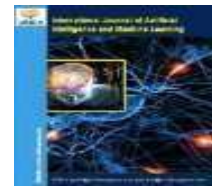




International Journal of Artificial Intelligence and Machine Learning

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



Research Paper

Open Access

Closed-Loop Order Risk Monitoring Using Statistical Learning and Operational Write-Back

Venu Gopal Kakarla

Independent Researcher, USA

Abstract

An order that is unusually large for a given customer is not, by itself, a problem: it may be a legitimate restock, a merged account's first consolidated purchase, or a data-entry error, and the three look identical at the moment of detection. Left unexamined, the third case can exhaust inventory earmarked for other customers, strain fulfillment capacity, and generate returns once the mismatch surfaces downstream. Most statistical and machine-learning approaches to this problem stop at detection: an order is scored, flagged, and handed to a person to interpret, with no formal mechanism carrying that judgment back into the enterprise systems that actually govern fulfillment. This article takes the position that detection accuracy is the easier half of the problem. The harder half is closure: building a pipeline in which a statistical score becomes a dashboard entry, an alert, and, where warranted, a hold or write-back action inside the enterprise resource planning (ERP) system that originated the order, with a human reviewer positioned at the point of highest consequence rather than at every score. Drawing on the architecture of a production order-monitoring platform built for a medical-device supply chain, the article describes a two-speed statistical design that decouples periodic model training from frequent scoring, then traces the full path from ingestion through alerting to operational write-back and production support. The contribution offered here is the closed-loop architecture itself, a pattern for converting anomaly detection into operational intelligence, adaptable to other high-volume, high-consequence transactional settings.

Keywords: order anomaly detection, closed-loop decision systems, statistical learning, operational write-back, human-in-the-loop monitoring, supply chain risk intelligence

1. Introduction

Ten times a customer's typical order volume sounds alarming until you consider what it might mean. A genuine restock ahead of a promotion looks the same, at the moment it arrives, as a newly merged account placing its first consolidated purchase, which looks the same as a data-entry error about to consume inventory meant for someone else. Recognizing that the order is statistically unusual is the easy part; most anomaly-scoring approaches manage that reliably. Deciding what happens next, before the order has already moved through fulfillment, is where the real difficulty sits.

A substantial portion of anomaly-detection work in operational settings never reaches that second question. The statistics and machine-learning literature on detecting anomalies is well established (Chandola et al., 2009), and its extension into supply-chain and transactional data has grown considerably, particularly since 2022 (Wang & Gong, 2025; Glaser et al., 2022). What has grown more slowly, in both the literature and the systems built on top of it, is the operational half: what an enterprise actually does once an order has been flagged. A dashboard nobody checks in time offers no more risk reduction than no dashboard at all.

This article takes up that second half directly. It describes the architecture of a closed-loop order-risk monitoring system, referred to here in generalized terms as an enterprise order-risk platform operating over a medical device order monitoring system, consistent with the confidentiality constraints governing this account. The system detects anomalous order-volume patterns, but its distinguishing feature is not the statistical method behind that detection, which relies on comparatively standard threshold and scoring techniques. What distinguishes it is the architecture connecting detection to action: a two-speed design that separates periodic model training from frequent scoring, closed by alerting, dashboarding, and, where the case warrants it, a hold or write-back action inside the ERP system that originated the order.

Closing the loop, not refining the detector, is this article's central claim about where the harder engineering problem actually lives. Section 2 positions that claim against three literatures (statistical anomaly detection, closed-loop and human-in-the-loop decision architectures, and production machine-learning reliability) and identifies where each stops short of the operational question this article addresses. Section 3 describes the data foundation the system draws on. Section 4 details the two-speed statistical design and its rationale. Section 5

follows the loop from score to operational action. Section 6 turns to production support: who owns a failure when one of these stages breaks, and why that question is harder than it first appears. Section 7 discusses generalizability and limitations, and Section 8 concludes.

2. Related Work

2.1 Statistical and Machine-Learning Anomaly Detection

Few problems in applied statistics have been surveyed as thoroughly as anomaly detection. Chandola et al. (2009) remains a standard reference for the field's taxonomy (point, contextual, and collective anomalies) and for a conclusion that has aged well: the right technique depends on the structure of the domain, not on any single dominant algorithm. Statistical process control, developed for manufacturing quality monitoring, has since been adapted to time-series anomaly detection more broadly (Zheng et al., 2016; Mesnil & Petitgas, 2009). Its control-chart logic, comparing a running signal against an adaptive baseline, maps well onto the order-monitoring problem: what matters is not whether an order exceeds some fixed threshold, but whether it deviates from that specific customer's own historical pattern. In supply-chain and transactional settings, recent work has begun treating anomaly detection as a first-class supply-chain risk problem in its own right rather than a generic statistics exercise borrowed wholesale (Wang & Gong, 2025; Glaser et al., 2022), a shift that brings demand seasonality and customer-level history into the scoring logic instead of leaving them out.

2.2 Closed-Loop and Human-in-the-Loop Decision Architectures

What happens after a signal fires is addressed by a different, less unified literature. Closed-loop supply chain decision-support research has mostly used "closed loop" in the sustainability and reverse-logistics sense (recovering products and materials) rather than the operational-intelligence sense intended here, though the underlying architectural concern carries over: a decision-support system delivers value only insofar as something connects its output back into an operational process (Wan et al., 2023; González Rodríguez et al., 2020). Where humans fit into that connection is the subject of a more directly relevant body of work. Mosqueira-Rey et al. (2023) and Wu et al. (2021) each frame human-in-the-loop design as a spectrum of possible intervention points rather than a binary choice between full automation and full manual review. Lazaros et al. (2026) extend that framing to high-consequence domains, treating oversight as a design requirement built in from the start rather than a fallback bolted on afterward. Bhattacharya et al. (2023), writing about manufacturing, make a related point that this article leans on directly in Section 5: a human operator adds the most value not at every decision point, but at the boundary where automated judgment is least reliable. That boundary is not approached neutrally by the people staffing it. Dietvorst et al. (2015) find that reviewers tend to abandon algorithmic recommendations after seeing even one visible error, discounting the system's output far more than its actual error rate would justify. De Jong et al. (2025) complicate that picture somewhat, finding that disclosing a model's accuracy can shift reliance in either direction depending on how the disclosure is framed, a detail that matters for how a flagged order is actually presented to a reviewer, taken up again in Section 5.2.

2.3 Production Reliability and Operational Architecture

Information-systems research on bimodal or "two-speed" architecture has generally addressed enterprise IT strategy at the organizational level rather than the internal cadence of a single statistical system (Kusanke, 2023), but the logic scales down: separating a slow, stability-oriented training cycle from a fast, responsiveness-oriented scoring cycle is a two-speed design in miniature, and Section 4 treats it on those terms. Once such a system reaches production, its reliability becomes a question of practice as much as design. Hewage and Meedeniya (2022) survey the tooling landscape for operationalizing machine learning, while Bayram and Ahmed (2025) note that production robustness has drawn less research attention than model accuracy, a gap that Section 6's discussion of failure ownership tries to address at the level of a single system. Alert volume introduces its own failure mode: Tariq et al. (2025), writing from a security-operations context, show that alert fatigue can erode the value of an automated flagging system independent of how accurate the underlying detection is, a risk equally present here. None of this matters, though, if the inputs are unreliable to begin with. Hazen et al. (2014) and Božić et al. (2024) each argue, from different vantage points, that data-quality problems in supply-chain master data are an underexamined but leading cause of downstream analytical failure, which is why Section 3 treats data foundations as a design concern rather than a preliminary formality.

Detection, human review, and production reliability are each well served by their own literature. What is comparatively rare is treating them as one connected architecture, and Table 2 summarizes where the architecture described in this article sits relative to the two more common alternatives, detection-only systems and periodic batch-review systems, before the remainder of the article describes it in full.

Table 2. Detection-only, periodic batch-review, and closed-loop approaches compared.

Dimension	Detection-Only	Periodic Batch Review	Closed-Loop Architecture) (This
Detection latency	Near-immediate scoring, no action path	Delayed until the next review cycle	Near-immediate scoring with immediate routing
Human-oversight point	None built in	Manual review of a batch, after the fact	Positioned at the highest-consequence boundary only
Downstream action	None; flag is terminal	Manual follow-up if a reviewer has time	ERP hold / release / write-back
Feedback capture	Not applicable	Rarely formalized	Reviewer overrides feed the next training cycle

3. Data Foundation for Order Risk Detection

3.1 Order and Transactional Data

The order transaction stream is the system's primary input: volume, product mix, requested delivery timing, and customer identifier, captured as each order is placed. Because the detection logic compares an incoming order against that customer's own historical pattern rather than a population-wide threshold, customer-level order history functions as a first-class input to scoring, not a reference table consulted afterward.

3.2 Inventory and Forecast Context

The same order-volume spike carries very different risk depending on what else is true at the time. Against ample on-hand inventory, it is a minor curiosity; against a forecast that already anticipated a shortage, it is a service-affecting event. The system joins order-level scoring against current inventory position and existing forecast signal for exactly this reason, so an anomaly score reflects available operational headroom as well as statistical unusualness.

3.3 Product Hierarchy and Calendar Effects

Product hierarchy data, mapping individual SKUs up into families and categories, lets the system distinguish a single-product anomaly from a pattern spanning a broader category, which in turn determines whether an alert routes to one account manager or to a wider operations review. Calendar effects, such as promotional periods, fiscal-quarter-end ordering patterns, or known seasonal shifts, are incorporated as context rather than filtered out. An order that is unusual but timed to a known calendar event should not score the same way as an identical volume spike with no such explanation.

3.4 Data-Quality Prerequisites

All of the above depends on master data that is actually consistent: product identifiers, customer hierarchies, and unit-of-measure conventions, in particular. Hazen et al. (2014) and Božić et al. (2024) both treat this as a primary risk to analytical reliability rather than a secondary one, and this system reflects that view concretely in Section 6, where data-quality failures are tracked as their own incident class rather than absorbed indiscriminately into “algorithm” failures.

4. Statistical Learning and Scoring Pattern: A Two-Speed Design

4.1 Why Two Speeds

Model training and order scoring run on two separate cadences in this architecture, and conflating them, as simpler designs sometimes do, is a mistake worth naming explicitly. Training runs weekly; scoring runs hourly, against whichever model was most recently trained. This is a two-speed design in the sense the bimodal-IT literature describes at the organizational level (Kusanke, 2023), scaled down to a single system: training is the slower, stability-oriented mode, incorporating a full week of order history before refreshing; scoring is the faster, responsiveness-oriented mode, applying that model to new orders as they arrive rather than waiting for training to catch up.

4.2 Rationale for the Weekly Training Cadence

Customer-level order-volume patterns are noisy over short horizons. A model retrained after every incoming batch would chase that noise rather than describe underlying ordering behavior. Weekly retraining gives the statistical threshold logic enough history to separate a genuine shift in a customer's pattern from short-term fluctuation, while remaining responsive enough that a sustained behavioral change is captured within a small number of training cycles.

4.3 Rationale for the Hourly Scoring Cadence

Scoring cannot wait for training's weekly rhythm without undermining the entire point of the system. An anomalous order flagged after the fulfillment window has already closed offers no real opportunity to intervene. Hourly scoring keeps the interval between order placement and risk assessment short enough that

the closed-loop actions described in Section 5, dashboard visibility, alerting, and ERP write-back, remain operationally useful rather than merely retrospective.

4.4 Statistical Threshold Logic

The scoring layer applies statistical threshold logic to each customer's own order history, following the adaptive, self-referential baseline approach found in the statistical process control literature (Zheng et al., 2016; Mesnil & Petitgas, 2009). An order is scored against a baseline reflecting that customer's typical volume, variability, and seasonal pattern, not a single global cutoff. This keeps the system's output interpretable to the operations staff reviewing it: a threshold breach can be explained in terms a reviewer already understands, namely that this order is unusually large for this particular customer, which matters considerably for the human-review stage discussed next.

4.5 Why This Is Not Simply a Better Detector

A more sophisticated model, a deep-learning sequence model, for instance, could be substituted into the scoring layer without altering anything else about the architecture. Doing so would likely improve detection at the margins. It would not touch the actual bottleneck, which is what happens to a score once one exists. Section 5 takes up that question.

5. Closed-Loop Operational Action

5.1 From Score to Consumption

A scored order is routed to three consumption points at once rather than in sequence: a shared dashboard for ongoing visibility, an automated email alert for time-sensitive cases, and a shared output file for downstream reporting (Figure 1). This parallel routing reflects a deliberate choice against forcing every stakeholder through a single interface. An analyst reviewing daily volume trends and a reviewer responding to one flagged order in real time have different needs from the same scoring event, and the architecture treats both as first-class outputs rather than ranking one above the other.

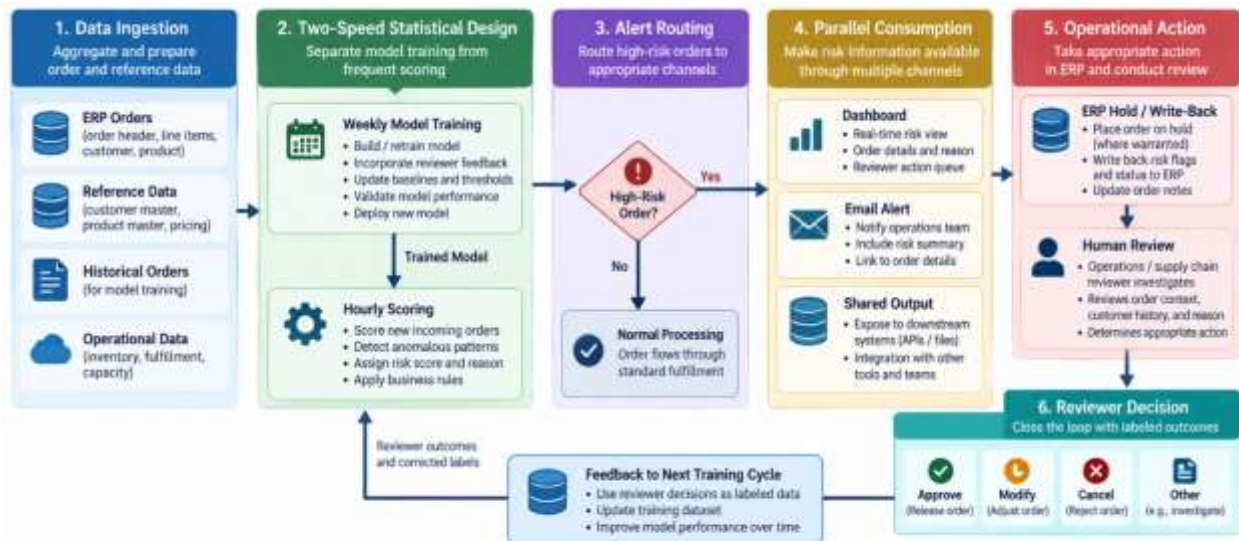


Fig. 1. Closed-loop architecture flow: ingestion, weekly training, and hourly scoring feed alert routing and parallel consumption (dashboard, email alert, shared output), which close through ERP hold/write-back and human review, with reviewer decisions feeding the next training cycle.

Figure 1. Closed-loop architecture flow: ingestion, weekly training, and hourly scoring feed alert routing and parallel consumption (dashboard, email alert, shared output), which close through ERP hold/write-back and human review, with reviewer decisions feeding the next training cycle.

5.2 Human Review as a Positioned Intervention, Not a Universal One

Most scored orders never reach a human reviewer, and that omission is deliberate rather than a shortcut taken under resource pressure. Following the human-in-the-loop literature's framing of oversight as a spectrum rather than a binary (Mosqueira-Rey et al., 2023; Bhattacharya et al., 2023), review sits at the boundary where an incorrect automated action carries the highest cost: orders that would trigger an ERP hold or write-back (Figure 2). Every other scored order clears automatically. This choice also responds to the algorithm-aversion problem raised in Section 2.2. Because a reviewer who encounters even one visible false positive tends to discount the system's output well beyond what its actual error rate justifies (Dietvorst et al., 2015), reviewer

attention is reserved for cases where a well-placed override is worth more than the trust cost a false flag imposes. The explanation package a reviewer sees alongside a flagged order follows the accuracy-disclosure findings of de Jong et al. (2025), which favor a stated confidence context over a bare flag, though the specific framing used in this system was refined iteratively through practice rather than derived directly from a single study.

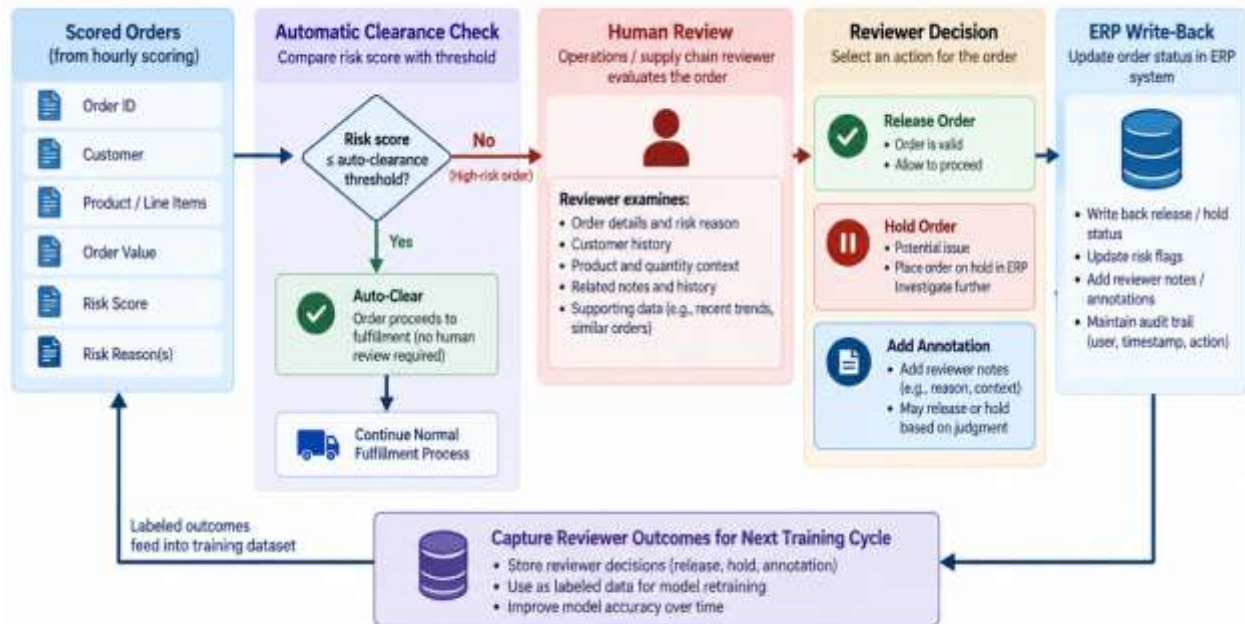


Fig. 2. Human review and ERP write-back control loop: only orders failing the auto-clearance threshold reach a human reviewer, whose decision (release, hold, or annotation) writes back into the ERP system and is captured for the next training cycle.

Figure 2. Human review and ERP write-back control loop: only orders failing the auto-clearance threshold reach a human reviewer, whose decision (release, hold, or annotation) writes back into the ERP system and is captured for the next training cycle.

5.3 ERP Write-Back as the Actual Point of Closure

Generating an alert does not close the loop. The loop closes when a decision, a reviewer's or, for the small share of cases meeting a stringent auto-clearance threshold, the system's own, is written back into the ERP system that originated the order: a hold, a release, or an annotation for downstream fulfillment staff. Most anomaly-detection literature treats this step as out of scope. It is also the step on which the system's practical value is either realized or lost. An order correctly flagged but never connected to an action inside the system of record has not, from an operational standpoint, actually been detected.

5.4 Feedback into the Next Cycle

Reviewer decisions feed back into the following week's training cycle, so a pattern of overrides on a particular category of flagged order, a seasonal promotion the model has not yet learned to expect, say, becomes something the next model version can account for rather than a recurring source of friction between the system and the people reviewing its output.

6. Production Support and Reliability

6.1 A System with Four Distinct Failure Surfaces

"Is it working" is not a useful question to ask of a system like this one, because it collapses four genuinely different failure modes into a single symptom. A fault in this architecture originates in one of four layers, algorithm, ingestion, platform, or infrastructure, and each calls for a different owner, a different diagnostic approach, and a different remediation path (Table 1).

Table 1. Production failure types and support ownership.

Failure Surface	Ownership	Diagnosis	Remediation
-----------------	-----------	-----------	-------------

Algorithm (model drift, threshold miscalibration)	Data science / architecture	Score-distribution monitoring	Off-cycle retraining or threshold adjustment
Ingestion (late, missing, or malformed source data)	Data engineering	Ingestion completeness checks	Source-system escalation or backfill
Platform (job orchestration, scheduling, pipeline failures)	Platform engineering	Job-run logs and dependency graphs	Job restart or dependency fix
Infrastructure (compute, storage, network availability)	Cloud infrastructure / DevOps	Platform health metrics	Infrastructure-team incident response

6.2 Why This Taxonomy Matters More Than It Appears To

Skip this taxonomy, and the default response pattern in most production data systems takes over: every incident routes to whichever team gets paged first, regardless of where the fault actually originated. That wastes the responding team's time and delays the real fix. Naming these four surfaces up front, and routing incidents accordingly, is a modest governance decision with an outsized effect on mean time to resolution, especially during hypercare periods right after a release, when ambiguous incidents are most common.

6.3 Timing Risk and Fallback Controls

The two-speed design from Section 4 introduces a timing risk of its own. If the weekly training job fails silently, hourly scoring keeps running against an increasingly stale model, with no visible symptom until drift has gone far enough to produce a noticeable rise in false positives or false negatives. The architecture addresses this with an explicit staleness check: scoring jobs verify the age of the model in use and raise a governance alert, distinct from the order-risk alerts described in Section 5, when a training cycle has been missed. A parallel fallback applies if the hourly scoring job itself fails: new orders route to a conservative default, flagged for manual review, rather than being allowed to pass through unscored.

6.4 Governance Beyond the Technical Layer

Reliability here is not purely a technical property. Because reviewer overrides feed back into model training (Section 5.4), a governance process is needed to periodically check whether that feedback loop is behaving as intended, whether overrides are captured consistently across reviewers, for instance, or whether one reviewer's idiosyncratic pattern is shaping the next model version more than it should. That is a governance question with a technical implementation underneath it, not a purely technical one, and the system's operating model treats it accordingly.

7. Discussion

7.1 Generalizability of the Pattern

The specific technology choices in this article, the cloud data platform, the ERP system, are incidental to the argument being made. The closed-loop pattern itself generalizes to any high-volume transactional setting where a statistical signal needs to become an operational action rather than remain an analytical artifact: frequent scoring decoupled from periodic training; parallel routing to dashboard, alert, and shared output; review positioned at the highest-consequence boundary; write-back into the system of record; and a governed feedback loop back into training. Fraud scoring, claims triage, and quality-inspection routing all share this underlying structure, even where the specific detection technique differs considerably.

7.2 Limitations

This article describes an architecture and the reasoning behind it rather than reporting a controlled evaluation of detection accuracy, false-positive rate, or return on the operational effort the system requires. Those would be legitimate follow-on studies, but they sit outside the qualitative, architecture-focused scope taken up here. The algorithm-aversion and accuracy-disclosure findings cited in Sections 2.2 and 5.2, similarly, come from literature in adjacent domains rather than from direct validation against this system's own reviewer population; the framing choices described in Section 5.2 should be read as decisions informed by that literature, not as findings established by this article itself.

7.3 The Human-Oversight Tradeoff

Positioning human review only at high-consequence boundaries reduces reviewer burden and helps contain alert fatigue (Tariq et al., 2025), but it comes at a cost worth stating plainly: the system, not a person, makes the initial judgment about which orders even reach review. That tradeoff is inherent to any human-in-the-loop design of this kind, and it deserves acknowledgment rather than being treated as fully resolved by the architecture described here.

8. Conclusion

"Is this order unusual" is the narrower and easier of the two questions this problem actually poses. The harder one, "what happens now that we know," is where most anomaly-detection systems stop short, and it is the

question this article has tried to answer with an architecture rather than a better algorithm: a two-speed statistical design, closed by ERP write-back and positioned human review. The statistical techniques underlying the scoring layer are comparatively standard. What makes the system useful in production is that a score never dead-ends in a dashboard nobody has time to check. Converting anomaly detection into operational intelligence turns out to be less a modeling achievement than an architectural one, and it is the architecture, not the detector, that this article has tried to make legible enough for someone else to build on.

Acknowledgments

The author used Claude (Anthropic, claude-sonnet model) to assist with drafting and language refinement; the conceptual framework, its architectural design decisions, and all analytical conclusions originated with and were determined by the author, who takes full responsibility for the accuracy, originality, and integrity of this manuscript, consistent with IJAIML's requirement that AI-tool use be disclosed and that AI tools not be listed as authors.

Funding

This research received no external funding.

Author Contribution Statement

Venugopal Kakarla: Conceptualization, Methodology, Investigation, Writing – Original Draft, Writing – Review & Editing, Visualization.

Conflict of Interest Statement

The author declares no conflict of interest.

Data Availability Statement

This article describes a system architecture and its design rationale, grounded in the cited literature; no proprietary or client-identifying data are disclosed, no primary numeric results are reported, and no new datasets were generated.

References

1. Bayram, F., & Ahmed, B. S. (2025). Towards trustworthy machine learning in production: An overview of the robustness in MLOps approach. *ACM Computing Surveys*, 57(5), 1–35. <https://doi.org/10.1145/3708497>
2. Bhattacharya, M., Penica, M., O'Connell, E., Southern, M., & Hayes, M. (2023). Human-in-loop: A review of smart manufacturing deployments. *Systems*, 11(1), 35. <https://doi.org/10.3390/systems11010035>
3. Božić, D., Živičnjak, M., Stanković, R., & Ignjatić, A. (2024). Impact of the product master data quality on the logistics process performance. *Logistics*, 8(2), 43. <https://doi.org/10.3390/logistics8020043>
4. Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. *ACM Computing Surveys*, 41(3), 1–58. <https://doi.org/10.1145/1541880.1541882>
5. de Jong, S., Bos, M. W., van Berkel, N., & Lamers, M. H. (2025). Algorithm appreciation or aversion: The effects of accuracy disclosure on users' reliance on algorithmic suggestions. *Behaviour & Information Technology*, 1–20. <https://doi.org/10.1080/0144929X.2025.2535732>
6. Dietvorst, B. J., Simmons, J. P., & Massey, C. (2015). Algorithm aversion: People erroneously avoid algorithms after seeing them err. *Journal of Experimental Psychology: General*, 144(1), 114–126. <https://doi.org/10.1037/xge0000033>
7. Glaser, A. E., Harrison, J. P., & Josephs, D. (2022). Anomaly detection methods to improve supply chain data quality and operations. *SMU Data Science Review*, 6(1), Article 3.
8. González Rodríguez, G., Gonzalez-Cava, J. M., & Méndez Pérez, J. A. (2020). An intelligent decision support system for production planning based on machine learning. *Journal of Intelligent Manufacturing*, 31(5), 1257–1273. <https://doi.org/10.1007/s10845-019-01510-y>
9. Hazen, B. T., Boone, C. A., Ezell, J. D., & Jones-Farmer, L. A. (2014). Data quality for data science, predictive analytics, and big data in supply chain management: An introduction to the problem and suggestions for research and applications. *International Journal of Production Economics*, 154, 72–80. <https://doi.org/10.1016/j.ijpe.2014.04.018>
10. Hewage, N., & Meedeniya, D. (2022). Machine learning operations: A survey on MLOps tool support. *arXiv:2202.10169*. <https://arxiv.org/abs/2202.10169>

11. Kusanke, K. (2023). Achieving structural ambidexterity through bimodal IT: A conceptual model and research agenda. In *Digitalization and Sustainability* (pp. 221–239). Edward Elgar Publishing. <https://doi.org/10.4337/9781800888807.00020>
12. Lazaros, K., Vrahatis, A. G., & Kotsiantis, S. (2026). Human-in-the-loop artificial intelligence: A systematic review of concepts, methods, and applications. *Entropy*, 28(4), 377. <https://doi.org/10.3390/e28040377>
13. Mesnil, B., & Petitgas, P. (2009). Detection of changes in time-series of indicators using CUSUM control charts. *Aquatic Living Resources*, 22(2), 187–192. <https://doi.org/10.1051/alr/2008058>
14. Mosqueira-Rey, E., Hernández-Pereira, E., Alonso-Ríos, D., Bobes-Bascarán, J., & Fernández-Leal, Á. (2023). Human-in-the-loop machine learning: A state of the art. *Artificial Intelligence Review*, 56(4), 3005–3054. <https://doi.org/10.1007/s10462-022-10246-w>
15. Tariq, S., Chhetri, M. B., Nepal, S., & Paris, C. (2025). Alert fatigue in security operations centres: Research challenges and opportunities. *ACM Computing Surveys*, 57(9), 1–38. <https://doi.org/10.1145/3723158>
16. Wan, X., Yang, D., Wang, T., & Deveci, M. (2023). Closed-loop supply chain decision considering information reliability and security: Should the supply chain adopt federated learning decision support systems? *Annals of Operations Research*, 349(1), 169–205. <https://doi.org/10.1007/s10479-023-05477-1>
17. Wang, R., & Gong, C. (2025). Supply chain anomaly detection and prediction models based on large-scale time series data. In *Proceedings of the 2025 International Conference on Digital Economy and Information Systems* (pp. 8–13). ACM. <https://doi.org/10.1145/3745133.3745136>
18. Wu, X., Xiao, L., Sun, Y., Zhang, J., Ma, T., & He, L. (2021). A survey of human-in-the-loop for machine learning. *arXiv:2108.00941*. <https://arxiv.org/abs/2108.00941>
19. Zheng, D., Li, F., & Zhao, T. (2016). Self-adaptive statistical process control for anomaly detection in time series. *Expert Systems with Applications*, 57, 324–336. <https://doi.org/10.1016/j.eswa.2016.03.029>