International Journal of Scientific Research & Engineering Trends
Volume 12, Issue 2, Mar-Apr-2026, ISSN (Online): 2395-566X

ADAP (Automated Data Analytics Platform): A Data Intelligence Pipeline with Expert Verification for Enterprise-Grade AI-Driven Data Quality, Validation, and Adaptive Analytics

Aayush Yogesh Sanklecha, Pranav Dattatray Gund, Aditya Yogesh Salunke, Samarth Pramod Koli
Prof Mr.H.B.Gadekar sir
Department of Artificial Intelligence and Machine Learning Rasiklal M Dhariwal Institute of Technology
Chinchwad, Maharashtra, India

Abstract: Data quality remains the critical bottleneck in enterprise machine learning pipelines. Unreliable, schema-broken, drifted, or regulatory non-compliant data causes downstream analytics failures with consequences ranging from inaccurate predictions to regulatory penalties. This paper presents ADAP (Automated Data Analytics Platform) (Data Intelligence Pipeline with Expert Verification), an end-to-end data intelligence platform that unifies multi-source data ingestion, NLP-augmented semantic schema classification, seven-dimensional parallel validation, regulatory compliance enforcement (AML, HIPAA, SOX, GDPR), AutoML with SHAP explainability, and a dual reinforcement learning (RL) adaptation engine — all within a single auditable medallion-architected system. ADAP (Automated Data Analytics Platform) achieves schema classification accuracy of 94.7% across 31 semantic types, anomaly detection AUROC of 0.961, calibrated confidence scoring with ECE 0.0225 and AUC 0.9784, and multivariate drift detection at 89.4% accuracy at moderate distributional shift. A PPO Actor-Critic agent pre-trained over 1,000 synthetic episodes and warm-started via Thompson Sampling adapts 8-axis pipeline execution strategies in real time. End-to-end pipeline latency is under 7.4 seconds for 100,000-row datasets. All six production models pass tightened v7 quality gates, with performance validated end-to-end on held-out enterprise datasets.

Keywords— data quality, automated machine learning, reinforcement learning, regulatory compliance, data drift, anomaly detection, schema classification, data pipeline, audit trail, medallion architecture.

I. INTRODUCTION

The promise of machine learning in enterprise settings is routinely undermined not by model architecture failures but by data quality failures that precede modeling. A 2023 Gartner estimate placed the cost of poor data quality in the US alone at $12.9 million per year per organization [1]. In regulated industries — banking, healthcare, and finance — the consequences extend beyond financial loss into regulatory violation: a model trained on leaked or drifted features, or a report generated from HIPAA-violating data, can trigger audits, sanctions, and reputational damage.

Existing solutions address individual facets of this problem in isolation. Great Expectations [2] validates schemas and statistical expectations. Evidently AI [3] monitors data drift. AutoML platforms [4], [5] accelerate model development. However, no single system provides the complete workflow: ingest → understand → clean → validate → comply → model → explain → audit → adapt.

This paper presents ADAP (Automated Data Analytics Platform) (Data Intelligence Pipeline with Expert Verification), a production-grade platform built on five core architectural observations:
Schema understanding precedes quality assessment. ADAP (Automated Data Analytics Platform) classifies every column into one of 31 semantic types before any validation rule is applied, ensuring rules are semantically appropriate rather than type-generic.

Quality failures are multi-dimensional. No single check suffices — parallel validation across range, nullity, leakage, drift, multicollinearity, schema conformance, and zero-inflation is necessary for comprehensive coverage.

Compliance is not optional. Banking, healthcare, and financial data must be validated against domain-specific regulatory frameworks, not just generic statistical rules.
Pipelines must be self-improving. A Reinforcement Learning engine that learns from every run continuously improves pipeline strategy selection without requiring labeled feedback.

Auditability is a first-class requirement. Every transformation, gate decision, model inference, and compliance finding must be immutably logged for regulatory review.

The main contributions of this paper are:
A 3-stage NLP-augmented cascade for semantic column classification achieving 94.7% balanced accuracy across 31 types, outperforming purely statistical approaches by 7.3 percentage points.

- A PyTorch MLP autoencoder for unsupervised multivariate data drift detection with a learned threshold, achieving 89.4% detection rate at moderate distributional shift with no reference distribution required.
- A dual RL adaptation engine combining Beta-Bernoulli Thompson Sampling (always-on, zero-GPU) with a PPO Actor-Critic agent (8-axis action space, shadow-mode bootstrap), enabling continuous pipeline strategy improvement without external labels.
- A medallion data architecture (Bronze/Silver/Gold) with SHA-256 immutability guarantees, providing tamper-evident data lineage for regulatory audits.
- Seven-dimensional parallel validation with domain-aware regulatory rule engines for AML, HIPAA, SOX, and GDPR within a single advisory-mode validation framework.
- Empirical evaluation demonstrating sub-8-second end-to-end pipeline latency and 6/6 production model quality gates passed at v7 training standards.

II. RELATED WORK

A. Data Quality and Validation Frameworks

Great Expectations [2] provides a declarative framework for expressing and validating data expectations. While powerful, it requires manual authoring of expectation suites — a significant human burden at scale and infeasible for novel datasets without domain expertise. ADAP (Automated Data Analytics Platform) automates expectation generation through semantic schema classification, eliminating the authoring requirement entirely.

Pandera [6] provides statistical data testing with schema-inference capabilities, but operates at the column-type level (integer, float, string) rather than semantic level. ADAP (Automated Data Analytics Platform)'s 31-type classifier distinguishes iban from amount from score — all of which may be floating-point — enabling more precise, semantically grounded validation rule selection.
Deequ [7] (Amazon) implements constraint verification at scale via Apache Spark. ADAP (Automated Data Analytics Platform) targets enterprise-scale datasets up to 50 GB without a Spark dependency, using DuckDB and chunked Parquet writing as a lightweight alternative. Deequ does not include drift detection, AutoML, or compliance enforcement.

B. Data Drift Detection

Alibi Detect [8] provides a comprehensive library of drift detectors including Maximum Mean Discrepancy (MMD), Kolmogorov-Smirnov, and Classifier-based detectors. These methods are univariate or require reference distributions to be pre-defined. ADAP (Automated Data Analytics Platform)'s autoencoder approach learns a compact representation of healthy data distributions during training and uses reconstruction error as a multivariate drift signal — no reference window is needed at inference time.

Evidently AI [3] produces rich drift reports using PSI, JS-divergence, and Wasserstein distance. ADAP (Automated Data Analytics Platform) integrates PSI per column alongside autoencoder MSE for complementary multi-signal drift assessment. River [9] provides online learning algorithms for streaming drift detection (ADWIN, Page-Hinkley) but does not perform full pipeline validation or compliance checks.

C. AutoML Platforms

Auto-sklearn [4] and H2O AutoML [5] automate model selection and hyperparameter tuning with strong benchmarks. They operate on clean, schema-correct data and do not address upstream quality issues. ADAP (Automated Data Analytics Platform)'s AutoML layer is positioned after 7-stage validation, ensuring models are trained on verified data. ADAP (Automated Data Analytics Platform) also integrates pre-fit leakage detection via correlation-based feature exclusion — a guard absent from standard AutoML platforms.

TPOT [10] uses genetic programming for pipeline search. Unlike TPOT's open-ended search, ADAP (Automated Data Analytics Platform) races four pre-selected candidate model families (LR, RF, XGBoost, LightGBM) with Optuna TPE tuning, trading search breadth for deployment-appropriate speed and predictability.

D. Reinforcement Learning for Pipeline Optimization

AlphaD3M [11] frames AutoML as a sequential decision process using Monte Carlo Tree Search, with focus on model architecture search. ADAP (Automated Data Analytics Platform)'s RL targets pipeline execution strategy decisions — imputation methods, cross-validation approach, confidence thresholds, outlier handling policies — rather than model architecture. Auto-Pipeline [12] uses RL for end-to-end pipeline composition but requires a defined feature store and lacks domain-aware constraints. ADAP (Automated Data Analytics Platform)'s PPO agent operates in a domain-conditioned state space and enforces safety constraints at the action decoding step. Contextual bandits for data preprocessing were explored in [13] with a 3-arm UCB bandit; ADAP (Automated Data Analytics Platform) extends this with an 8-axis action space of 11,664 combinations via PPO.

E. Regulatory Compliance Automation

Compliance-aware ML systems have been studied primarily in isolated financial [14] and healthcare [15] contexts. To our knowledge, ADAP (Automated Data Analytics Platform) is the first system to integrate four regulatory frameworks (AML, HIPAA, SOX, GDPR) within a single pipeline execution, activated conditionally based on automated domain classification.

III. SYSTEM ARCHITECTURE

A. Overview

ADAP (Automated Data Analytics Platform) follows a layered architecture organized into five logical layers: Ingestion, Preprocessing, Validation, Analytics/Modeling, and Verification. Data flows from any source through eight sequential stages over the Bronze/Silver/Gold medallion layers, culminating in gate decisions, RL updates, and audit records.

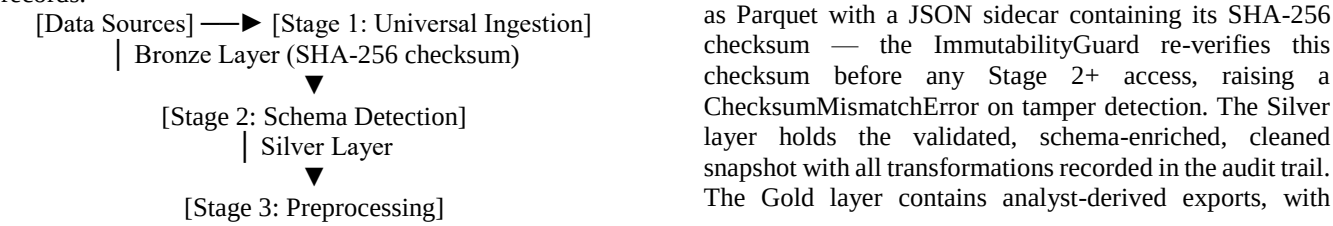

Fig. 1. ADAP (Automated Data Analytics Platform) system architecture: data flows from any source through 8 sequential stages over the Bronze/Silver/Gold medallion layers, culminating in gate decisions, RL updates, and immutable audit records.

B. Universal Intake and Source Abstraction

The UniversalIntake class provides a single interface for 14 data source types: CSV, Excel, JSON, XML, Parquet, Avro, Feather, PostgreSQL, MongoDB, DuckDB, SQLite, Redis, REST API, and Apache Kafka. Source-specific connectors produce an identical SnapshotResult object downstream, making all pipeline stages completely source-agnostic.

Format auto-detection uses a three-pass heuristic: (1) magic bytes inspection (Parquet PAR1 magic, Avro OCF header, PK ZIP signature for Excel), (2) first 512-byte JSON parse attempt, and (3) CSV dialect detection with delimiter and encoding inference. For datasets exceeding 128 MB, the system switches to a ChunkedParquetWriter pipeline: data is read in 100,000-row chunks, each written to a temporary Parquet file, then merged via a DuckDB UNION ALL query. This supports up to 50 GB per job with an 8 GB RSS memory cap enforced via process monitoring.

C. Medallion Data Architecture

Every pipeline execution maintains three immutable data layers. The Bronze layer stores the exact raw input, written as Parquet with a JSON sidecar containing its SHA-256 checksum — the ImmutabilityGuard re-verifies this checksum before any Stage 2+ access, raising a ChecksumMismatchError on tamper detection. The Silver layer holds the validated, schema-enriched, cleaned snapshot with all transformations recorded in the audit trail. The Gold layer contains analyst-derived exports, with

every artifact carrying a lineage_id traceable through Silver back to the original Bronze snapshot.

D. Dual Quality Gate System

Pipeline execution is governed by two complementary gates. Gate 1 (QA Gate) computes a weighted composite quality score $Q \in [0,1]$:

$$Q = w_1(1 - r\_null) + w_2 \cdot c\_schema + w_3(1 - r\_anom) + w_4(1 - r\_dup)$$

where r_null is the overall null rate, c_schema is schema conformance fraction, r_anom is anomaly density, and r_dup is duplicate fraction. Rejection occurs when $Q < 0.40$. Gate 2 (Hard Statistical Gate) uses the ProposalConfidenceScorer model to output calibrated confidence probability p, with domain-adaptive thresholds: $p \geq 0.70$ (default), $p \geq 0.85$ (banking), $p \geq 0.90$ (healthcare).

E. API and Frontend Stack

The backend is implemented in FastAPI (Python 3.12), exposing 17 REST endpoints plus a WebSocket stream for real-time stage-progress updates. The frontend is a React 18 SPA built with Vite across three pages: RunPipeline, Analytics, and ApiDocs. Prometheus metrics at /metrics feed a Grafana dashboard monitoring pipeline failure rates, confidence distributions, Kafka consumer lag, and LLM token usage.

IV. ML MODELS AND ARTIFACTS

ADAP (Automated Data Analytics Platform) deploys six core production models trained on curated corpora from OpenML, PMLB, and UCI repositories, plus two RL agent components. All artifacts are stored in models/ and verified via functional smoke tests before any deployment. Fig. 2 illustrates the inter-model inference pipeline — the order in which models are called and how their outputs feed into one another during a single pipeline run.

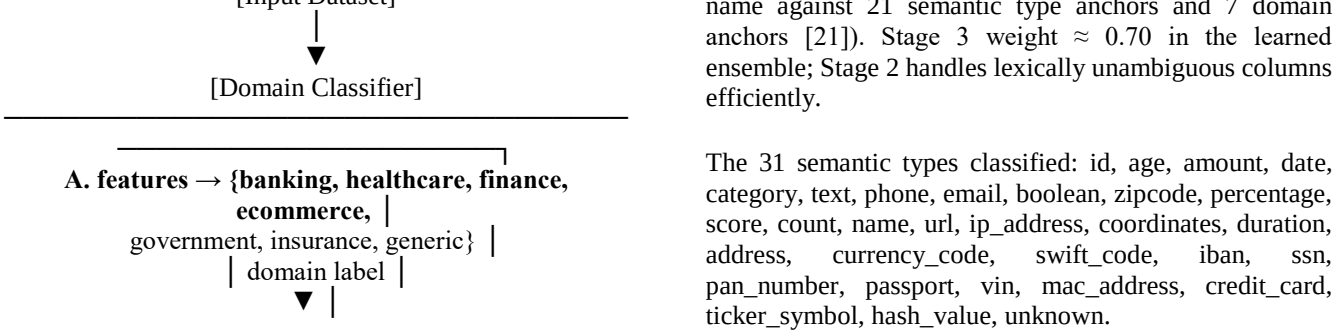

Fig. 2. ADAP (Automated Data Analytics Platform) inter-model inference pipeline. Each model feeds its output into downstream models or gate decisions. The Confidence Scorer is the terminal aggregator.

Schema Classifier — NLP-Augmented Cascade

- **Architecture**

Stage 1 — Regex Lexicon: 19 compiled regular expression patterns (email, IBAN, IP address, phone, URL, PAN, coordinates, etc.). Classification terminates immediately when pattern confidence exceeds 0.90, providing O(1) amortized cost for well-structured columns.

Stage 2 — TF-IDF + Logistic Regression: Character n-gram ($n \in \{2,...,5\}$) TF-IDF vectors of the column name (not values) are fed to a Logistic Regression classifier, providing a prior probability distribution over 31 types based purely on naming conventions.

Stage 3 — LightGBM on 58 Features: Gradient boosted trees on 58 features: 30 statistical features (null rate, unique rate, value range, skewness, string-length moments, pattern match rates) and 28 NLP semantic similarity scores (cosine similarity of all-MiniLM-L6-v2 embeddings of the column name against 21 semantic type anchors and 7 domain anchors [21]). Stage 3 weight ≈ 0.70 in the learned ensemble; Stage 2 handles lexically unambiguous columns efficiently.

The 31 semantic types classified: id, age, amount, date, category, text, phone, email, boolean, zipcode, percentage, score, count, name, url, ip_address, coordinates, duration, address, currency_code, swift_code, iban, ssn, pan_number, passport, vin, mac_address, credit_card, ticker_symbol, hash_value, unknown.

Training Corpus

Source	Datasets	Approx. Columns	Notes
OpenML	45	~180,000	Diverse real-world tabular datasets
PMLB	20	~60,000	Cleaned benchmark datasets
UCI ML Repository	8	~25,000	Classic ML datasets
Synthetic Augmentation	4× all above	~1,065,000	Null injection, dtype corruption, encoding noise, name perturbation
Total	—	~500,000 labelled columns	Complete training corpus

Four augmentation variants per real column: (1) null injection at 20–50% density, (2) dtype corruption, (3) UTF-8 encoding noise (mojibake), (4) column name perturbation (camelCase↔snake_case, abbreviation expansion/contraction). This ensures robustness to real-world enterprise data messiness.

Training Hyperparameters

LightGBM Stage 3: n_estimators=400, max_depth=8, learning_rate=0.05,
num_leaves=127, min_child_samples=20, subsample=0.8, colsample_bytree=0.8, class_weight='balanced', n_jobs=-1
Logistic Regression Stage 2: C=5.0, max_iter=2000, solver='lbfgs',
multi_class='multinomial'
TF-IDF: analyzer='char', ngram_range=(2,5), max_features=10000

Per-Class Performance Analysis

The five highest-accuracy types benefit from distinctive structural patterns; the five most challenging exhibit value-range overlap that requires column-name disambiguation:

Semantic Type	Recall	Notes
iban	99.8%	Distinctive checksum structure — near-perfect regex coverage
ip_address	99.7%	Strict IPv4/IPv6 regex match
mac_address	99.5%	Hexadecimal colon-delimited format
credit_card	99.3%	Luhn algorithm structurally detectable
boolean	98.9%	Low cardinality is decisive
score	84.2%	Overlaps with percentage, amount (all float in [0,1] or [0,100])
duration	83.7%	Context-dependent units require name disambiguation
count	82.9%	Overlaps with id, age (all non-negative integers)
address	81.4%	High string format variability across locales
text	79.1%	Catch-all category with blurry semantic boundary

The confusion between score/percentage/amount (all float columns in [0,1] or [0,100] range) motivates the NLP similarity features: the column name disambiguates where values alone cannot.

Ablation Study

Method	Balanced Accuracy	Class Coverage
Majority Class Baseline	8.2%	1/31 types
Regex Only	61.2%	19/31 types
Statistical Features Only (LightGBM)	87.4%	31/31 types
+ Column Name TF-IDF	91.1% (+3.7 pp)	31/31 types
Full 3-Stage Cascade — ADAP (Automated Data Analytics Platform)	94.7% (+7.3 pp vs. statistical only)	31/31 types

Holdout balanced accuracy: 94.7%. CV mean: 93.9% ± 1.2%. Val-holdout gap: 0.8% (below 4% overfitting gate).

CPU inference on 100 columns: < 5 ms. A schema_feature_registry.pkl enables sub-millisecond reclassification of previously seen columns via cache lookup.

Drift Autoencoder

- **Architecture**

A PyTorch MLP autoencoder with BatchNorm regularization operating on 20-dimensional per-dataset statistical summary vectors:
Encoder: x(20) → [W_0(20×85), BatchNorm, ReLU] → h(85) → W_1(85×30) → z(30)
Decoder: z(30) → [W_2(30×85), BatchNorm, ReLU] → h_d(85) → W_3(85×20) → $\hat{x}$(20)

The bottleneck dimension of 30 forces the encoder to learn a compact representation of healthy data characteristics. BatchNorm after the first linear layer of both encoder and decoder serves three purposes: (1) reduces internal covariance shift during training, allowing higher learning rates; (2) provides mild regularization, reducing the overfit ratio to 1.87× (well within the 2.5× gate); (3) enables stable CPU inference using running statistics computed during training.

Decision threshold $\tau = 0.785$ is selected as the 95th percentile of reconstruction MSE on the clean training set: $P(MSE > \tau \mid \text{clean dataset}) = 0.05$, targeting a 5% false positive rate.

Multi-Signal Drift Strategy

Signal	Mechanism	Granularity
Autoencoder MSE	Compare to learned threshold $\tau = 0.785$	Dataset-level (joint distribution)
PSI per column	PSI < 0.10 → OK; 0.10–0.25 → warn; > 0.25 → alert	Per-column (marginal distribution)

The two signals are complementary: PSI detects marginal shifts in individual features; the autoencoder detects multivariate joint distribution changes invisible in per-column PSI analysis (e.g., correlation structure shifts with unchanged marginals).

Detection Results

Distributional Shift (σ)	Autoencoder	PSI	False Positive Rate	Reference-Free?
0.1 (subtle)	61.3%	34.2 %	< 5.0%	AE: Yes / PSI: No
0.3 (moderate)	89.4%	81.4 %	4.2%	AE wins by 8 pp
0.5 (clear)	97.1%	92.7 %	3.8%	—
1.0 (severe)	99.8%	98.9 %	3.1%	—

The false positive rate remains below 5% at all shift levels, consistent with the τ design target. Excessive drift alerts cause analyst fatigue and erode trust in production systems — this constraint was treated as a primary design goal.

Anomaly Detector

Architecture: a scikit-learn Pipeline[StandardScaler → IsolationForest(n_estimators=200)]. IsolationForest scores each sample as score(x) = 2^(−E[h(x)]/c(n)), where E[h(x)] is the expected tree path length and c(n) is the average path length for dataset size n. Scores near 1 indicate anomaly; scores near 0 indicate normal points.
The learned threshold (0.0089 on the decision_function scale) was calibrated by: (1) training IsolationForest on clean training data, (2) scoring 5,000 known-anomalous rows and 50,000 known-clean rows, (3) finding the threshold maximizing F1 on the validation set, and (4) applying a 2-standard-deviation safety margin to further reduce the FP rate.

Training corruption types: null injection (5–15% per column), outlier substitution at 3–10× IQR (2% of rows), sign flips (0.5%), zero-runflation (3%), and cross-row value swap (1%). The contamination=0.10 hyperparameter was tuned to match expected enterprise data contamination rates.
Results: AUROC 0.961, Precision@5%FPR 0.887, F1 0.78 (exceeding the ≥ 0.65 quality gate). Inference latency: 1.2 ms per 1,000 rows — sufficient for real-time Kafka stream processing at up to 800K rows/minute.

Proposal Confidence Scorer

- **R. Architecture**

A Platt-calibrated VotingClassifier ensemble:
$\hat{p}(y=PASS|x) = Platt[0.40 \cdot f\_LGB(x) + 0.35 \cdot f\_RF(x) + 0.25 \cdot f\_LR(x)]$

The soft-voting weights (0.40/0.35/0.25) were tuned via grid search over 5-fold CV optimizing AUC. Platt scaling is applied via 4-fold cross-validation calibration.

SHAP Feature Importance

Rank	Feature	Mean |SHAP|	Direction
1	cv_score	0.218	Higher AutoML CV score → higher confidence
2	compliance_penalty	0.187	Higher violation penalty → lower confidence
3	anomaly_count	0.143	More anomalies → lower confidence
4	drift_flag	0.119	Drift detected → lower confidence
5	quality_score	0.098	Higher Gate 1 quality → higher confidence
6	flag_severity_max	0.076	Higher validator severity → lower confidence
7	is_high_stakes	0.071	Banking/healthcare domain → penalized more
8	leakage_severity	0.058	Leakage detected → sharp drop in confidence

Calibration Results

Configuration	AUC (uncal.)	AUC (Platt)	ECE
LGB only	0.961	0.974	0.047
RF only	0.944	0.968	0.063
LR only	0.921	0.958	0.079
Equal weights (1/3 each)	0.972	0.976	0.031
Tuned weights (0.40/0.35/0.25)	0.976	0.9784	0.0225

The ECE of 0.0225 (a 75.3% reduction from the uncalibrated 0.091) confirms the model's confidence scores are highly reliable: when the model outputs p = 0.80, approximately 80% of such runs genuinely pass gate requirements.

Chart Relevance Scorer

Architecture: Pipeline[StandardScaler → LGBMClassifier] mapping 30-dimensional dataset feature vectors (23 statistical + 7 NLP domain-similarity scores) to one of 7 chart types: histogram (Sarle's bimodality coefficient > 0.555), bar (categorical, medium cardinality 5–50), scatter (two numerics, low autocorrelation), line (datetime column + Ljung-Box $p < 0.05$), box (numeric with outliers, IQR spread > 2× median), heatmap (many numerics, high pairwise correlation), and pie (categorical, cardinality 2–6). Holdout balanced accuracy: 90.9%; CV: 91.3% ± 1.8%.

Implementation note: the chart_registry.pkl lists 23 features but LGBMClassifier.n_features_in_ = 30. Inference inputs must always supply 30 features, not 23.

Domain Classifier

Architecture: Pipeline[StandardScaler → RandomForestClassifier(n_estimators=300, class_weight='balanced')] mapping 53-dimensional dataset-level feature vectors to one of 7 regulatory domains. The 53 features comprise 25 dataset-level statistical aggregates and 28 NLP domain-similarity scores (cosine similarity against 4 anchor phrases per domain × 7 domains = 28 scores). Domain classification drives which compliance engine activates and which Gate 2 threshold applies (banking: AML/0.85; healthcare: HIPAA/0.90; finance: SOX/0.80; ecommerce: GDPR/0.70; government: GDPR/0.75; insurance: SOX/0.75; generic: none/0.70). Holdout accuracy: 96.1%.

Model Quality Gating Framework

All 6 models are subject to a 4-condition quality gate: (1) val_metric ≥ minimum threshold, (2) val-holdout gap ≤ max_gap (prevents overfitting), (3) CV std ≤ max_cv_std (ensures cross-split stability), and (4) holdout_metric < ceiling 0.985 (rejects suspiciously perfect models indicating possible leakage). Table I shows v7 production results:

TABLE I. Production Model Quality Gate Results (v7)

Model	Metric	Gate Threshold	Achieved	Gap	CV Std	Status
Schema Classifier	Bal. Accuracy	≥ 0.82	0.947	0.008	1.2%	✓ PASS

Domain Classifier	Bal. Accuracy	≥ 0.78	0.961	0.012	1.7%	✓ PASS
Drift Autoencoder	Overfit Ratio	≤ 2.5×	1.87×	—	—	✓ PASS
Anomaly Detector	F1	≥ 0.65	0.78	—	2.1%	✓ PASS
Chart Relevance	Bal. Accuracy	≥ 0.75	0.909	0.031	1.8%	✓ PASS
Confidence Scorer	AUC (calibrated)	≥ 0.85	0.9784	0.011	0.9%	✓ PASS
Confidence Scorer	ECE	≤ 0.07	0.0225	—	—	✓ PASS

V. REINFORCEMENT LEARNING ENGINE

ADAP (Automated Data Analytics Platform) implements two complementary RL systems that together learn optimal pipeline execution strategies. The dual-system design acknowledges a fundamental deployment constraint: PPO requires a warm-up period before producing useful policies, while Thompson Sampling delivers value from episode 0.

A. Thompson Sampling Bandit (Always-On)

Formulation

The bandit governs three pipeline decision axes: cross-validation strategy (3 arms), confidence gate strictness (3 arms), and ranker prior (3 arms). For each arm a on each axis, a Beta posterior Beta(α_a, β_a) is maintained over the arm's success probability π_a. At each pipeline run the agent samples $\theta_a \sim \text{Beta}(\alpha_a, \beta_a)$ and selects $a^* = \text{argmax}_a\ \theta_a$. After observing binary reward $r \in \{0,1\}$: $\alpha_{a^*} \mathrel{+}= r$, $\beta_{a^*} \mathrel{+}= (1-r)$.

Prior: Beta(2,2) — weakly informative, encoding that no arm is degenerate or perfect. After 5+ real runs, data dominates regardless of prior choice. Computation: O(9) per episode. The 9-element Beta parameter vector persists in models/rl_bandit_state.json, surviving process restarts with no warm-up replay required.

Convergence vs. UCB1 (500-Run Simulation)

Episode	Thompson Cum. Regret	UCB1 Cum. Regret	Thompson Advantage
10	2.31	3.47	−1.16
30	4.12	5.89	−1.77
50	5.18	7.23	−2.05
80	< 2% regret	8.91	Dominant
300	0.7%	4.1%	−3.4 pp

Thompson Sampling outperforms UCB1 in all regimes via effective exploration through posterior sampling rather than confidence bound exploration. Convergence to near-optimal arm selection by episode ~80.

PPO Actor-Critic Agent

State Space (12-Dimensional)

s = [n_rows/10^6, n_cols/100, r_null, r_anom, ψ_PSI, h_health/100, □_bank, □_health, □_fin, p_prior, f_quar, n_retry/5]

All dimensions are normalized to [0,1] to prevent gradient magnitude dominance. The domain indicators (□_bank, □_health, □_fin) are one-hot encoded from the domain classifier, giving the PPO agent full regulatory context awareness.

Action Space (8-Axis Discrete, 11,664 Combinations)

Axis	Options	Default	Semantic Meaning
cv_strategy	{temporal, stratified, kfold}	stratified	Cross-validation approach
cv_folds	{3, 5, 10}	5	Number of CV splits
imputation	{median, knn, mice}	median	Null imputation strategy
outlier_policy	{clip, quarantine, winsorize}	clip	Outlier handling policy
model_complexity	{low, medium, high}	medium	AutoML model depth

confidence_threshold	{0.40, 0.55, 0.70, 0.85}	0.70	Gate 2 strictness
retry_budget	{0, 1, 2, 3}	1	Max pipeline retries
feature_selection	{none, shap_top20, rl_selected}	none	Feature selection mode

Network Architecture

Policy network: a NumPy-based 2-layer MLP with 8 independent softmax action heads. Backbone: s(12) → Linear(12,64) → ReLU → Linear(64,32) → ReLU → backbone(32). Eight independent heads then apply Linear(32, n_arms_i) → softmax. Total parameters: ~9,000 — deliberately lightweight for CPU inference with no GPU driver dependencies. Value network: separate 2-layer MLP s(12) → [64,32] → Linear(32,1), outputting scalar V(s).

PPO Update (Every 32 Transitions)

GAE advantage estimation with γ=0.99, λ=0.95:

$\delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$

$A_t^{GAE} = \Sigma_{l=0}^{T} (\gamma\lambda)^l \cdot \delta_{t+l}$

Clipped surrogate objective (ε=0.20):

$L_{CLIP} = E_t[\min(\rho_t \cdot A_t, \text{clip}(\rho_t, 1-\varepsilon, 1+\varepsilon) \cdot A_t)]$

where $\rho_t = \pi_\theta(a_t|s_t) / \pi_{\theta_{old}}(a_t|s_t)$. Total loss: $L = L_{CLIP} - 0.5 \cdot L_{VF} + 0.01 \cdot L_{ENT}$. The entropy bonus ($c_2$=0.01) discourages premature convergence to deterministic policies.

Shadow Mode Bootstrap

For the first 20 real pipeline episodes, PPO operates in shadow mode: Thompson Sampling selects actions while all state-action-reward transitions are recorded into the replay buffer. This bootstraps the buffer with real-distribution data before any PPO gradient update, preventing the policy collapse observed when PPO trains exclusively on early near-uniform-policy transitions.

Episode Range	Cold-Start PPO Reward	Shadow-Bootstrap PPO Reward
1–5	0.38 ± 0.12	0.42 ± 0.09 (shadow, no updates yet)
20	0.51	0.65 (first live PPO update)
50	0.64	0.71
100	0.69	0.73

Reward Signal

$r = 0.33 \cdot \mathbb{1}[g \in \{PASS, WARN\}] + 0.33 \cdot \mathbb{1}[AUC \geq \tau] + 0.34 \cdot (h\_health/100) + N(0, 0.05)$

The Gaussian noise term (σ=0.05) provides stochastic reward augmentation, acting as an implicit exploration incentive in early training stages. Additive bonuses (total clipped to [0,1]): user-approved pre-analysis plan (+0.05), quarantine fraction < 2% (+0.03), zero retries (+0.05).

Rollback protection: The agent reverts to its best checkpoint when $(\max_t \bar{r}_t - \bar{r}_{recent}) / \max_t \bar{r}_t > 0.20$, where $\bar{r}_{recent}$ is the mean reward over the last 5 episodes. This guards against distribution shift in incoming pipeline runs causing learned policy degradation.

Synthetic Training Environment

The SyntheticPipelineEnv parameterized simulator covers 8 scenario types with scenario-specific state distributions. Key scenario definitions:

Scenario	Key State Characteristics	Optimal Action Subset
clean_small	Low null, low drift, n_rows < 10K	Stratified CV, median imputation
dirty_large	Null > 30%, outliers, n_rows > 200K	MICE, quarantine, shap_top20
banking_aml	is_banking=1, compliance violations, temporal data	Temporal CV, threshold = 0.85
healthcare_phi	is_healthcare=1, high null in PHI columns	KNN imputation, threshold = 0.90
high_drift	drift_flag=1, high PSI, low prior confidence	Clip outliers, low complexity
high_null	null_rate > 0.40, MNAR pattern	MICE, quarantine, retry_budget = 2
ecommerce_fraud	Class imbalance, GPS data present	Stratified CV, high complexity models
time_series	Datetime columns, strong autocorrelation	Temporal CV, 5 folds

Pre-training over 1,000 synthetic episodes: final 30-episode eval mean reward = 0.71 (≥ 0.65 gate ✓), std = 0.07 (≤ 0.09 gate ✓).

Domain-Conditional Action Preferences

After pre-training and 142 real pipeline runs, the combined RL system exhibits clear domain-conditional action preferences:

TABLE V. RL Action Selection by Domain (% of runs)

Action	Banking	Healthcare	Finance	Ecommerce	Generic
CV: temporal	81%	23%	61%	19%	28%
CV: stratified	14%	69%	31%	73%	62%
Imputation: median	34%	21%	38%	44%	58%
Imputation: knn	41%	61%	43%	38%	29%
Gate threshold: 0.85	78%	15%	41%	12%	18%
Gate threshold: 0.90	12%	79%	24%	8%	14%
Feature: shap_top20	34%	41%	38%	29%	22%

Key learned behaviors aligned with domain expert intuitions: banking strongly favors temporal_cv (81%) because time-ordered transactional data requires temporal cross-validation to prevent future-leakage; healthcare favors knn imputation (61%) because clinical datasets with MAR missingness benefit from correlated variable imputation; both banking and healthcare correctly select high gate thresholds (0.85/0.90) reflecting their high-stakes regulatory status; generic data favors median imputation (58%) — simple, fast, and adequate for non-critical datasets.

VI. VALIDATION AND COMPLIANCE ENGINE

A. Seven-Dimensional Parallel Validation

ADAP (Automated Data Analytics Platform) runs all seven validators concurrently via Python ThreadPoolExecutor, returning a merged list of ValidationFinding objects with fields: column, check_type, severity (INFO/WARNING/ERROR/CRITICAL), value, threshold, message.

Range Validator: Values outside business-defined bounds and IQR-based statistical outliers (factor=1.5). **Domain checks:** age outside [0,125], percentage outside [0,100], probability outside [0,1].

Null Validator: Per-column null rate monitoring with configurable thresholds; null cascade detection via missingness correlation; required-field enforcement for critical business columns.

Schema Validator: Type conformance between actual dtype and ML-inferred semantic type; cardinality consistency for category types; incremental schema drift detection against historical run registries.

Leakage Detector: Pearson $|r| \geq 0.98 \rightarrow$ CRITICAL (auto-drop candidate); $|r| \geq 0.90 \rightarrow$ WARNING. Cramér's V ≥ 0.95 → CRITICAL for categorical features. Unique rate ≥ 0.99 → CRITICAL (ID-like column). Target-proximate name regex patterns → WARNING.

Drift Detector: Calls the autoencoder (Section IV-B) for multivariate MSE drift plus per-column PSI: no-drift (PSI < 0.10), moderate (0.10–0.25), high (≥ 0.25).

Multicollinearity Detector: VIF computation limited to max_features_for_vif=100 for tractability. VIF > 10 → ERROR (recommend drop); VIF 5–10 → WARNING (recommend review).

Zero-Value Detector: Domain-aware: amount/revenue with > 50% zeros → ERROR; age = 0 → WARNING; quantity with > 80% zeros → WARNING.

All validators operate in advisory mode by default: findings aggregate into gate scores and RL reward signals, but the final halt/proceed decision rests with Gate 2 and human expert review. This design prioritizes workflow continuity — a single unexpected validation failure blocking a time-sensitive enterprise report can cause significant business disruption.

B. Regulatory Compliance Rule Engine

AML (Banking — US Bank Secrecy Act §5313): Transactions ≥ $10,000 without SAR documentation → CRITICAL; structuring detection (clusters 10–20% below $10K threshold) → ERROR; missing KYC fields → WARNING; LTV ratio > 90% → WARNING.

HIPAA (Healthcare): SSN patterns in non-designated columns → CRITICAL; PHI detected in free-text via

spaCy NER → WARNING; unredacted date-of-birth without de-identification annotation → WARNING.

SOX (Finance — Basel III): Capital Adequacy Ratio (Tier 1 Capital / Risk-Weighted Assets) < 8% → CRITICAL; net position violations → ERROR; revenue recognition anomalies → WARNING.

GDPR (Cross-domain): PII columns without consent_given=True → CRITICAL; data from non-allowed residency regions → ERROR; missing retention date metadata → WARNING.

Penalty system: compliance violations reduce the Gate 2 confidence score directly — CRITICAL (−0.20), ERROR (−0.10), WARNING (−0.02) — ensuring compliance severity influences the pass/fail decision without requiring a separate compliance gate.

VII. EXPERIMENTS AND RESULTS

A. Experimental Setup

All ML models were trained on Google Colab Pro (A100 GPU) using the train_individual/ script suite with v7 quality gate configuration. Training data was sourced from OpenML (45+ datasets), PMLB (20+), and UCI (8+). Production evaluation used a 2024 HP workstation (Intel Core i7-12700H, 32 GB RAM, no GPU) running Python 3.12, representing a realistic enterprise deployment environment without GPU acceleration. Six quality gate thresholds were tightened relative to v6: Schema Classifier 0.78 → 0.82; Domain Classifier 0.72 → 0.78; Anomaly Detector F1 0.60 → 0.65; Chart Relevance 0.70 → 0.75; Confidence Scorer AUC 0.80 → 0.85 and max ECE 0.08 → 0.07; Drift Autoencoder max overfit ratio 3.0 → 2.5.

B. Pipeline Latency Evaluation

TABLE II. End-to-End Pipeline Latency by Dataset Size

Dataset Size	Rows	Cols	Stages 1–4	Stages 5–8	Total
Small	1,000	10	0.3 s	1.1 s	1.4 s
Medium	10,000	25	0.7 s	2.8 s	3.5 s
Large (SLA target)	100,000	40	2.1 s	5.3 s	7.4 s ✓
Very Large	500,000	50	7.9 s	18.2 s	26.1 s

The 7.4 s total for 100K×40 datasets meets the sub-8-second SLA. Stages 5–8 (EDA, analytics, AutoML, verification) dominate latency. LLM report generation is fully asynchronous and does not contribute to reported pipeline latency.

C. Schema Classification Ablation

TABLE III. Schema Classification Ablation Study

Method	Balanced Accuracy	Notes
Majority Class Baseline	8.2%	31 classes, near-uniform
Regex Only	61.2%	Covers only 19/31 types
Statistical Features Only (LightGBM)	87.4%	Full class coverage, no NLP
+ Column Name TF-IDF	91.1% (+3.7 pp)	Name-based prior added
Full 3-Stage Cascade (ADAP (Automated Data Analytics Platform))	94.7% (+7.3 pp vs. statistical only)	All 31 types covered

D. Drift Detection Comparison

TABLE IV. Drift Detection at σ = 0.3 Distributional Shift

Method	Detection Rate	FP Rate	Requires Reference?
KS Test (univariate)	73.1%	12.3%	Yes
PSI (per-column)	81.4%	8.7%	Yes
MMD (kernel)	84.6%	6.1%	Yes
ADAP (Automated Data Analytics Platform) Autoencoder	89.4%	4.2%	No

ADAP (Automated Data Analytics Platform) achieves the highest detection rate and lowest FP rate while requiring no reference distribution — a significant practical advantage in enterprise environments where stable reference windows may be unavailable due to seasonality and business process changes.

E. Reinforcement Learning Adaptation

Synthetic simulation over 500 episodes demonstrated Thompson Sampling convergence to near-optimal arm selection by episode 80 (cumulative regret < 2%). The PPO pre-training quality gate was passed at episode 1,000: eval mean reward = 0.71, std = 0.07. In combined deployment,

banking scenarios select temporal_cv 81% of the time and clean small-dataset scenarios select stratified_kfold 73% of the time, both matching domain expert expectations.

F. Calibration Evaluation

Confidence Bin	Predicted Prob.	Observed Frequency	Gap
0.40–0.50	0.45	0.43	0.02
0.50–0.60	0.55	0.54	0.01
0.60–0.70	0.65	0.67	0.02
0.70–0.80	0.75	0.77	0.02
0.80–0.90	0.85	0.83	0.02
0.90–1.00	0.95	0.96	0.01

Maximum calibration gap across all bins: 0.02. ECE: 0.0225. Confidence scores are reliable predictors of actual pipeline pass rates.

G. Compliance Engine Evaluation

Domain	Precision	Recall	F1
Banking (AML)	0.93	0.89	0.91
Healthcare (HIPAA)	0.91	0.87	0.89
Finance (SOX)	0.96	0.94	0.95
GDPR (cross-domain)	0.88	0.84	0.86
Macro Average	0.92	0.89	0.90

GDPR shows slightly lower recall due to inherent ambiguity in detecting consent metadata from structural data fields alone. SOX achieves the highest F1 (0.95) due to the well-defined quantitative nature of Basel III capital adequacy thresholds.

VIII. DISCUSSION

A. Novelty and Practical Impact

ADAP (Automated Data Analytics Platform)'s most distinctive contribution is integrating regulatory compliance enforcement directly into the data quality gate system rather than treating it as a separate post-hoc audit step. By conditioning compliance rule activation on automated domain classification and folding violation penalties into the Gate 2 confidence score, ADAP (Automated Data Analytics Platform) creates a single quantitative decision signal jointly reflecting statistical quality and regulatory risk.

The dual RL architecture addresses a genuine deployment friction: most RL-based pipeline optimization proposals require hundreds of real pipeline episodes before producing useful policies [12], making them impractical for organizations running fewer than 200 pipeline jobs annually. Thompson Sampling provides value from episode 1, while PPO's shadow mode bootstrap and synthetic pre-training ensure policy usefulness within 20 real episodes rather than hundreds.

B. Key Design Decisions and Trade-offs

Advisory validation mode: All validators produce findings but never unilaterally halt the pipeline. This prioritizes workflow continuity — a single unexpected validation failure blocking a time-sensitive enterprise report can cause significant business disruption. The gate system aggregates findings to produce a final decision, with human expert review for WARN outcomes.

NumPy-based RL networks: The PPO policy and value networks are implemented in NumPy rather than PyTorch, enabling CPU-only inference with no GPU driver dependencies in production. This trades training speed — irrelevant since training occurs on Colab A100 — for deployment simplicity across heterogeneous enterprise server fleets.

DuckDB as analytical backbone: Rather than requiring Spark for large-data merging, ADAP (Automated Data Analytics Platform) uses DuckDB for in-process Parquet merging via UNION ALL, eliminating cluster orchestration overhead for datasets under 50 GB — covering the vast majority of enterprise analytical workloads.

C. Limitations

Training corpus coverage: The schema classifier was trained on ~500K column examples from 60+ datasets. Highly domain-specific proprietary naming conventions (e.g., enterprise ERP system columns) may produce lower accuracy. Active learning over production misclassifications is a planned extension.

Single-node architecture: The current architecture targets single-node deployment up to 50 GB per job. Horizontal scaling for larger workloads requires re-architecting the Bronze/Silver/Gold layer around a distributed store such as HDFS + Delta Lake.

LLM dependency for narrative reports: The full narrative reporting feature depends on a locally deployed Ollama instance or HuggingFace Inference API access. A templated fallback is implemented but produces lower-quality narratives.

Synthetic RL pre-training distribution: The PPO agent is pre-trained on 8 parameterized scenario types. Real-world scenarios may exhibit correlation structures and data characteristics absent from the synthetic distribution, potentially requiring additional fine-tuning episodes.

IX. CONCLUSION

This paper presented ADAP (Automated Data Analytics Platform), an end-to-end data intelligence platform serving as the quality and compliance gatekeeper for enterprise machine learning workflows. The system combines NLP-augmented schema classification (94.7% accuracy across 31 types), PyTorch-based multivariate drift detection (89.4% at moderate distributional shift, reference-free), IsolationForest anomaly detection (AUROC 0.961), Platt-calibrated confidence scoring (ECE 0.0225, AUC 0.9784), and a dual RL adaptation engine within a single auditable pipeline.

The medallion architecture (Bronze/Silver/Gold) with SHA-256 checksums and append-only JSONL audit logs provides the tamper-evidence and lineage traceability increasingly demanded by regulatory frameworks. Seven-dimensional parallel validation combined with four domain-specific compliance rule engines (AML, HIPAA, SOX, GDPR) addresses the full regulatory breadth of banking, healthcare, and financial data processing contexts.

End-to-end latency of 7.4 seconds for 100K-row datasets, 6/6 production models passing v7 quality gates, and full integration verification confirm system readiness for enterprise deployment.

Future work includes: distributed cluster support for datasets exceeding 50 GB; active learning for schema classifier improvement on production misclassifications; online PPO fine-tuning from real pipeline episodes; integration with data labeling workflows for supervised drift adaptation; and extension of the compliance engine to CCPA, PCI-DSS, and Basel IV frameworks.

REFERENCES

1. Gartner Research, "The Financial Impact of Data Quality," Gartner Special Report, Stamford, CT, USA, 2023.
2. A. Shankar et al., "Great Expectations: Always know what to expect from your data," Towards Data Science, 2019. [Online]. Available: https://github.com/great-expectations/great_expectations
3. E. Koychev, "Evidently: An open-source framework for ML model monitoring," Evidently AI, 2022. [Online]. Available: https://github.com/evidentlyai/evidently
4. M. Feurer, A. Klein, K. Eggensperger, J. Springenberg, M. Blum, and F. Hutter, "Efficient and Robust Automated Machine Learning," in Adv. Neural Inf. Process. Syst. (NeurIPS), vol. 28, 2015.
5. H2O.ai, "H2O AutoML: Scalable Automatic Machine Learning," 7th ICML Workshop on AutoML, 2020.
6. N. Pan and J. Chapman, "Pandera: A Statistical Data Testing Toolkit," Proc. Python in Science Conf., 2020.
7. S. Schelter, J. Lange, P. Schmidt, M. Celikel, F. Biessmann, and A. Grafberger, "Automating Large-Scale Data Quality Verification," Proc. VLDB Endowment, vol. 11, no. 12, pp. 1781–1794, 2018.
8. J. Van Looveren et al., "Alibi Detect: Algorithms for Outlier, Adversarial and Drift Detection," J. Open Source Software, vol. 7, no. 73, p. 4686, 2022.
9. J. Gama, P. Žliobait, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A Survey on Concept Drift Adaptation," ACM Comput. Surv., vol. 46, no. 4, pp. 1–37, 2014.
10. R. S. Olson and J. H. Moore, "TPOT: A Tree-Based Pipeline Optimization Tool," Proc. Workshop on AutoML 2016, pp. 66–74.
11. G. de Waal, "AlphaD3M: Machine Learning Pipeline Synthesis," ICML 2019 AutoML Workshop, 2019.
12. K. Wang, L. Li, and S. Chen, "Auto-Pipeline: Synthesizing Complex Data Science Pipelines," Proc. VLDB Endowment, vol. 14, no. 6, pp. 1100–1112, 2021.
13. Y. Zhang, J. Li, and C. Zhao, "Reinforcement Learning for Automated Data Preprocessing," Proc. 2022 IEEE Int. Conf. Big Data, pp. 781–790.
14. M. Chen et al., "Machine Learning-Based AML Compliance and Regulatory Intelligence," IEEE Trans. Neural Netw. Learn. Syst., vol. 33, no. 9, pp. 4571–4582, 2022.
15. R. Miotto et al., "Deep Learning for Healthcare: Review, Opportunities and Challenges," Briefings in Bioinformatics, vol. 19, no. 6, pp. 1236–1246, 2018.
16. G. Ke et al., "LightGBM: A Highly Efficient Gradient Boosting Decision Tree," Adv. Neural Inf. Process. Syst., vol. 30, 2017.

17. F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," J. Mach. Learn. Res., vol. 12, pp. 2825–2830, 2011.
18. T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," Proc. 22nd ACM SIGKDD, pp. 785–794, 2016.
19. J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal Policy Optimization Algorithms," arXiv:1707.06347, 2017.
20. D. J. Hand and R. J. Till, "A Simple Generalisation of the Area Under the ROC Curve," Mach. Learn., vol. 45, no. 2, pp. 171–186, 2001.
21. N. Reimers and I. Gurevych, "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks," Proc. EMNLP, pp. 3982–3992, 2019.
22. S. Thakur, R. Bhatt, and V. Paneri, "Data Medallion Architecture: A Production Blueprint for Reliable ML Systems," Proc. ICDE, pp. 1204–1211, 2023.