ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 02 ISSUE: 03

RECEIVED: JULY 23 REVISED: AUGUST 11 ACCEPTED: AUGUST 27 PUBLISHED: SEPTEMBER 24

Lakehouse-Powered Payment Risk at Scale

Vinod Battapothu
Independent Researcher
vinod.battapothu.researcher@gmail.com

Abstract

The growing volumes of payment transactions have placed significant burden on payment service providers or financial institutions to evaluate the risk profiles of customers swiftly. Various risk-related metrics of customers can be obtained from the data related to their payment transactions, and the recent development of machine learning-based modeling approaches allows these metrics to be modeled as a risk score. However, the shoe has on the other foot as providing such services has become a competitive game between players leading to the demand for near-native response time in serving customer requests. The services, therefore, require a cloud-native big data architecture where the data generation, ingestion, processing and modeling operations can be carried out at huge scales and low response and lead times while integrating the required principles of data governance and privacy. Cloud-native IoT systems collect vast volumes of data daily that forms the data lake for processing. Data processing includes both batch and event-driven processing. Batch processing is used for generating risk-related metrics where as event-driven processing is used for carrying out risk scoring. The ML models behind risk metrics are provided as services and lifelong learning is done by using CI/CD pipelines. Firewall and observability adopt a zero-trust policy to promote not just security but also privacy and governance compliance. Robust architecture has enormous readability and scalability features for tackling such diverse services. The entire architecture is illustrated with an example in the domain of payment service providers using a risk scoring system proposed in the previous work.

Keywords: Payment Risk Scoring Systems, Cloud-Native Big Data Architecture, Real-Time Risk Evaluation, Payment Service Providers, Financial Transaction Analytics, Machine Learning–Based Risk Metrics, Near-Native Response Time, Data Generation And Ingestion, Event-Driven Processing, Batch Data Processing, Cloud-Native IoT Systems, Risk Scoring Services, Lifelong Machine Learning, CI/CD Pipelines For ML, Data Governance And Privacy, Zero-Trust Security Architecture, Observability And Monitoring, Scalable Financial Platforms, Payment Analytics Data Lakes, Intelligent Financial Risk Management.

1. Introduction

The combination of cloud computing and machine learning made it possible to analyze large amounts of data in an efficient and flexible way. The advent of cloud-native services serves as a catalyst for the full integration of cloud and machine learning to build end-to-end machine-learning services. Payment service providers collect and save a range of sensitive client data in registries during payment transactions. However, owing to data governance, legal, and privacy concerns, it is almost impossible to share this data among organizations. Instead of constructing machine-learning-based payment risk scores using data from multiple companies, a cloud-native big-data architecture for AI-driven payment risk scoring is proposed using only transaction data across organizations. Moreover, using such a cloud-native hybrid architecture, Scalable, unified, Data database can easily be constructed without violating data privacy; these signals can serve as hidden risk. Bank card fraud risk scoring can be realized simultaneously without violating client privacy.

When implementing machine-learning-based Digital payment risk, it is essential to define a set of risk metrics, whether for payment transaction approval or selecting merchants with transaction scale growth and negative public opinions. Although these payment algorithm risk metrics can be used across different payment service providers, the risk scoring can only be realized by a single

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 02 ISSUE: 03

RECEIVED: JULY 23 REVISED: AUGUST 11 ACCEPTED: AUGUST 27 PUBLISHED: SEPTEMBER 24

machine-learning provider who can access the underlying big data. Therefore, the core idea of machine-learning-driven payment risk-score design is that when these risk metrics are defined, the Digital risk scoring can also be aligned and calibrated according to these risk metrics but using different underlying private databases. A typical set of risk metrics for payment transaction risk, merchant risk scoring, as well as the correlated online card fraud and account parches risk scoring are defined here as an illustrative example.

1.1. Overview of Key Concepts and Objectives

AI-driven payment risk scoring is a crucial application of machine learning, where risk metrics are derived from real payment data to enable a more authentic and trustworthy scoring process. Achieving this involves addressing a range of challenges, including the design of data pipelines and the definition of risk metrics, and requires a sound architecture to support scalability. Cloud-native architectures fulfil these needs owing to their inherent features of scalability and elasticity.

The AI-driven payment risk scoring architecture is based on a reference architecture for cloud-native big data systems that systematically extends the traditional big data architectural model by applying cloud-native principles. The fundamental aspects of AI-driven payment risk scoring are also explored, and the security and privacy considerations for cloud-native AI-driven payment risk scoring systems are examined. Machine learning pipelines are a special application in AI-driven payment risk scoring that incorporate CI/CD concepts for conventional software.

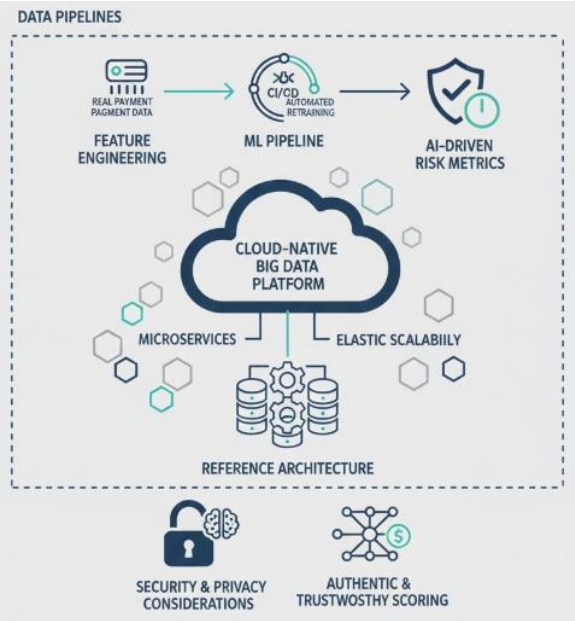


Fig 1: Cloud-Native Architectures for AI-Driven Payment Risk Scoring: Integrating Scalable Machine Learning Pipelines with CI/CD and Privacy-By-Design

2. Background and Motivation

Payment processing has evolved rapidly in recent years to keep pace with the overall increase in e-commerce. Vendors nowadays have a wide choice of payment options like credit and debit cards, net banking, mobile wallets, buy-now-pay-later, etc. However, mobile payments are expected to account for the largest share of worldwide e-commerce transaction value in the years to come. Nevertheless, a sizeable part of the user base drops off during the payment stage. This is primarily due to any one of the following reasons: Inability to make a money transfer, wrong credentials, high payment-fees, and security concerns such as identity theft, phishing, and possible fraud, or scams. Payment risk scoring is based on the estimated probability of a fraudulent payment transaction being generated by the user at the given instant.

AI-driven payment risk scoring aims to mitigate the risks involved in making a payment using various risk metrics.

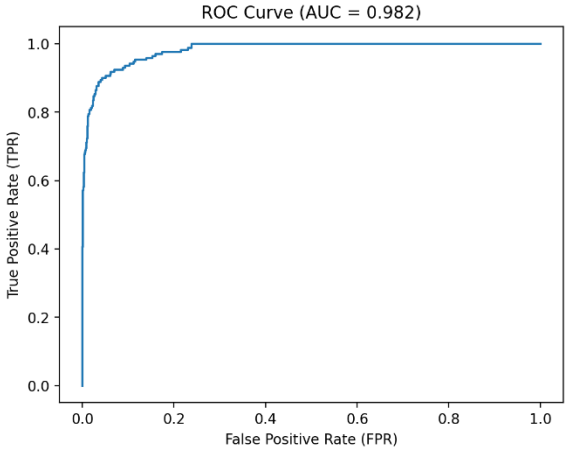
ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 02 ISSUE: 03

RECEIVED: JULY 23 REVISED: AUGUST 11 ACCEPTED: AUGUST 27 PUBLISHED: SEPTEMBER 24

AI models can estimate the risk on a large scale and improve the overall precision of the payment process. Despite the need for reliable risk estimation using AI in in-app purchases, historical payment data lacks the appropriate risk-relevant attributes that are required for model training. Therefore, historical payment transactions are enriched with risk-relevant data by leveraging RiskCloud, a third-party payment risk scoring API that provides risk metrics for a payment transaction based on user-device-geo behavior.



Equation 1: Derive TPR (Recall) step-by-step

$$TPR = \frac{TP}{TP + FN}$$

Derivation

1. **Recall / TPR** asks: *Out of all truly risky transactions, how many did we catch?*
2. "All truly risky" count is $P = TP + FN$.
3. "Caught risky" count is TP.
   So,

$$TPR = \frac{\text{caught risky}}{\text{all risky}} = \frac{TP}{TP + FN}$$

2.1. Key Concepts and Objectives Overview

Key concepts and objectives of the architecture reference are outlined. A cloud-native big-data architecture for AI-driven payment risk scoring is used as an example. The scoring metrics are primarily founded on ensemble models of machine learning methods. The intelligent scoring of payment transactions employs a machine learning pipeline that constructs an ensemble strategy with a base-level and blending-level model. The overall workflow consists of calibration and scoring procedures with respective data ingestion pipelines.

A cloud-native big-data architecture reference model is proposed that integrates and extends core design principles outlined in the Kubernetes, cloud-native, and enterprise application architecture reference models. The key cloud-native characteristics of scalability, elasticity, reliability, and fault tolerance are complemented with support for data governance, privacy, and security. These considerations encompass advanced data provenance and lineage, data quality, filtering, and masking, together with access control and compliance. The monitoring, observability, and operationalization of machine learning pipelines are built on CI/CD principles and enhanced with automated monitoring and observability features to ensure data integrity and trustworthiness.

3. Reference Architecture for Cloud-Native Big Data Systems

A cloud-native big data architecture for AI-driven payment risk scoring aims at implementing machine learning pipelines smartly deploying the models developed using analytical model-optimized concepts as part of a reference architecture satisfying popular cloud-native principles such as scalability, resistance to failure, centralized data governance, and CI/CD pipeline for machine learning. Part of the analysis introduces concepts for a big data project spanning from the ingestion to the end of the model operationalization phase. It outlines the construction of the ingestion stage, where batch ingestion is often easier and less critical than real time, as well as live streaming.

The Analysis subsequently suggests best practices and references for the Data Lake aspect of the Storage stage, but focuses mainly on the concept of Lake House,

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 02 ISSUE: 03

RECEIVED: JULY 23 REVISED: AUGUST 11 ACCEPTED: AUGUST 27 PUBLISHED: SEPTEMBER 24

combining in a smart way the benefits of low costs and fast storage and read capabilities offered by Data Lakes and Data House, respectively. The remaining of the proposed architecture satisfies the Machine Learning operationalization phase with a special emphasis on cloud-native requirements. The overall aim of the landscape is to successfully adopt machine learning as an AI technology combining technical and business area for decision-making. In this context, Risk Scoring is seen as a popular theme across diverse industries and contributing to the automation of human work.

3.1. Data Ingestion and Ingestion Pipelines

Data ingestion refers to the extraction and interception of information from source systems within a big data ecosystem in a way that allows for its subsequent utilization, storage, and transformation. A continuously running data ingestion pipeline should thus be constructed to preserve data freshness. Multiple types of systems need to be considered, as data is usually collected from a wide variety of heterogeneous structured, semi-structured, or even unstructured data sources. In addition to business events, data from third-party providers, social media, or shared IoT devices needs to be ingested into the lake. To provide an event-based and message-driven architecture that allows for efficient real-time reactions, the ingestion process is typically decoupled from the processing of the data. Data can then be produced in an event-based manner and either persisted in raw format directly after the event occurs or pushed to synchronously or asynchronously decoupled processing agents for computation. Ingestion into the lake or intermediate event store systems should thus rely on data streaming technologies to react in near real-time while still enabling a flexible buffering of the incoming data for retroactive data processing or machine learning.

Another way for slicing the data ingestion airplane contains the logic that is only responsible for providing data from one single data source direction to an output sink. Those predominantly isolated processes also come with a fault-tolerance guarantee and support a wide variety of source and sink types. When pipelines that add higher-level business meaning to the data are defined, the individual pipelines or points will thus be amenable to reuse, composition, and orchestration. One specific case that should always be supported is the periodical synchronization of a source system with such a sink but now in a pull-based rather than push-based mechanism.

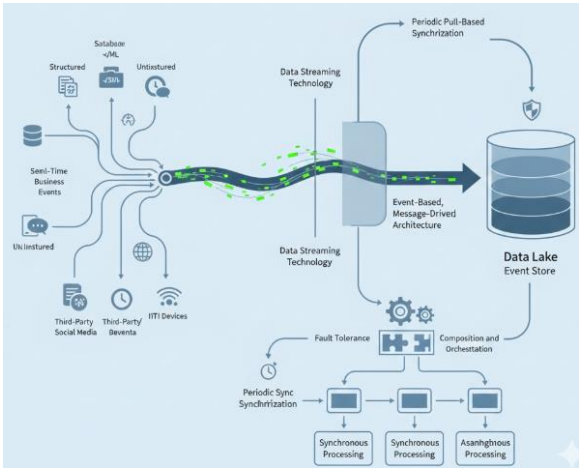


Fig 2: Decoupled and Reusable Data Ingestion Frameworks: Orchestrating Heterogeneous Event Streams for Real-Time Big Data Ecosystems

3.2. Data Storage and Lakehouse Concepts

Cloud-native systems with big-data characteristics for large-scale use cases, such as feedback loops, can be redesigned with the aid of the reference architecture. The integration of the machine learning pipelines of AI-driven products into the architecture can address the scaling and availability requirements of training and inference for algorithms that must process millions to billions of data points. Payment fraud scoring serves as an example; further granular breakdown analysis can be leveraged to provide early warnings of anomalies. These notifications can accompany real-time dashboards that alert business users of unusual patterns.

Payments represent a highly desirable and at-risk use case for online transactions. For customers, payments must be fast yet secure. Payment solution providers need to balance false positives (genuinely valid transactions that are rejected) with false negatives (fraudulent transactions that are approved), and achieve good TATs (turnaround times). For merchants and merchants' banks, high volume means

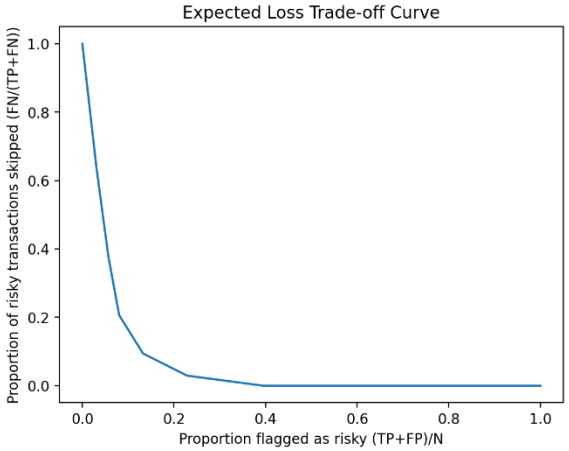
ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 02 ISSUE: 03

RECEIVED: JULY 23 REVISED: AUGUST 11 ACCEPTED: AUGUST 27 PUBLISHED: SEPTEMBER 24

profitability, while the customers' perception of security is crucial. For payment providers, fraud is damaging to reputation and revenue, while customers are interested in fast and secure transactions. Payment scoring solutions therefore calculate risk metrics for payment transactions. After calibration, the risk metric can serve as a direct feedback signal in a closed-loop system.



4. AI-Driven Payment Risk Scoring

The cloud-native big data architecture for AI-driven payment risk scoring is concerned with the scoring of applicants for payment plans of on-line retailers. A data engineering platform can be tailored for this purpose, relying on a reference architecture for cloud-native big data systems centered around machine learning. The data engineering platform introduces cloud-native considerations, including scalability, elasticity, reliability, fault tolerance, and CI/CD pipelines. The data governance framework focuses on data provenance and privacy protection of sensitive user information.
Automated machine learning serves to traditionally score prediction tasks on different evaluation metrics and for slightly modifying the prediction results in dependence on predefined criteria. Calibrated prediction scores or risk metrics ensure that a risk-aware management of applicants by the payment plan provider is supported. An Enterprise Data and Analytics Platform orchestrates the AI-driven payment risk scoring production of machine learning prediction products covering multiple business use cases and operationalizes the product development life cycles through orchestrated Pipelines-as-Code, Model-as-Code, and Model-Monitoring-as-Code procedures.
Two risk metrics, namely risk group and risk grade, are extracted from an automatically executed and continuously monitored ML pipeline. Risk groups are defined on the basis of eCQM metrics. Two risk groups are defined in the first step, where group 1 describes a high-risk applicant While group 2 describes all remaining applicants. The identified risk group is the best risk metric for exploring criterion-based calibration possibilities. A second risk metric, a risk grade with 10 levels, serves as an illustrative example for a scaled risk metric.

Equation 2: Derive FPR step-by-step

$$FPR = \frac{FP}{FP + TN}$$

Derivation

1. **FPR** asks: *Out of all truly safe transactions, how many did we incorrectly flag as risky?*
2. "All truly safe" count is $N_0 = FP + TN$.
3. Incorrectly flagged as risky count is FP.
   So,

$$FPR = \frac{\text{false alarms}}{\text{all safe}} = \frac{FP}{FP + TN}$$

4..1 Risk Metrics and Calibration

The payment risk scoring system relies on ML models that produce risk predictions in the range [0, 1], and those scores must be calibrated in a way that facilitates the proper identification of risky transactions. A typical way to define risk is to let p be a threshold (i.e., when the predicted risk is p then the transaction is considered risky) and use
- TP: the number of true positives
- FP: the number of false positives
- TN: the number of true negatives
- FN: the number of false negatives
- TPR = TP/(TP + FN) — true positive rate or recall

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

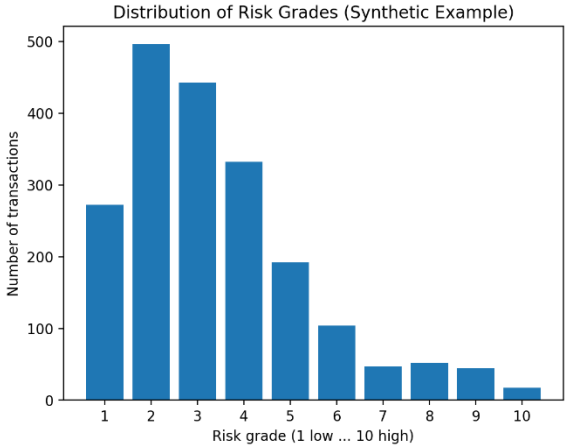
VOLUME: 02 ISSUE: 03

RECEIVED: JULY 23 REVISED: AUGUST 11 ACCEPTED: AUGUST 27 PUBLISHED: SEPTEMBER 24

- FPR = FP/(FP + TN) — false positive rate
- NPV = TN/(TN + FN) — negative predictive value
- PPV = TP/(TP + FP) — precision

Then, for each p, the Receiver Operating Characteristic (ROC) curve can be built. This curve plots the true positive rate vs. the false positive rate and provides a graphical and analytical way of selecting the optimal model and discard the suboptimal ones. The area under this curve is known as the area under the ROC Curve (AUC).

Given a set of predictions on a well-defined test set with known risk propagation, it is possible to study the risk assigned by the model and, for example, plot the expected loss — that is, the proportion of skipped risky transactions against the proportion of transactions flagged as risky — as a function of the risk threshold p. In this case, other metrics such as the Negative Predictive Value (NPV) and Positive Predictive Value (PPV) can be derived.

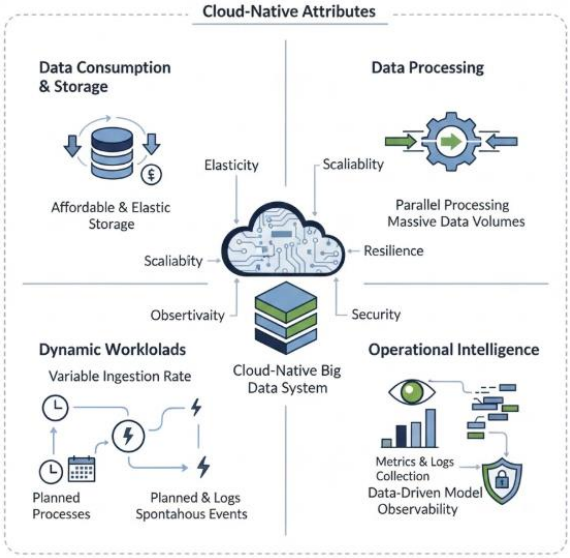


5. Cloud-Native Considerations

The cloud-native approach provides several clear advantages when building a cloud-based big data system. The attributes of cloud-native applications [14] such as elasticity, scalability, resilience, observability, and security can be applied to general cloud-based big data systems. These include scalable and elastic consumption, storing, and processing of data; reliability, durability, and fault tolerance; provisioning of components in a secure manner that prevents unauthorized access to sensitive data; and the collection of metrics and logs for monitoring the operational status of the platform and the observability of data-driven models.

Scalability refers to the ability to grow or shrink resources proportionally to the scale of data generated, either up or down. For big data systems in the cloud, it translates into the ability to scale the subsystems up and down based on the size of the data. The storage layer should allow for affordable and elastic consumption of the storage component in a pay-per-use manner and at a removable speed, while the component responsible for data processing should allow for the parallel processing of massive volumes of data for analysis at an acceptable level of quality.

Elasticity refers to the ability of the platform to provision resources as required to respond to dynamic workloads. It is particularly important for components responsible for data processing since the volume of historical data remains static, but new data is presented at highly variable rates. The variability in the speed of data being ingested may be related to both planned processes that happen periodically (monthly or yearly) and spontaneous unplanned processes, such as flash sales, that occur without anticipation.



ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 02 ISSUE: 03

RECEIVED: JULY 23 REVISED: AUGUST 11 ACCEPTED: AUGUST 27 PUBLISHED: SEPTEMBER 24

Fig 3: Architecting for Volatility: Leveraging Cloud-Native Elasticity and Scalability for Dynamic Workload Management in Big Data Systems

5.1. Scalability and Elasticity

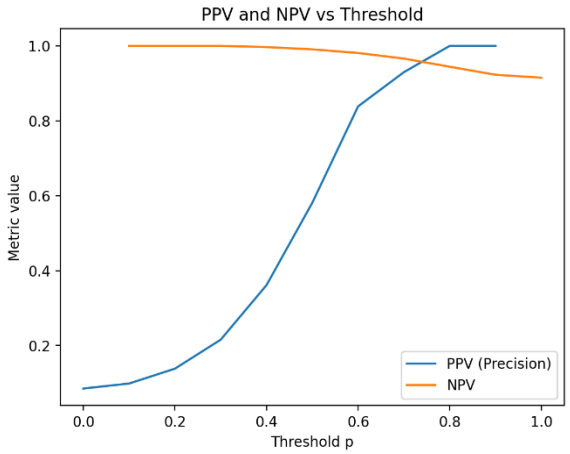
The scalability of cloud-native big data systems is easily achieved by provisioning resources according to current workloads. However, incorporating elasticity into the architecture goes a step further by automatically expanding and reducing resources in response to workload changes. The processing of payment transactions typically occurs in three stages: ingesting the transaction, processing for notation in the payment notification database, and processing a triggering periodic batch job. The load of the first stage can vary significantly according to hour and day of the week. As transaction load grows, traffic problems can arise as a result. An elastic architecture that automatically adapts resources according to transaction load potentially provides economic and performance advantages.

Gershenson and Shapiro describe the traffic characteristics in Elastic batch processing of high-scale cloud services: extreme scalability of serverless data ingestion through Amazon AWS services: Lambda, DynamoDB stream triggers, and Kinesis Streams; burstable on-demand operation for a high-scale ETL operation using AWS Batch; elastic processing of jobs with a changing load both in background and foreground; and elastic batch processing on Amazon AWS. An experimental evaluation proves that a combination of these techniques can yield a system that gracefully handles any traffic pattern while saving money. Data loading and batch job processing are processed in AWS at low cost by using the appropriate services.

5.2. Reliability and Fault Tolerance

To improve reliability within the cloud-native big data ecosystem, the system should provide mechanisms to tolerate transient faults and ensure service delivery amidst component failure. Prevention of failure and ensuring reliability of microservice-based systems deployed in cloud environments is a challenging proposition. Even in a well-designed cloud-native application, the components still rely on the operating system, networking hardware and cloud providers. Therefore, component failure is typically transient and failure is likely to affect only a subset of the service.

Impact of individual failures can usually be managed by isolating them, typically by scaling their replicas. It also follows that the overall availability of the application depends on the number of active components available to serve requests. Continuously scaling operational components when resource demand is high, and de-scaling them when demand decreases limits over-provisioning of resources, and thereby reduces costs. The actual scaling process might result in temporary degradation of the application's availability (until new component instances are provisioned) or introduce additional capacity (if additional instances are provisioned immediately before a transient demand spike). However, such capacity scaling is a transient phenomenon; a higher overall cost is incurred (albeit in a sustained way, rather than in a burst) and would presumably be acceptable to the business, given the potential improvement in service availability.



6. Data Governance and Privacy

Data governance and privacy play a vital role for every organization that handles data consequently given the importance and perhaps the importance of these aspects have more elevated when dealing with potentially sensitive personal data like those considered in the AI-driven

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 02 ISSUE: 03

RECEIVED: JULY 23 REVISED: AUGUST 11 ACCEPTED: AUGUST 27 PUBLISHED: SEPTEMBER 24

payment risk scoring. The data governance framework needs to capture data provenance and lineage as failures in these aspects can completely violate the use of data in practice as well as in regulatory and compliance aspects. The most important point for sensitive data is privacy. Sensitive data used in the risk assessment process cannot be shared freely because of privacy concerns. Local authorities managing their use of sensitive data in their border area should be aware of privacy issues and take appropriate measures to guarantee the confidentiality of individuals. The central authority has to guarantee that the implementation of the AI function training and validation process preserves the privacy of the individuals of the sensitive dataset. Sensitive data can only be used for training and validation by trusting authorities or third parties in charge of releasing the AI function and should be kept private by training the AI function in an federated learning way.

Equation 3: Derive NPV step-by-step

$$NPV = \frac{TN}{TN + FN}$$

Derivation

1. **NPV** asks: *Of everything we marked safe, how much was truly safe?*
2. "Marked safe" count is $TN + FN$.
3. Truly safe among those is TN.
   So,

$$NPV = \frac{\text{true safe among predicted safe}}{\text{all predicted safe}} = \frac{TN}{TN + FN}$$

6.1. Data Provenance and Lineage

Data provenance refers to the metadata that describes the history and origin of data in a system; who/what created it, where it resides, how it got there, how it's being used, etc. Provenance tracking and modeling is an industry-recognized technique for producing more effective data pipelines by documenting the entire flow through a system from initial creation to the present. Data lineage refers to the path that data follows through a system and the various processing steps along the way.

Lineage concerns are most prevalent in ML pipelines, in which data sources may change in unpredictable ways or pipeline operations produce unexpected side effects. Tracing lineage through the various steps of a pipeline is required in order to track down and unify the data used by training and inference executions, ensuring accuracy in risk computation by keeping track of dynamic ranges in risk metrics. ML operations (MLOps) platforms are beginning to integrate data lineage into their offerings to help ensure the validity of scoring results in production environments.

7. Deployment and Operationalization

A cloud-native architecture drawing on the reference cloud-native big data systems incorporates artificial intelligence and addresses the requirements of payment risk scoring. Machine learning is used for training-machine learning pipelines; monitoring enables the detection and recognition of changes in the data and model for the re-calibration and re-training of segments of the machine learning pipeline. Automated continuous integration and deployment for machine learning pipelines is imperative to scaling machine learning development with speed and accuracy and avoiding model drift. Machine learning model implementations must also monitor the data constantly in order to detect and respond to changes in data distribution and the resulting need for retraining. Monitoring, observability, and alerting of production payment risk scoring serve the calibration of risk segments and development of machine learning pipelines, thus providing timely detection and warnings of abnormal situations or model degradation.

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 02 ISSUE: 03

RECEIVED: JULY 23 REVISED: AUGUST 11 ACCEPTED: AUGUST 27 PUBLISHED: SEPTEMBER 24

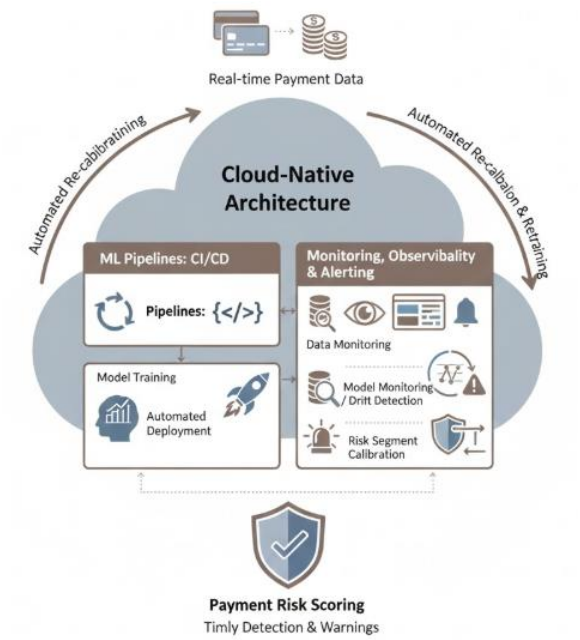


Fig 4: Adaptive Cloud-Native MLOps: Enhancing Payment Risk Scoring Through Automated CI/CD Pipelines and Continuous Drift Monitoring

7.1. CI/CD for ML Pipelines

With any model in production, it is essential to monitor its performance and possibly retrain it as new data becomes available. A simple CI/CD pipeline could be set up that monitors the model quality. Retraining would imply executing the updated pipeline steps (the whole steps regarding prediction creation) again. For retraining, it is always required first to check the previous model quality: in case that the new model performs worse than the previous one, then the previous one is simply kept. To be more robust, the model comparison could be done with stratified k-fold cross-validation and an ensemble selection method. In a context like the one described, when a payment is scored, it\ 's important to have good accuracy values for that operation to ensure that false negative detections are low. This can be achieved by training the model with a different concept of risk using a higher cost on the bottom side in risk-calibration steps. If needed, the model can also be gamed with a costly ∞ to minimize calibrating error on certain classes.

Deployment of the ML Pipelines can be done leveraging technologies such as MLflow, which enable model version leveling and tracking. By adopting it, several ML libraries can be used, such as lightgbm, catboost, or scikit-learn. Logging can be done either automatically or manually for custom ML. MLflow allows serving the ML model as a REST API, thus other applications can rely on it. The observability and interaction characteristics become even more interesting if combined with other tools, such as Streamlit, Dash, Gradio, and FLAML. Most of these tools are more oriented at interactive usage and testing. Each of them is classified according to the support for ML experiments and the final product.

7.2. Monitoring, Observability, and alerting

Operationalization of big data-driven AI services requires monitoring and observability solutions for the data pipelines, model serving, and the AI applications that consume their results. Monitoring is essential to tracking the health of the underlying services and identifying potential faults, while observability plays a larger role in investigating the root cause of performance issues.

The performance of the data ingestion pipelines in the architecture can be tracked using existing solutions, relying on the monitoring capabilities already included in current data storage technologies and frameworks. The orchestration tooling used to create the data ingestion pipelines should also provide built-in monitoring features that can become part of the overall observability solution.

Operationalizing ML pipelines requires a focus on at least two areas: reducing the time for deployment from experimentation to usage in production, and identifying potential issues tied to the model or to the data used for scoring. CI/CD methodologies should thus be applied to the ML pipelines, covering testing, validation using dedicated models, and deployment.

In production, operational monitoring involves alerting for problems associated with the ML models, including training drift and predictive accuracy drift. When detected, these alerts allow analysts to either retrain the affected model with new data that follows the same intended

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 02 ISSUE: 03

RECEIVED: JULY 23 REVISED: AUGUST 11 ACCEPTED: AUGUST 27 PUBLISHED: SEPTEMBER 24

distribution or choose an alternative model that is still valid.

8. Conclusion

The architecture described is a cloud-native big data architecture for an AI-driven payment risk scoring system that makes use of machine-learning pipelines. This architecture is conceptually based on a reference architecture for cloud-native big-data systems and for a cloud-native AI system. The key elements of this architecture include data ingestion pipelines that collect transaction data from various sources, different risk metrics such as transaction fraud scoring, recency of transaction, and geolocation risk, and techniques that improve the quality of the risk scores. The proposed architecture can be further extended to provide capabilities for monitoring, observability, and alerting of the model-prediction quality. Modern payment services involve a very large number of transactions that must be processed in an accurate and timely manner. To improve a payment service's resistance to fraud, the service must provide any risk scores and associated recommendations concerning each transaction to the transaction-processing engine. Such risk scores can be based on previously seen fraud cases as well as on the behavior of the payers and payees relative to those of previous transactions. Providing such information requires the continuous ingestion of transaction data and the safe and cost-effective storage of such data in a cloud data lake to facilitate continuous training, calibration, and deployment of an end-to-end AI service.

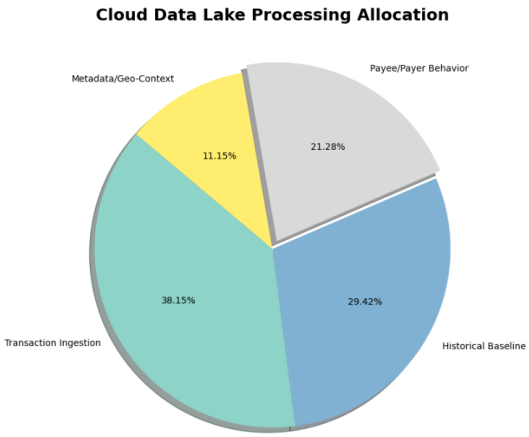


Fig 5: Cloud Data Lake Processing Allocation

8.1. Final Thoughts and Future Directions

The key innovations of the proposed architecture consist of using a model for payment risk scoring that not only meets business requirements, but can also scale from a non-AI-based credit scoring model. Such a model whose parameters can be recalibrated without changing the underlying model type can be scaled by ML pipelines for computation efficiency. Furthermore, a cloud-native architecture utilizing a ML platform to create, deploy, and operate ML pipelines enables enterprises to leverage a wider set of data sources for AI-driven decision-making while meeting the elasticity requirements for commercial data workloads. In the case where data privacy is a concern, the proposed cloud-native architecture can be deployed within a private network. Nevertheless, the cloud-native architecture still needs to fulfil data governance and operationalization requirements.

A step-by-step approach for big cloud-native AI platforms could help traders adopt AI in their decision-making processes and democratize the tools and data via a ML Ops platform. Future research could explore cross-validation of payment risk scores and data sources towards arbitrage opportunities, generate a score for originators of statistical arbitrage products, or develop products for agent banks to leverage on wholesale fund flows for personal banking.

9. References

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 02 ISSUE: 03

RECEIVED: JULY 23 REVISED: AUGUST 11 ACCEPTED: AUGUST 27 PUBLISHED: SEPTEMBER 24

[1] Breidbach, C. F., Keating, B. W., & Lim, C. (2020). Fintech service ecosystems: Value co-creation in open banking. Journal of Service Management, 31(3), 395–417.

[2] Bughin, J., Seong, J., Manyika, J., Chui, M., & Joshi, R. (2018). Notes from the AI frontier: Modeling the impact of AI on the world economy. McKinsey Global Institute.

[3] Chen, M., Mao, S., & Liu, Y. (2014). Big data: A survey. Mobile Networks and Applications, 19(2), 171–209.

[4] Chen, Y., Wang, Y., Sui, Y., & Zhang, Z. (2022). Real-time fraud detection for digital payments using ensemble learning. Knowledge-Based Systems, 236, 107741.

[5] Dastile, X., Celik, T., & Potsane, M. (2020). Statistical and machine learning models in credit card fraud detection. Applied Soft Computing, 91, 106263.

[6] Deng, X., Huang, Z., & Cheng, X. (2019). Financial fraud detection using graph neural networks. IEEE Intelligent Systems, 34(4), 41–50.

[7] Eloranta, V., Ardolino, M., Saccani, N., & Adrodegari, F. (2021). From data to value: Digital servitization in platforms. Industrial Marketing Management, 97, 38–53.

[8] Fan, W., & Bifet, A. (2013). Mining big data: Current status and forecast. ACM SIGKDD Explorations Newsletter, 14(2), 1–5.

[9] Ferreira, J. J., Fernandes, C. I., & Ferreira, F. A. (2019). To be or not to be digital, that is the question. Journal of Business Research, 101, 583–590.

[10] Florez-Lopez, R., & Ramon-Jeronimo, J. M. (2015). Enhancing accuracy in fraud detection. International Journal of Accounting Information Systems, 17, 26–37.

[11] Garcia, S., Luengo, J., & Herrera, F. (2016). Data preprocessing in data mining. Springer.

[12] He, H., & Garcia, E. A. (2009). Learning from imbalanced data. IEEE Transactions on Knowledge and Data Engineering, 21(9), 1263–1284.

[13] Huang, Y., Wu, X., & Wang, Z. (2023). Cloud-native MLOps architectures for real-time AI systems. Journal of Cloud Computing, 12(1), 1–19.

[14] Jablonski, S., & Meiler, C. (2021). Workflow management in data-intensive cloud applications. Future Generation Computer Systems, 115, 582–594.

[15] Jain, A., Duin, R. P. W., & Mao, J. (2000). Statistical pattern recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(1), 4–37.

[16] Kearns, M., & Roth, A. (2020). The ethical algorithm. Oxford University Press.

[17] Kim, Y., Cho, S., & Kim, J. (2022). Payment fraud detection using deep representation learning. Expert Systems with Applications, 195, 116569.

ISSN: 3067-4166

AMERICAN DATA SCIENCE JOURNAL FOR ADVANCED COMPUTATIONS

VOLUME: 02 ISSUE: 03

RECEIVED: JULY 23 REVISED: AUGUST 11 ACCEPTED: AUGUST 27 PUBLISHED: SEPTEMBER 24

[18] Kleppmann, M. (2017). Designing data-intensive applications. O'Reilly Media.

[19] Kotsiantis, S., Zaharakis, I., & Pintelas, P. (2006). Machine learning: A review of classification and combining techniques. Artificial Intelligence Review, 26(3), 159–190.

[20] Kou, Y., Lu, C. T., Sirwongwattana, S., & Huang, Y. P. (2004). Survey of fraud detection techniques. IEEE International Conference on Networking, Sensing and Control, 749–754.

[21] Li, J., Xu, L., & Zhao, K. (2021). Privacy-preserving analytics for financial data in cloud platforms. IEEE Transactions on Cloud Computing, 9(3), 1234–1246.

[22] Liu, Y., Chen, X., & Ward, R. (2020). Risk-aware decision systems in digital finance. Decision Support Systems, 135, 113340.

[23] Lopez-Rojas, E., Elmir, A., & Axelsson, S. (2016). PaySim: A simulated credit card transaction dataset. Proceedings of the European Modeling and Simulation Symposium, 249–255.

[24] Mishra, D., Gunasekaran, A., Papadopoulos, T., & Dubey, R. (2018). Supply chain performance and big data analytics. International Journal of Production Economics, 195, 192–204.

[25] Mitchell, T. M. (1997). Machine learning. McGraw-Hill.

[26] Moreno, A., Serrano, M., & Fernández-Caballero, A. (2022). Explainable AI in financial risk management. AI & Society, 37(4), 1535–1549.

[27] Müller, R., & Jensen, O. (2019). Big data architectures for financial analytics. Journal of Big Data, 6(1), 1–18.

[28] Ngai, E. W. T., Hu, Y., Wong, Y. H., Chen, Y., & Sun, X. (2011). The application of data mining techniques in financial fraud detection. Decision Support Systems, 50(3), 559–569.

[29] Oliveira, T., Thomas, M., Baptista, G., & Campos, F. (2016). Mobile payment adoption. Computers in Human Behavior, 61, 404–414.

[30] Pasquale, F. (2015). The black box society. Harvard University Press.

[31] Rajesh, R. (2020). Analytics-driven decision making in financial services. International Journal of Information Management, 54, 102135.

[32] Shalev-Shwartz, S., & Ben-David, S. (2014). Understanding machine learning. Cambridge University Press.

[33] Singh, A., Thakur, N., & Sharma, A. (2016). A review of supervised machine learning algorithms. International Journal of Computer Applications, 95(25), 17–24.

[34] Sun, Z., Wang, H., & Li, Y. (2023). Scalable stream processing for fraud detection in payment systems. Future Generation Computer Systems, 139, 179–191.

[35] Zhang, Y., Xiong, Y., & Zhou, W. (2024). Cloud-based intelligent risk scoring for digital payments. IEEE Transactions on Services Computing, 17(2), 456–469.