The OSCAR PUBLISHING Services
American Journal of Applied Science and Technology

VOLUME Vol.05 Issue 10 2025
PAGE NO. 317-323

Streaming Intelligence For Real-Time Fraud Detection: A Practical And Theoretical Framework Using Online Learning, Anomaly Detection, And Stream Processing

Dr. Kavita R. Iyer
Department of Computer Science and Engineering, Indian Institute of Technology Madras, India

Received: 05 September 2025; **Accepted:** 03 October 2025; **Published:** 31 October 2025

Abstract: Financial fraud has become a dynamic, high-velocity adversarial problem driven by the global scale of digital payments, card-not-present commerce, and instantaneous settlement rails. Rapid detection requires systems that combine low-latency stream processing, adaptive machine intelligence, and robust operational governance. This article develops a unified, publication-ready framework for streaming fraud intelligence that synthesizes architectural patterns (Kafka-style ingestion and materialized state), online machine learning methods (incremental learners, adaptive windowing), anomaly detection approaches (Isolation Forests, LOF), ensemble and gradient-boosted tree methods (XGBoost, LightGBM), and graph-based network detection techniques. Building on seminal and contemporary research (Rajeshwari & Babu, 2016; Carcillo et al., 2018; Bifet & Gavalda, 2007) and practitioner resources (Redis Inc., 2023; Tinybird Blog, 2023), the framework prescribes a layered pipeline: ultralow-latency fast path for authorization decisions, contextual mid-path scoring for refined risk, deferred deep analysis for network and laundering detection, and alarm-verification with human-in-the-loop adjudication. We elaborate feature-engineering patterns suitable for streaming environments (bounded sliding windows, exponential decay aggregates, probabilistic sketches), detail drift detection and mitigation strategies (adaptive windows, online weight adaptation, active learning), and discuss trade-offs among latency, accuracy, explainability, and regulatory accountability. Finally, we propose a prioritized empirical validation program—shadow deployments, synthetic adversarial injections, and federated cross-institution pilots—and operational controls for auditability and privacy. This synthesis provides researchers and practitioners with a conceptual and operational blueprint to design resilient, explainable, and deployable real-time fraud detection systems.

Keywords: Real-time fraud detection; streaming analytics; online learning; anomaly detection; Kafka; adaptive ensembles.

INTRODUCTION:

The migration of commerce into digital channels has transformed both opportunity and risk for financial institutions, merchants, and consumers. Card-present and card-not-present transactions, mobile payments, API-based banking, and instantaneous settlement rails generate a volume and velocity of events that outpace the response capabilities of traditional batch analytics and static rule systems. In such environments, fraudulent actors exploit technological scale and speed: automated account testing, synthetic identity creation, transaction laundering through networks of merchants and mule accounts, and coordinated low-and-slow campaigns that carefully evade threshold-based rules (Rajeshwari & Babu, 2016; Manoharan et al., 2024). Consequently, fraud detection has shifted from an investigatory, after-the-fact process to an operational function requiring near-real-time inference and action.

Real-time fraud detection must satisfy multiple, sometimes conflicting, objectives. Systems need to generate accurate risk signals while adding minimal latency to authorization paths, maintain robustness to evolving adversarial behaviors, limit false positives that erode customer trust, and provide explainable evidence that supports disputes and regulatory

reviews (Hanae et al., 2023; Bello et al., 2023). These requirements implicate multiple technological domains: streaming data infrastructure for high-throughput event ingestion and stateful aggregation (Dunning & Friedman, 2016), incremental or online machine learning for concept drift adaptation (Bifet & Gavalda, 2007; River Documentation, 2024), anomaly detection techniques that identify novel, previously unseen patterns (Liu et al., 2008; Breunig et al., 2000), ensemble and tree-based models for high predictive performance (Chen & Guestrin, 2016; Microsoft LightGBM, 2023), and graph analytics for network-level collusion discovery (Molloy et al., 2016).

Academic and industrial literatures document piecemeal advances: research prototypes for streaming credit-card fraud (Carcillo et al., 2018; Thennakoon et al., 2019), practitioner case studies demonstrating Kafka and SQL-oriented pipelines (Tinybird Blog, 2023), and vendor solutions promising low-latency detection with Redis or SingleStore backends (Redis Inc., 2023; SingleStore, 2023). Yet the field lacks a consolidated theoretical and practical framework that explains how to map model families to streaming primitives, design feature pipelines respecting bounded state and latency, and operationalize continuous learning safely within regulated financial operations.

This article fills that gap by presenting Streaming Intelligence for Real-Time Fraud Detection (SIRFD), a comprehensive framework that unites streaming architecture, online and offline learning methods, anomaly detection theory, and governance practices. SIRFD emphasizes: (1) layered detection pipelines (fast-path, mid-path, deferred analysis); (2) online learning and adaptive window strategies to handle concept drift; (3) hybrid anomaly detection combining density-based and isolation methods; (4) practical engineering patterns for bounded state and approximate counting; and (5) alarm verification and human adjudication to reduce false positives and provide training signals. Throughout, claims and design recommendations are grounded in the cited literature and in best practices from industrial implementations.

The remainder of the article offers deep theoretical elaboration and specific operational guidance: a methodology grounded in conceptual synthesis and engineering constraints; a rich results section that articulates the SIRFD pipeline and the expected behavioral trade-offs; a discussion that unpacks counter-arguments, regulatory concerns, and limitations; and a conclusion that outlines a prioritized empirical validation agenda.

METHODOLOGY

Because the objective is to produce an integrative theoretical and design framework rather than report a novel dataset experiment, the methodology synthesizes peer-reviewed research, conference proceedings, and practitioner documentation to derive prescriptive architectural and algorithmic patterns. The method comprises five interlocking steps: corpus curation, thematic extraction, mapping of models to streaming primitives, architectural derivation, and empirical validation planning.

Corpus curation targeted literature on streaming fraud detection, online learning, anomaly detection, tree-based and ensemble methods, and engineering reports. Key academic sources include early streaming fraud frameworks (Carcillo et al., 2018; Rajeshwari & Babu, 2016), adaptive streaming learning (Bifet & Gavalda, 2007), density and isolation anomaly detectors (Breunig et al., 2000; Liu et al., 2008), and graph analytics for transactional networks (Molloy et al., 2016). Practitioner sources—Redis Inc. (2023), SingleStore (2023), Tinybird (2023), and vendor whitepapers—were integrated to ground the framework in current deployment realities. Documentation for LightGBM and XGBoost informed pragmatic choices for mid-path scoring models (Microsoft, 2023; Chen & Guestrin, 2016).

Thematic extraction identified recurring patterns and design tensions: latency vs. model complexity, interpretability vs. accuracy, drift resilience, false-positive management, and privacy/legal constraints. These themes guided the mapping of model families to streaming primitives. For example, stateless, low-dimensional models with explainable outputs are suited for colocated fast-path inference; stateful models requiring historical aggregates are mapped to KTables or feature stores materialized from compacted topics; anomaly detectors with high compute cost are scheduled in deferred processing bins or micro-batches. Online learning frameworks (River) were considered for models needing continuous updates without full retraining (River Documentation, 2024).

Architectural derivation synthesized these mappings into the SIRFD layered pipeline, specifying dataflows, state management strategies, windowing semantics, and fallback behaviors under overload. The design emphasizes idempotent processing, bounded state, and failure recovery semantics (exactly-once where feasible). Additionally, the methodology elaborates feature engineering patterns appropriate for streaming contexts, such as exponential decay aggregates and probabilistic sketches, and prescribes

drift detection and adaptation mechanisms including adaptive windowing (ADWIN) and online weight adaptation.

Finally, empirical validation planning outlines experiments feasible in real deployments: shadowing (scoring without enforcement to measure false positives in situ), controlled injections of synthetic fraud variants (including GAN-generated adversarial samples per OpenAI and internal whitepapers), adversarial red-team exercises, and cross-institutional pilots under privacy-preserving protocols. Metrics extend classical classification measures to include time-to-detection distributions, cost-weighted utility (fraud loss prevented vs. remediation and customer loss), and operational KPIs such as review throughput and tail latency.

Throughout, the methodology explicitly records assumptions: availability of sufficient telemetry (device fingerprinting, merchant metadata), access to adjudicated labels for supervised learning and calibration, organizational MLOps maturity for model deployment and monitoring, and legal clearance for retention of audit logs under applicable privacy regimes.

RESULTS

The results of the methodological synthesis are the SIRFD architecture, a compendium of streaming feature engineering patterns, a taxonomy of model placements, adaptive learning strategies, and an empirically oriented validation program. Each component is explained in detail below.

SIRFD Layered Architecture: Dataflow and Responsibilities

SIRFD arranges processing into four primary layers plus an orthogonal governance and alarm-verification plane:

1. Event Ingestion & Normalization Layer: This layer receives raw events (authorization requests, device telemetry, authentication logs, merchant metadata) through a high-throughput broker (e.g., Kafka). Events are partitioned by a stable routing key—hashed primary account number, customer ID, or device fingerprint—to preserve per-entity ordering essential for stateful computations (Dunning & Friedman, 2016). Normalization includes schema validation, canonical merchant code mapping, and lightweight enrichment via reference data (blacklists, merchant risk scores).

2. Fast-Path (Authorization) Layer: Responsible for sub-200ms response decisions, this layer computes compact, explainable features (short window velocity counts, device match booleans, simple behavioral embeddings) and executes low-latency models—regularized logistic regression, shallow decision trees, or small gradient boosting models optimized for inference speed (Rajeshwari & Babu, 2016; Carcillo et al., 2018). Actions include allow, step-up (challenge), conditional approval with monitoring, or soft decline. The fast path prioritizes minimal latency and low false-positive costs for customer experience.

3. Contextual Enrichment & Mid-Path Scoring Layer: Running in parallel, this layer performs richer aggregation across longer windows (hourly/daily), KTable joins with persistent customer/device profiles, and computes mid-range features such as rolling spend averages, merchant chargeback rates, and cross-device counts. More expressive models (XGBoost, LightGBM) operate here under relaxed latency constraints (sub-second to a few seconds) to produce refined risk scores, prioritize human review queues, and adjust fast-path thresholds adaptively (Chen & Guestrin, 2016; Microsoft, 2023).

4. Deferred Deep Analysis & Network Detection Layer: Heavy analytic tasks—graph construction and analysis for collusion detection, deep autoencoder anomaly scoring, and adversarial synthesis—are performed asynchronously on windowed data. Graph snapshots are built from streamed edges (card–merchant, device–account) and analyzed using community detection and representation learning to find rings and wash patterns indicative of laundering (Molloy et al., 2016). Generative methods (GANs) synthesize rare fraud morphologies for augmentation and robustness testing following practices described in generative AI guidance (OpenAI, 2024).

5. Alarm-Verification & Governance Plane (Cross-Cutting): Alerts from mid- and deep layers flow into an alarm-verification pipeline combining rule-based suppression, text analytics of merchant/user notes, and verification classifiers trained on historical adjudications (Sima et al., 2018). Human analysts adjudicate prioritized alerts; their decisions feed active learning loops to improve model calibration. Governance enforces audit logging, feature provenance tags, model versioning, and retention policies for compliance.

Feature Engineering Patterns for Streaming Contexts

Feature design in streaming systems must balance informativeness with bounded state and computation. SIRFD prescribes the following patterns:

- Bounded Sliding Windows: Implement fixed

windows for velocity features (e.g., counts in last 1 minute, 10 minutes, 24 hours) with retention policies that bound state stores, enabling predictable memory usage (Carcillo et al., 2018).

- Exponential Decay Aggregation: Apply decay factors to historical aggregates to weight recent behavior more heavily, allowing sensitivity to sudden behavior shifts while maintaining long-term context (Bifet & Gavalda, 2007).
- Micro-embeddings for Behavioral Signatures: Maintain compact vector representations for customer or device behavior updated incrementally using streaming updates; these enable similarity comparisons and rapid drift detection.
- Probabilistic Sketches: Use Count-Min sketches and HyperLogLog to estimate counts and distinct counts (unique devices per account) with low memory—key for detecting diffusion patterns across actors under high throughput.
- Feature Provenance and Versioning: Tag features with transformation version, source topic, and timestamp to provide traceability for audit and explainability.

Anomaly Detection Taxonomy and Streaming Adaptation

SIRFD embraces a hybrid anomaly detection approach. Classical density-based methods (LOF) detect local deviations in density (Breunig et al., 2000), while isolation methods (Isolation Forest) are effective in high-dimensional spaces and can be adapted to streaming via incremental tree updates or periodic re-fitting on windowed data (Liu et al., 2008). Both families have complementary strengths: LOF is sensitive to local neighborhood structure, beneficial for detecting small-scale deviations; Isolation Forests scale well and are robust in many settings.

For streaming application, SIRFD recommends:

- Streaming Adaptive Windowing (ADWIN): Monitor performance and feature distributions using adaptive windowing that grows or shrinks automatically when statistical tests detect change, enabling dynamic re-training windows (Bifet & Gavalda, 2007).
- Online Isolation Ensembles: Maintain a pool of small isolation trees updated via incremental techniques or rotated windows to preserve sensitivity to new anomalies.
- Hybrid Scoring: Combine anomaly scores with supervised risk probabilities—e.g., weighted composite score where anomaly detectors increase attention to low-probability but high-uncertainty events.

Model Placement and Serving Strategies

Mapping models to stream primitives requires careful alignment of latency, state, and update semantics:

- Colocated Stateless Models: Fast-path models with no external dependencies are embedded within stream processors for minimal serialization overhead.
- Stateful Models with Feature Stores: Models requiring user history read from materialized views (KTables) or feature stores built on compacted Kafka topics; these stores guarantee available recent state under controlled retention.
- Asynchronous Heavy Models: Deep nets and graph learners are served off-path; their outputs (watchlist updates, enriched labels) feed back into mid-path models and case systems.
- Online Learners: River and similar libraries enable incremental model updates without full retraining, suitable for drift responses (River Documentation, 2024).

Adaptive Ensembles and Drift Management

Adaptation is required to handle nonstationary fraud landscapes:

- Ensemble Weight Adaptation: Adjust ensemble weights using short-window performance metrics, enabling the system to emphasize models currently most effective (Bello et al., 2024).
- Active Learning for Label Efficiency: Surface high-uncertainty cases to human reviewers to acquire labels that most improve model boundaries, crucial where fraud labels are scarce.
- Adversarial Augmentation: Use GANs and synthetic data generation to craft edge-case fraud samples, improving robustness and training coverage (OpenAI, 2024; Goodfellow et al., 2014).

Operational Controls: Latency, Throughput, and Failover

SIRFD prescribes operational controls:

- Latency Budgets and Graceful Degradation: Define strict latency SLAs for the fast path; under overload, degrade to simpler heuristics rather than blocking authorizations.
- Exactly-Once or At-Least-Once Semantics: Use exactly-once semantics where feasible for state updates to avoid duplication; otherwise enforce idempotent processing.
- Observability: Instrument model inputs, score distributions, drift metrics, and tail latencies to enable rapid diagnosis.

Empirical Validation Roadmap and Metrics

SIRFD recommends a staged validation strategy:

1. Shadow Deployment: Run models in production traffic without enforcement to measure real false positive impacts and time-to-detection in situ.

2. Synthetic Injection Tests: Inject crafted fraud patterns, including GAN-generated adversarial samples, to stress test detection under controlled conditions.

3. Adversarial Red-Team Exercises: Organize structured penetration tests simulating adaptive attackers.

4. Cross-Institution Federation Pilots: Evaluate federated learning schemes for network detection while enforcing privacy constraints.

Evaluation metrics combine classification fidelity (precision, recall, AUC) with operational KPIs (average and tail time-to-detection, analyst hours per 1,000 alerts, customer complaint rates, and cost-weighted utility).

DISCUSSION

The SIRFD framework illuminates key trade-offs, exposes practical constraints, and offers design prescriptions grounded in both theory and deployment realities. We examine these aspects in depth.

Balancing Latency and Predictive Power

High-capacity models (deep nets, graph embeddings) can detect subtle, networked fraud but cannot meet sub-200ms authorization windows. SIRFD's design compartmentalizes inference—fast, simple models make immediate calls; mid-path models refine and escalate; deep analyses run deferred. This separation maintains customer experience while enabling thorough investigative capability (Carcillo et al., 2018). A counter-argument favors deploying powerful models with hardware acceleration to meet latency; while possible, the operational cost and added engineering complexity (specialized inference clusters, GPU provisioning) make a mixed approach more pragmatic for most institutions.

Explainability and Regulatory Requirements

Regulators and customers demand explanations for decisions that affect access or financial standing. Model explainability is therefore an operational requirement (Hanae et al., 2023). SIRFD recommends explainable models for automated decisions and maintains provenance trails and human adjudication logs for higher-complexity model outputs. Explainability techniques—local surrogate models, SHAP value summaries—can be employed for both fast and mid-path models, ensuring the narrative required for disputes.

Concept Drift, Data Scarcity, and Label Quality

Fraud labels are sparse and noisy, and fraud tactics drift. Active learning and online adaptation help, but they require judicious human resources to label prioritized cases. The framework's ensemble adaptation and adaptive windowing manage drift; however, when attack morphologies change fundamentally (new channels, regulatory shifts), structural model reengineering may be required. Additionally, label quality influences model calibration—systematic biases in adjudication (e.g., over-flagging certain merchant categories) must be monitored and corrected.

Network Detection and Cross-Institution Cooperation

Many fraud schemes span institutions. Cross-institution network analysis promises significant detection gains but faces legal and privacy barriers. Federated learning and privacy-preserving aggregation are promising but bring complexity and potential statistical limitations. Legal frameworks and industry consortia are required to operationalize cross-institution intelligence without violating competition law or privacy statutes (Molloy et al., 2016).

Adversarial Arms Race and Defensive Strategies

Fraud is adversarial: attackers respond to detection improvements. Defensive strategies must therefore be dynamic: ensemble diversity, adversarial training, rapid retraining loops, and operational defenses (rate limiting, watchlists). Generative adversarial approaches both simulate emergent attacks for training and expose model vulnerabilities. Nevertheless, adaptive attackers will find new vectors, so the system must be designed for continuous evolution rather than a one-time deployment (Goodfellow et al., 2014).

Privacy, Auditability, and Evidence Retention

Financial institutions must retain sufficient evidence for disputes and regulatory audits while protecting privacy. SIRFD prescribes feature provenance, secure off-line audit stores, and cryptographic commitments to log digests where immutability is required without exposing raw PII in permanent ledgers. Data retention policies should balance forensic needs with privacy regulation (Hanae et al., 2023).

Operational Cost and Organizational Readiness

Implementing SIRFD requires significant engineering and organizational capability—stream engineering, MLOps, a fraud operations center, and legal/compliance support. The framework supports

incremental adoption—start with fast-path scoring and alarm verification, then integrate mid-path enrichment and deferred graph analytics—allowing institutions to build capability while managing risk and cost.

Limitations and Future Research Directions

SIRFD is a synthesis; empirical benchmarking across multiple payment channels, geographies, and institution sizes is required. Specific research needs include: quantitative latency-accuracy curves for model families under real load; empirical performance of online isolation forests and streaming LOF variants; evaluation of federated graph learning methods for cross-institution detection under privacy constraints; and human factors research on optimal adjudication workflows and customer communication strategies.

A prioritized research agenda includes shadow deployments in diverse production contexts, systematic adversarial testing protocols, and pilot cross-institution federated experiments to quantify detection uplift vs. privacy cost.

CONCLUSION

This article presented SIRFD, a comprehensive framework for streaming intelligence in real-time fraud detection that integrates stream processing infrastructure, adaptive machine learning, anomaly detection techniques, and governance controls. By decomposing detection into layered responsibilities—fast-path authorization, mid-path refinement, deferred deep analysis—and by prescribing practical feature engineering, adaptation strategies, and validation roadmaps, SIRFD provides both theoretical guidance and operational prescriptions. Real-world validation through shadow deployments and adversarial testing will be crucial to quantify performance and to refine the design. In the evolving adversarial landscape of digital finance, systems that combine low latency, continuous adaptation, explainability, and robust governance will be essential to protect consumers and institutions while preserving trust in digital payments.

REFERENCES

1. Arora, N., & Arora, S. (2022). Real-Time Fraud Detection Using Machine Learning and Stream Processing. ResearchGate.
2. Bifet, A., & Gavalda, R. (2007). Learning from Time-Changing Data with Adaptive Windowing. In Proceedings of the 2007 SIAM International Conference on Data Mining.
3. Breunig, M. M., Kriegel, H.-P., Ng, R. T., & Sander, J. (2000). LOF: Identifying Density-Based Local Outliers. SIGMOD Record.
4. 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.
5. Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.
6. Crawford, Redis Inc. (2023). Real-Time Fraud Detection with Redis Enterprise. Redis Inc. (Vendor documentation).
7. Dunning, T., & Friedman, E. (2016). Streaming Architecture: New Designs Using Apache Kafka and MapR Streams. O'Reilly Media.
8. Edge, M. E., Sampaio, P. R. F., & Choudhary, M. (2007). Towards a Proactive Fraud Management Framework for Financial Data Streams. In Third IEEE International Symposium on Dependable, Autonomic and Secure Computing (DASC 2007).
9. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative Adversarial Nets. Advances in Neural Information Processing Systems, 27.
10. Hanae, A. B. B. A. S. S. I., Abdellah, B. E. R. K. A. O. U. I., Saida, E. L. M. E. N. D. I. L. I., & Youssef, G. A. H. I. (2023). End-to-End Real-Time Architecture for Fraud Detection in Online Digital Transactions. International Journal of Advanced Computer Science and Applications, 14(6).
11. John, S. N., Anele, C., Kennedy, O. O., Olajide, F., & Kennedy, C. G. (2016). Realtime Fraud Detection in the Banking Sector Using Data Mining Techniques/Algorithms. In 2016 International Conference on Computational Science and Computational Intelligence (CSCI).
12. Liu, F. T., Ting, K. M., & Zhou, Z.-H. (2008). Isolation Forest. Proceedings of the 2008 Eighth IEEE International Conference on Data Mining.
13. Litty, A. (2024). Real-Time Fraud Detection in Financial Transactions: Leveraging Machine Learning and Stream Processing for Dynamic Risk Assessment. (Preprint).
14. Manoharan, G., Dharmaraj, A., Sheela, S. C., Naidu, K., Chavva, M., & Chaudhary, J. K. (2024). Machine Learning-Based Real-Time Fraud Detection in Financial Transactions. In 2024 International Conference on Advances in Computing, Communication and Applied

Informatics (ACCAI), IEEE.

15. Microsoft. (2023). LightGBM Documentation. Microsoft.
16. Molloy, I., Chari, S., Finkler, U., Wiggerman, M., Jonker, C., Habeck, T., Park, Y., Jordens, F., & Schaik, R. (2016). Graph Analytics for Real-Time Scoring of Cross-Channel Transactional Fraud.
17. OpenAI. (2024). Generative AI for Fraud Simulation and Detection. Internal whitepapers.
18. Quah, J. T., & Sriganesh, M. (2008). Real-Time Credit Card Fraud Detection Using Computational Intelligence. Expert Systems with Applications, 35(4), 1721–1732.
19. Rajeshwari, U., & Babu, B. S. (2016). Real-Time Credit Card Fraud Detection Using Streaming Analytics. In 2016 2nd International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT), IEEE.
20. River Machine Learning. (2024). River Documentation: Online Machine Learning in Python. River Project.
21. Scikit-learn Developers. (2024). Scikit-learn: Machine Learning in Python. scikit-learn.org.
22. SingleStore. (2023). On the Swipe: Real-Time Fraud Detection Case Study.
23. Singla, A., & Jangir, H. (2020). A Comparative Approach to Predictive Analytics with Machine Learning for Fraud Detection of Realtime Financial Data. In 2020 International Conference on Emerging Trends in Communication, Control and Computing (ICONC3), IEEE.
24. Sima, A.-C., et al. (2018). A Hybrid Approach for Alarm Verification Using Stream Processing, Machine Learning and Text Analytics. EDBT 2018.
25. Thennakoon, A., Bhagyani, C., Premadasa, S., Mihiranga, S., & Kuruwitaarachchi, N. (2019). Real-Time Credit Card Fraud Detection Using Machine Learning. In 2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence), IEEE.
26. Tinybird Blog. (2023). Building a Real-Time Fraud Detection System with Kafka and SQL.
27. Vinod Jain, Mayank Agrawal, & Anuj Kumar. (2020). Performance Analysis of Machine Learning Algorithms in Credit Cards Fraud Detection. Proceedings of ICRITO.
28. Redis Inc. (2023). Real-Time Fraud Detection with Redis Enterprise. Vendor documentation.
29. Alam, M. A., Nabil, A. R., Mintoo, A. A., & Islam, A. (2024). Real-Time Analytics in Streaming Big Data: Techniques and Applications. Journal of Science and Engineering Research, 1(01), 104–122.
30. Carcillo, F., et al. (2018). SCARFF: Scalable Framework for Streaming Credit Card Fraud Detection with Spark. Information Fusion, 41, 182–194.