inputs and model version, which is the property that makes the explanation auditable rather than merely reasonable.

This produces an auditable chain, shown in Table 3.

Table 3. The adverse action derivation chain.

Stage	Artifact recorded

Input data	Values present at decision time
Features	Derived values, with availability status
Model and policy evaluation	Model version, policy version, rules evaluated
Decision	Outcome and the decision identifier
Contributing factors	Ranked factor contributions
Approved reason codes	Mapping applied, codes selected
Adverse action notice	Text delivered to the applicant

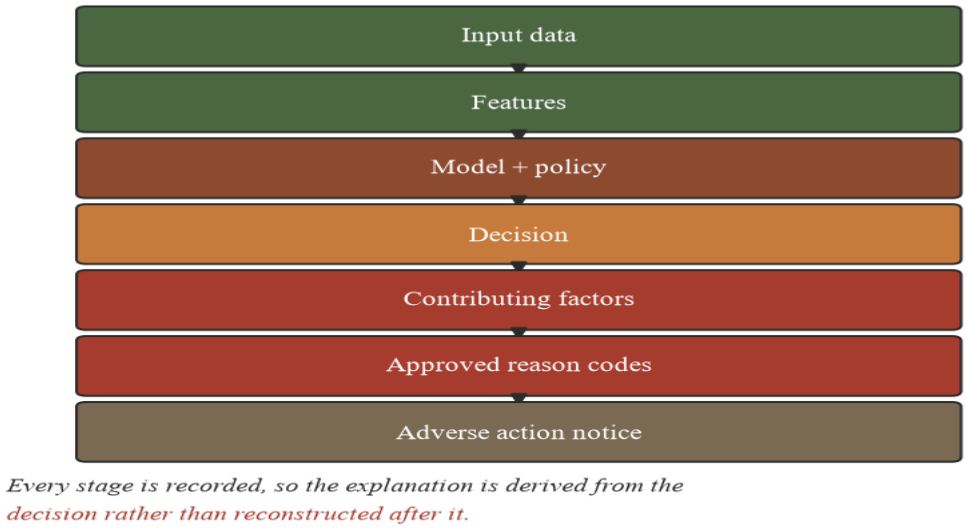


Figure 4. The adverse action derivation chain. Each stage is recorded, which is what makes the resulting explanation reproducible for the same inputs and model version.

6.3 Validation

Before production release, reason codes are validated to confirm they are accurate, understandable, supported by the actual decision logic, consistently mapped, and compliant with applicable requirements. Testing includes edge cases and regression tests, specifically to establish that a policy or model change cannot produce a reason that misdescribes the basis for the decision.

The critical control is that the explanation is generated from the decision rather than reconstructed afterwards by an analyst or an operations team. Reconstruction is the failure mode, and it is attractive precisely because it produces better-reading notices.

7. Portfolio Loss Monitoring

The credit decision is not complete when the applicant is approved. It is complete when the resulting portfolio has been shown to perform within its expected risk boundaries, which is a statement about monitoring design as much as about credit philosophy.

Cohort-based observation is the governing view. Performance is monitored by approval cohort, product, risk segment, channel, and policy or model version, covering early-stage delinquency, roll rates, charge-offs, loss curves, utilisation, repayment behaviour, and observed against expected loss. Comparing customers approved under the same policy version, tracked over consistent periods, is what distinguishes normal portfolio seasoning from genuine deterioration in decision quality. An aggregate portfolio view cannot make that distinction, because a stable aggregate is consistent with an improving old cohort masking a deteriorating new one.

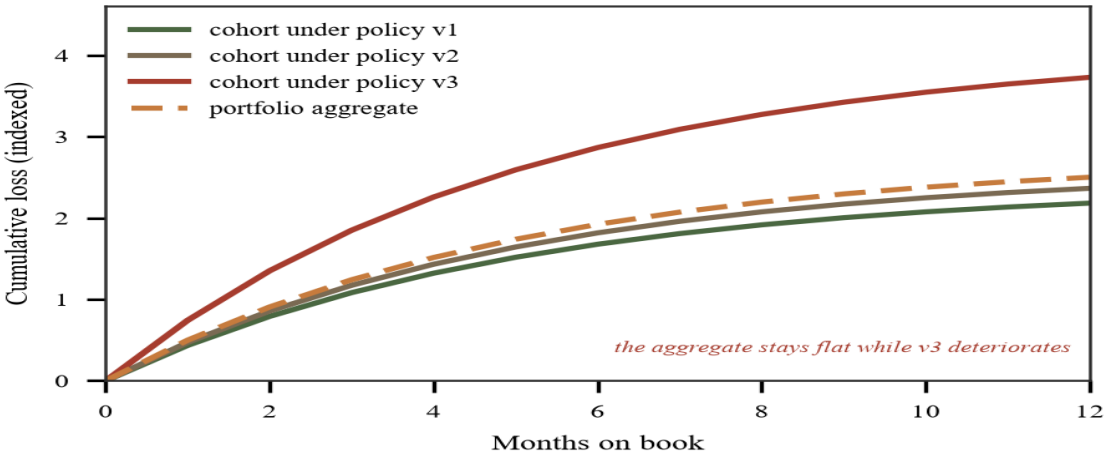


Figure 5. Why the cohort view governs. A stable portfolio aggregate is consistent with a recent policy version deteriorating, because older cohorts continue to season normally underneath it. Curves are illustrative of the mechanism, not measured.

Alongside cohort performance, the system monitors population and feature drift, approval rate changes, shifts in risk score distribution, and the relationship between predicted and observed outcomes.

Table 4. Signals that a policy has degraded.

Signal	**What it indicates**
Observed losses materially exceeding expected	The loss model no longer describes the approved population
Early delinquency rising for recent cohorts	Decision quality has deteriorated recently, ahead of loss emergence
Loss curves worsening against historical cohorts	Deterioration persists beyond early-stage noise
Approval rate moving without an intentional change	Something upstream changed: data, features, or population
Risk characteristics of approvals shifting unexpectedly	Population drift, or a policy interacting differently with it
Model discrimination or calibration deteriorating	The model has aged relative to current behaviour
Predicted and observed repayment diverging	Calibration failure, ahead of a discrimination failure

A policy can appear stable in aggregate while degrading materially within a particular customer or product segment, which is why segmentation is a monitoring requirement rather than an analytical refinement.

Thresholds precede release. Predefined thresholds and escalation triggers are established before a policy is released. When performance crosses them, the response is investigation, policy adjustment, model recalibration, or rollback, rather than waiting for losses to fully materialise. Setting thresholds after a concerning trend appears is not monitoring; it is negotiation.

8. Fairness Testing

Fairness is treated as an empirical outcome of the decisioning system rather than as a policy statement. The testing framework compares approval and decline outcomes across relevant applicant segments while controlling for differences in underlying credit risk.

The analysis covers approval and decline rates, credit limits and offer rates, pricing where applicable, adverse action reason distributions, delinquency, loss rates, and model performance across segments. It examines whether similarly situated applicants receive materially different outcomes and whether particular segments experience disproportionate declines.

Testing runs on historical samples and controlled pre-production datasets, followed by ongoing production monitoring. Results are segmented by factors relevant to the applicable fair lending framework and reviewed for differences that are both statistically and economically significant. Where a disparity appears, the analysis examines the underlying features, policy rules, model contributions, and data quality to determine whether it is attributable to legitimate risk characteristics or to an unintended decisioning effect.

Both the decision and its consequences are tested. A policy should not merely produce similar approval rates across groups; the approved populations should also show appropriately aligned risk and performance characteristics. Testing only the decision can produce a system that approves at similar rates while extending credit that performs very differently, which is a worse outcome disguised as a fairer one. Any material unexplained disparity is an investigation and remediation trigger, before or after deployment.

9. Approval, Conversion, and Where the Tension Resolves

The decision balances approval and conversion objectives against credit and risk controls, and the balance is genuinely contested rather than a matter of finding the right threshold.

The tension is structural. Conversion improves as approval widens, and credit performance deteriorates as approval reaches applicants the policy previously excluded. Both effects are real, both are measurable in principle, and they arrive on different timescales: conversion moves immediately and observably, while credit performance moves over months and becomes attributable only through the cohort analysis in Section 7.

That asymmetry is what makes the tension hard to arbitrate rather than merely hard to optimise. A policy loosening produces a visible conversion gain this week and a loss consequence that will not be reliably measurable for two or three quarters. An organisation reviewing the change at any point before then is comparing a measured benefit against an estimated cost, and estimates lose to measurements in most rooms.

Three design responses follow, none of which resolves the tension and all of which make it arbitrable.

Expected loss is estimated at the point of change, not after it. Section 5 requires back-testing analysis producing an expected impact range covering approval rate, credit mix, and expected loss before exposure. The purpose is to put a number on the delayed side of the trade at the moment the immediate side is being celebrated.

Holdouts preserve the counterfactual. A retained holdout population under the prior policy is what eventually converts the estimate into a measurement, and it has to be established at release because it cannot be reconstructed afterwards.

Cohort attribution binds the outcome to the decision. The monitoring design in Section 7 exists so that when credit performance does move, it is attributable to a specific policy version rather than to the portfolio in general.

The organisational point underneath is that the credit function and the growth function are not disagreeing about facts. They are weighting a measured near-term effect against an estimated longer-term one, and the architecture's contribution is to make the second quantity exist early enough to be weighed at all.

10. Failure Modes and Fallback Behaviour

Three failure classes recurred in real-time decisioning, and each produced an architectural response rather than a policy adjustment alone.

Overly restrictive rules suppress good customers. A rule that is defensible in isolation can decline applicants whose subsequent behaviour would have been sound. The response was to tighten thresholds against observed

performance rather than against prior expectation, which requires the cohort monitoring of Section 7 to be in place first.

Signals become unavailable or stale. A feature that is present during development is not guaranteed to be present in production indefinitely. The response was monitoring for feature availability and freshness as a first-class operational metric, and establishing explicit fail-safe behaviour for missing signals rather than allowing implicit defaults.

Fallback paths diverge from the primary policy. This is the most consequential of the three. A fallback that behaves differently from the primary path means that during degradation the platform is running an unreviewed policy, and it is running it precisely when conditions are abnormal. The response was to compare fallback decisions against the primary policy before allowing them to affect production traffic, so the fallback's behaviour is a known quantity rather than an emergent one.

The broader lesson from these is that models and policies cannot be treated as static. Customer behaviour, transaction patterns, and economic conditions change, which motivated stronger champion and challenger testing, cohort-level performance monitoring, policy versioning, and predefined rollback thresholds. The first production release is the beginning of the learning cycle rather than the end of the project.

11. Evaluation Design

Because this paper reports no measured results, it specifies instead what must be measured for the architecture to be evaluated. Table 5 defines the measurement set.

Table 5. Measurements required to evaluate the architecture.

Dimension	Measurement
Decision performance	Approval, decline, referral, and offer rates by cohort, segment, product, and policy version
Latency	Median, P95, P99, and timeout and error rates across the complete decision path
Credit performance	Early delinquency, roll rates, charge-offs, and loss rates by approval cohort and risk segment
Expected against observed	Predicted versus realised delinquency and loss by model and policy version
Policy change impact	Pre- and post-release approval rates, risk mix, conversion, and subsequent credit performance, using controlled or holdout populations
Data health	Feature availability, freshness, missingness, drift, and fallback utilisation

The evaluation method follows from the design in Section 5: establish baseline metrics before a policy change, define expected ranges and guardrails, measure the post-release population against those baselines, and retain decision-level lineage sufficient to attribute an observed change to a specific model, policy, feature, or infrastructure release.

Two of these deserve emphasis. Latency must be measured at the tail, not the median, because the median decision meets the budget in any competent implementation and the customer experience is determined by the requests that do not. Fallback utilisation must be measured continuously, because it is the rate at which the platform is making decisions by a path other than the reviewed one, and it is invisible in approval-rate reporting.

12. Where the Architecture Is Least Settled

Three parts of this design are stated with more confidence than the reasoning behind them supports, and it is better to name them than to let a reader discover them.

The proxy substitution question. Section 4 permits validated proxy variables where a preferred signal is unavailable. What validation establishes that a proxy is adequate, and how the resulting decision should be marked so that its basis differs from a decision made on the primary signal, is not specified. This matters for adverse action: if a decline rests on a proxy, the reason code should arguably reflect the proxy's meaning rather than the absent signal's, and the paper does not resolve which.

The boundary between rules and models. The policy is expressed as a combination of models, eligibility rules, thresholds, and pricing logic, and the paper treats that combination as given. Which decisions belong in a rule and which in a model is a live design question with direct explainability consequences: a rule produces a reason trivially and generalises poorly, while a model generalises well and produces a reason only through the machinery of Section 6. An architecture that pushes more decisions into rules buys explainability cheaply and loses discrimination, and the correct point on that trade is not established here.

Latency budget allocation within the decision. Section 3 fixes a total budget and the paper does not divide it. Feature retrieval, policy evaluation, model inference, and reason-code generation each consume part of it, and the allocation determines what is feasible: a model complex enough to need substantial inference time forecloses feature retrievals that would otherwise fit. Treating the budget as a single number rather than as an allocation across stages is the same error the composite budget literature identifies in another domain, and this paper commits it.

Table 6. Where the design is least settled.

Question	**Current position**	**What would settle it**
When is a proxy adequate	Validated proxies permitted, validation unspecified	A stated validation standard and a decision on reason-code treatment
Rules against models	Combination taken as given	Measured comparison of discrimination lost against explainability gained
Budget allocation within the decision	Single total	Per-stage measurement, then an allocation

None of these is a defect in the argument the paper makes. They are the questions a team implementing it would hit in the first month, and the paper's silence on them is a limit on how far it can be followed.

13. Evidence and Its Limits

What is established. The work described was performed on consumer payments products between February and July 2022, covering decisioning requirements, the data required at decision time, integration with decisioning systems, and the latency envelope for a real-time checkout experience.

What is not established. No specific approval rates, latency percentiles, delinquency rates, loss curves, or policy-change impacts are attributed to that work, because documented production measurements sufficient to defend them were not retained and could not be independently verified. Table 7 records the status of each claim class.

Table 7. Evidence status by claim class.

Claim class	**Status**
Checkout decisioning work performed on live consumer payments products	Established
Latency envelope of milliseconds to a few seconds	Established as the operating requirement. Not reported as a measured distribution
Signal availability classification in Table 1	Design analysis grounded in the work
Feature availability practices	Design position, applied in the work described

Policy release and change control model	Design position
Adverse action derivation chain	Design position, consistent with the regulatory requirement
Cohort monitoring design and degradation signals	Design position
Fairness testing framework	Design position
Failure modes in Section 9	Observed in early implementations
Approval rates, latency percentiles, delinquency, loss curves	Not measured. No figures reported
Impact of any specific policy change	Not measured

The distinction between a design position and a measured result runs through this paper, and the reason for insisting on it is domain-specific. In credit, an unsupported performance figure is not merely weak evidence; it is the kind of claim that a model risk function, an auditor, or a regulator would expect to be able to trace to a source. A paper that cannot supply that trace is better off stating the architecture and defining the measurement than supplying a number nobody can verify.

13. Practical Guidance for an Implementing Team

The architecture is described at the level of positions. A team building against it faces sequencing choices the positions do not settle, and four are worth stating.

Build the feature availability telemetry before the model. Which signals are actually present at decision time, at what freshness, for what proportion of requests, is knowable before any model is trained and it determines which features are worth training on. Teams routinely discover the availability profile after the model is built, which forces either a fallback for a feature the model depends on heavily or a retrain.

Fix the reason-code mapping before the policy, not after. Section 6 requires each rule and model outcome to carry a pre-approved reason mapping. Producing that mapping is a compliance and product exercise involving people outside engineering, and it constrains what the policy can express: a rule whose decline cannot be described in an approved reason is a rule that cannot ship. Discovering that constraint after the policy is designed produces either a rushed mapping or a policy change late.

Instrument the fallback path from the first release. The fallback executes rarely at first, which makes it easy to defer instrumenting. It also executes precisely during degradation, when the platform is least understood, and by the time its behaviour matters the opportunity to have measured it from the start has passed.

Establish the holdout at launch. Section 5 requires a baseline and expected impact range per change. A holdout population under the prior policy is the only mechanism that converts an estimated effect into a measured one, and it cannot be created retrospectively.

The common property is that all four are cheap at the start and expensive later, and all four are easy to defer because none of them is required for the first decision to be returned correctly.

14. LIMITATIONS

The decision is treated as a single event. Everything here assumes one decision at checkout with one outcome. Real installment products involve a sequence: an initial eligibility determination, a per-transaction decision, and subsequent decisions about limit changes or renewals. Each is an adverse action opportunity and each has its own feature availability profile, and the paper treats only the first substantively.

Reason-code coverage is not addressed. Section 6 requires every policy rule and model outcome to carry a pre-approved reason mapping. What happens when a decline arises from an interaction of factors that no single mapping

describes, and how the principal reasons are selected when several contribute comparably, is the hardest part of adverse action in a model-driven system and is not treated here.

No measured evaluation. The architecture is not demonstrated to outperform an alternative on any dimension. Every comparative claim is analytical.

Much of the material is prescriptive rather than descriptive. Several sections state how a component should behave. Where a practice was applied in the work described it is stated as such; elsewhere it is a design position, and the reader should not read the two registers as equivalent.

No production incident is reported. Section 9 describes recurring failure classes rather than a specific traced incident with a documented remediation. The lessons are real; their evidentiary weight is lower than an incident narrative would carry.

The latency envelope is stated, not characterised. The requirement was milliseconds to a few seconds. What the realised distribution looked like, where the tail sat, and how often the budget was exceeded are not reported.

Fairness testing is described, not evidenced. Section 8 sets out a testing framework. No results from running it are presented, and a fairness framework without results is a methodology rather than a finding.

The conversion side of the trade is not developed. Section 3 states that approval and conversion objectives pull against credit and risk controls, and the paper then treats the risk side almost exclusively. How conversion impact is estimated when a policy tightens, and who arbitrates between the two functions, is a real part of running this system and is not addressed here.

Product scope is narrower than the title suggests. The work concerned checkout-time credit decisioning on consumer payments products. Short-term installment products are the natural application and the architecture is stated in those terms, but the paper does not report installment-specific portfolio outcomes.

15. Future Work

Instrument the decision path end to end. The single most valuable measurement is the latency distribution of the complete decision path with tail percentiles and timeout rates, because it establishes whether the central architectural constraint is actually being met.

Measure fallback divergence. Quantifying how often the fallback path executes, and how its decisions differ from what the primary policy would have returned on the same inputs, converts the Section 9 position into evidence and sizes a risk that is currently unmeasured.

Establish the availability-weighted value of features. Section 4 argues that feature selection at checkout should account for availability and latency alongside predictive contribution. Formalising this as a selection criterion and measuring realised performance against a purely predictive selection would test the claim directly.

Run and report the fairness testing. Executing the framework in Section 8 and reporting the resulting distributions, including where no disparity was found, would move the strongest compliance claim in this paper from method to result.

Close the loop from decision to portfolio. Linking decision-level lineage to cohort outcomes such that a loss curve can be attributed to a specific policy version is the capability that makes everything in Section 7 actionable, and it is an engineering investment rather than an analytical one.

A note on sequencing. The five items above are not independent and the order is close to strict. Decision-path instrumentation must exist before fallback divergence can be measured, because divergence is measured against what the primary path would have returned. Decision-level lineage must exist before a loss curve can be attributed to a policy version, which means the closing item enables the fairness reporting rather than following it. An organisation attempting these in a different order will find the later ones blocked by the absence of the earlier ones, and the common

failure is starting with the fairness reporting because it is the most visible obligation and discovering that the attribution to support it was never built.

The cheapest of them, latency instrumentation across the full decision path, is also the one whose absence is least defensible, since the architecture's central constraint is a time budget nobody is currently measuring against.

16. CONCLUSION

Compressing a credit decision into a purchase changes the decision rather than merely accelerating it. The data set is fixed at request time, the policy must be expressible in a form that evaluates within the budget, the decline must carry an explanation derived from the decision itself, and the resulting portfolio must be watched by cohort because the aggregate will not reveal what a single approval vintage is doing.

Three positions carry most of the argument. Feature availability belongs in feature selection, not downstream of it, because a signal that arrives late contributes nothing and a signal that is sometimes missing contributes complexity on every request. Adverse action reasons must be generated programmatically from the decision through a pre-approved mapping, because an explanation reconstructed afterwards cannot be shown to describe what the system actually did. And the decision is not finished at approval, which makes cohort monitoring part of the decisioning architecture rather than a reporting function attached to it.

What this paper does not do is demonstrate any of this quantitatively. The work was performed on live products, and the measurements that would substantiate the architecture were not retained in a form that could be defended. Rather than supply estimates, Section 11 states what a fair evaluation would measure and Section 12 records precisely which claims currently rest on design reasoning. Instrumenting the decision path and the fallback divergence would convert the two most consequential of those into evidence, and both are ordinary engineering work rather than research.

REFERENCES

1. Consumer Financial Protection Bureau, Regulation B, Equal Credit Opportunity Act, 12 CFR 1002.9, Notifications.
2. United States Congress, Fair Credit Reporting Act, 15 U.S.C. 1681m, Requirements on users of consumer reports.
3. Consumer Financial Protection Bureau, Consumer Financial Protection Circular 2022-03: Adverse Action Notification Requirements in Connection with Credit Decisions Based on Complex Algorithms, May 2022.
4. S. M. Lundberg and S.-I. Lee, "A Unified Approach to Interpreting Model Predictions," in Advances in Neural Information Processing Systems 30, 2017. arXiv:1705.07874.
5. M. T. Ribeiro, S. Singh and C. Guestrin, "Why Should I Trust You? Explaining the Predictions of Any Classifier," in Proc. 22nd ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, 2016, pp. 1135-1144. arXiv:1602.04938.
6. Board of Governors of the Federal Reserve System and Office of the Comptroller of the Currency, Supervisory Guidance on Model Risk Management, SR 11-7 / OCC 2011-12, April 2011.