Journal of Computer Science and Technology Studies
ISSN: 2709-104X
DOI: 10.32996/jcsts
Journal Homepage: www.al-kindipublisher.com/index.php/jcsts

JCSTS
AL-KINDI CENTER FOR RESEARCH AND DEVELOPMENT

| RESEARCH ARTICLE

Avoiding SRE Anti-Patterns in AI Workloads: A Framework for Production-Ready Machine Learning Systems

Sreejith Kaimal
Principal SRE, C3.AI, USA
Corresponding Author: Sreejith Kaimal, **E-mail**: kaimalsreejiths@gmail.com

| ABSTRACT

The convergence of artificial intelligence workloads with site reliability engineering principles necessitates a fundamental transformation in how organizations conceptualize and implement production machine learning systems. This framework addresses the unique challenges emerging when traditional SRE practices encounter the probabilistic nature of ML systems, where failures manifest not as binary outages but as gradual performance degradation across multiple dimensions. The identification of critical anti-patterns reveals systematic vulnerabilities in production AI environments, including the pervasive tendency to prioritize deployment velocity over reliability, sophisticated monitoring systems that generate overwhelming noise without meaningful signal, and infrastructure-centric observability that tracks traditional metrics while missing model-specific degradation patterns. Through the establishment of dual-track error budgets, comprehensive Service Level Indicator frameworks bifurcated between infrastructure and model dimensions, and the emergence of Data SRE as a specialized discipline bridging statistical expertise with operational excellence, organizations can create sustainable AI operations. The framework emphasizes that success in production AI environments depends fundamentally on disciplined avoidance of well-documented anti-patterns through proactive reliability engineering, automated validation pipelines, and explainable AI integration for systematic debugging, transforming reactive incident management into predictive operational excellence.

| KEYWORDS

Machine Learning Operations, Site Reliability Engineering, Service Level Objectives, Model Drift Detection, Production Readiness Framework

| ARTICLE INFORMATION

ACCEPTED: 03 October 2025 **PUBLISHED:** 19 October 2025 **DOI:** 10.32996/jcsts.2025.7.10.49

1. Introduction: The Intersection of SRE and AI Operations

The increasing spread of artificial intelligence and machine learning-based systems within production has radically changed the environment of site reliability engineering, presenting unparalleled operational issues that conventional frameworks are unable to tackle. The classic work by Sculley et al. illustrates that ML systems incur technical debt at a much faster rate compared to traditional software, with but a minority of actual ML code present in real-world ML systems—the overwhelming majority is comprised of intricate surrounding infrastructure and data pipeline management [1]. This architectural fact requires a basic rethinking of reliability engineering practice, as companies learn that production ML system maintenance takes significantly more engineering time than model development.

The realization of specialized cross-domain roles shows the recognition in the industry that AI reliability needs people with expertise across multiple domains. While classical software systems require reliability metrics to address mostly availability and latency measurements, AI applications present multidimensional complexity in the form of model accuracy degradation, data drift effects, prediction confidence fluctuations, and cascading business impact metrics. Current research into technical debt

trends shows that ML systems have certain anti-patterns, such as erosion of boundaries between components, feature entanglement, and obscure feedback loops, that make maintenance a nightmare much worse than in conventional software stacks [1]. These issues arise especially intensely when data dependencies result in versioning problems, with changing anything that changes everything in the integrated ML pipeline system.

The complexity of operations escalates considering the fact that contemporary production environments instantiate hundreds or thousands of models in tandem, each mandating separate monitoring, retraining schedules, and performance tracking. Happer's study shows how code smells specific to ML implementations can be mitigated to cut down technical debt buildup by the use of automated refactoring tools uniquely tailored for AI systems, although uptake is still restricted by organizational reluctance and the absence of standard practices [2]. The research describes that active debt management using automated tooling avoids the build-up of maintenance overhead that otherwise eats up exponentially growing engineering resources as systems get larger.

This convergence has created new operating paradigms wherein principles developed in early SRE frameworks need substantial extension and reinterpretation to contend with the probabilistic nature of ML workloads. The systematic investigation of anti-patterns found in many production deployments offers key observations on failure modes unique to AI systems. Organisations adopting holistic frameworks that blend foundational SRE principles such as Error Budgets, Service Level Objectives, and strategies for eliminating toil with ML-specific factors like model monitoring protocols, data quality assurance mechanisms, and explainable AI debugging strategies show significantly better operational results [2]. The evidence indicates that production AI success tends to rely less on cutting-edge model designs and more on systematic prevention of widely known anti-patterns through disciplined operational practices and architectural choices that optimize for long-term maintainability rather than short-term deployment speed.

System Aspect	Finding/ Characteristic
Technical debt accumulation speed	Significantly faster than traditional software
Engineering effort ratio	Maintenance time tends to be greater than development time
Change propagation effect	Single change affects entire pipeline
Component boundary integrity	Erosion between system components
Feature dependency pattern	Entanglement is causing maintenance issues
Automated tool adoption rate	Low due to organizational resistance

Table 1: Technical Debt Accumulation in Machine Learning Systems [1,2]

2. Theoretical Foundation: Scaling SRE Principles to AI Workloads

Extending Site Reliability Engineering principles to artificial intelligence workloads requires a radical rethinking of common reliability metrics and operational practices. The Error Budget concept, central to SRE philosophy as articulated by Beyer et al., requires substantial adaptation when applied to machine learning systems where failures manifest not as binary service outages but as gradual performance degradation across multiple dimensions [4]. Whereas traditional services may provide error budgets strictly in terms of downtime percentages or request failure rates, AI systems require model accuracy degradation, prediction confidence fluctuation, and data quality degradation to be factored into reliability calculations, thereby establishing a multidimensional error budget model that extends across infrastructure as well as algorithmic performance fronts.

The Service Level Indicator, Service Level Objective, and Service Level Agreement trilogy is the quantitative foundation used to manage expectations of reliability in production. A Service Level Indicator (SLI) represents a carefully defined metric that measures a specific aspect of the service—in AI systems, this extends beyond traditional metrics to include prediction latency, prediction accuracy, and feature freshness. The Service Level Objective (SLO) establishes a target value or range for each SLI, creating measurable goals such as maintaining p99 prediction latency below 100ms or ensuring model accuracy exceeds 95%. Finally, the Service Level Agreement (SLA) formalizes these objectives into a contract with defined consequences if SLOs are not met, providing accountability and clear escalation paths when reliability targets are breached [4]. For AI workloads, this requires splitting into Infrastructure SLOs that include classic metrics like API response times at different percentiles, service availability as a percentage, and throughput capacity, in addition to Model SLOs, which include prediction accuracy rates, data freshness intervals, feature completeness ratios, and confidence score distributions.

Studies by Kiyak et al. illustrate how supervised learning models such as High-Level K-Nearest Neighbors perform best when certain operational thresholds are complied with, and classification accuracy suffered dramatically when computational resources were below established levels or where training data volumes were too low [3]. This result highlights the importance of mapping dual-track error budgets that understand interdependence between infrastructure stability and model performance, where violating either can cascade into system-wide reliability breakdowns. Organizations must establish Model SLOs derived from these SLIs that might mandate prediction accuracy >95%, data staleness <5 minutes, feature completeness >99%, and p50 confidence scores >0.8.

The law of toil elimination, well-documented in SRE literature as repetitive manual labor that increases linearly with service expansion, acquires special significance in ML operations where model deployment complexity introduces exponential operational load [4]. Configuration entropy expresses itself through mushrooming YAML files, off-book infrastructure-as-code modules, and divergent deployment patterns that introduce maintenance hassles far outpacing those in conventional software systems. The deployment of self-service platforms that have guardrails built in tackles this challenge head-on by automating manual chores while insisting on consistency via tested templates and pre-defined workflows.

The mathematical foundations of ML algorithms bring special reliability concerns that are not present in conventional software development. Classification models display performance fluctuations depending on shifts in data distribution, with Kiyak et al. observing that even slight statistical fluctuations in input features can lead to accuracy declines of more than 15% in production contexts [3]. This probabilistic character requires reliability structures to factor in statistical variation as a natural system attribute in lieu of holding all departures from the model as failures, with the need for nuanced error budget analysis distinguishing between allowable model uncertainty and true performance degradation in need of action.

3. Key Anti-Patterns in AI Production Systems

The study of production AI failures identifies a number of recurring anti-patterns that consistently erode system reliability and organizational pace. The "Ship First, Secure Later" anti-pattern is likely the most insidious danger to sustainable AI operations, emerging when competitive pressures supplant engineering discipline. Polyzotis et al. recognize data management as the leading point of failure in production ML systems, observing that weak data validation and monitoring infrastructure generate quiet failures that build up undetected for long periods before bringing about disastrous model degradation [5]. This method blatantly disobeys the Error Budget rule by sacrificing short-term deployment speed for long-term debt towards reliability, with teams often skipping checks of production readiness and rolling out models without setting adequate Service Level Indicators or Objectives.

The lack of defined SLIs for model performance creates a perilous operations blind spot where teams are unable to quantify degradation patterns. Without SLOs, teams cannot determine acceptable performance thresholds, and without SLAs, there are no consequences or accountability when models fail. A fraud detection model might maintain perfect infrastructure availability, meeting its Infrastructure SLO of 99.9% uptime, while experiencing a 20% false negative rate due to training data drift, violating an untracked Model SLO for accuracy [5]. Studies illustrate that ML production pipelines need end-to-end data quality monitoring across each transformation step, but organizations commonly only check final model outputs, skimping on essential intermediate failure points that cascade into system-wide reliability issues. Technical debt inevitably accumulates to completely drain error budgets, leaving organizations in emergency fix cycles that stop feature development and erode customer trust by way of lowered service quality.

The "NoiseOps" anti-pattern arises when companies roll out complex AIOps solutions without creating effective contextual frameworks or relevant performance metrics. Das points out that global TensorFlow deployments in complicated environments produce immense telemetry data volumes, but without domain-specific filtering and prioritization, the monitoring systems create cacophony-type alert storms that hide real incidents [6]. Advanced anomaly detection systems, while technically impressive, often generate hundreds of false positive alerts when deployed without model-aware tuning, creating alert fatigue that reduces team responsiveness to actual failures. This occurs because the system lacks properly defined Model SLIs—it monitors infrastructure metrics that are easy to measure rather than model-specific indicators that actually matter. The resolution requires incorporating feedback loops that connect model-specific SLIs (prediction confidence, feature drift indicators) with traditional infrastructure monitoring, creating SLOs that reflect genuine business impact rather than arbitrary technical thresholds.

The "Infrastructure-Centric Observability" anti-pattern indicates a basic misunderstanding of failure modes of AI systems and complete monitoring needs. Organizations often put in place meticulous Infrastructure SLIs monitoring CPU usage, memory usage, and network latency while being oblivious to Model SLIs measuring real predictive performance. Polyzotis et al. highlight that data dependencies in ML systems introduce latent failure modes wherein models have perfect infrastructure availability while suffering from disastrous accuracy because of upstream data quality issues [5]. Das explains that TensorFlow deployments

incur specialized monitoring, including tensor shape mismatches, detection of gradient explosion, and training convergence measures outside the domain of traditional infrastructure monitoring [6]. This gap in observability is a result of organizational silos between infrastructure teams who understand traditional SLIs and data science teams who understand model behavior but may not think in terms of SLIs/SLOs. The critical need for Data SREs who can bridge these domains becomes apparent—professionals who can establish holistic SLI frameworks that encompass both operational and statistical dimensions of system health, define meaningful SLOs that balance performance with resource constraints, and negotiate SLAs that align technical capabilities with business requirements.

Metric Component	Specification
p99 latency SLO	Below 100ms
Model accuracy SLO	Exceeds 95%
Data staleness threshold	Less than 5 minutes
Feature completeness SLO	Greater than 99%
p50 confidence scores	Greater than 0.8
Accuracy decline risk	More than 15%

Table 2: SLO/SLI Framework Components [3,4]

4. End-to-End Monitoring Framework for AI Systems

Establishing an end-to-end monitoring framework for AI systems mandates a drastic departure from legacy infrastructure-focused practices, starting with the right definition and execution of Model Service Level Indicators that take into account distinctive features of machine learning systems. Ogeti et al. point out that cloud-deployed ML models present specific observability challenges, such as network latency fluctuations, distributed system failures, and resource contention problems that existing monitoring frameworks are not able to cope with effectively [7]. Contemporary AI observability needs to set Service Level Indicators, including prediction accuracy percentages, feature freshness metrics, monitoring time since last update, data drift metrics computing statistical distance between real-time and training distributions, and prediction confidence score percentiles that together capture significant properties of model performance directly influencing business results. Each SLI must be carefully selected to represent a meaningful aspect of model performance that directly impacts business outcomes or user experience.

The translation of these SLIs into actionable Service Level Objectives requires deep collaboration between Data SREs, data scientists, and business stakeholders to balance competing priorities and resource limitations. While a data scientist might propose an SLO of 99% model accuracy, the Data SRE must balance this against computational costs and latency requirements, potentially settling on a more pragmatic SLO of 95% accuracy with p99 prediction latency under 100ms [7]. These SLOs then form the basis for SLAs—formal contracts that specify consequences when objectives are not met, such as customer credits, escalation procedures, or mandatory model retraining cycles. Evidence suggests that cloud-hosted ML deployments see performance fluctuation according to geographic distribution, with differences in latency across regions possibly leading to model timeout failure and compromised user experiences in case SLOs are left undefined or excessively optimistic.

Data drift monitoring is an essential Model SLI that calls for advanced statistical methods for detection and quantification. Mannapur offers an in-depth analysis of methods of detecting drift, noting that Kolmogorov-Smirnov tests efficiently detect distribution changes in continuous variables, but chi-square tests are more appropriate for categorical features, where Population Stability Index calculations give complete drift measurements across entire spaces of features [8]. When feature distributions shift beyond SLO thresholds—indicating the model is operating on data fundamentally different from its training set—automated alerts trigger investigation workflows. Concept drift, defined in distinction from data drift by changes in underlying feature-target variable relationships instead of just input distributional shifts, necessitates ongoing checks against ground truth labels or well-maintained canary datasets in order to catch incipient performance degradation before it becomes customer-impacting.

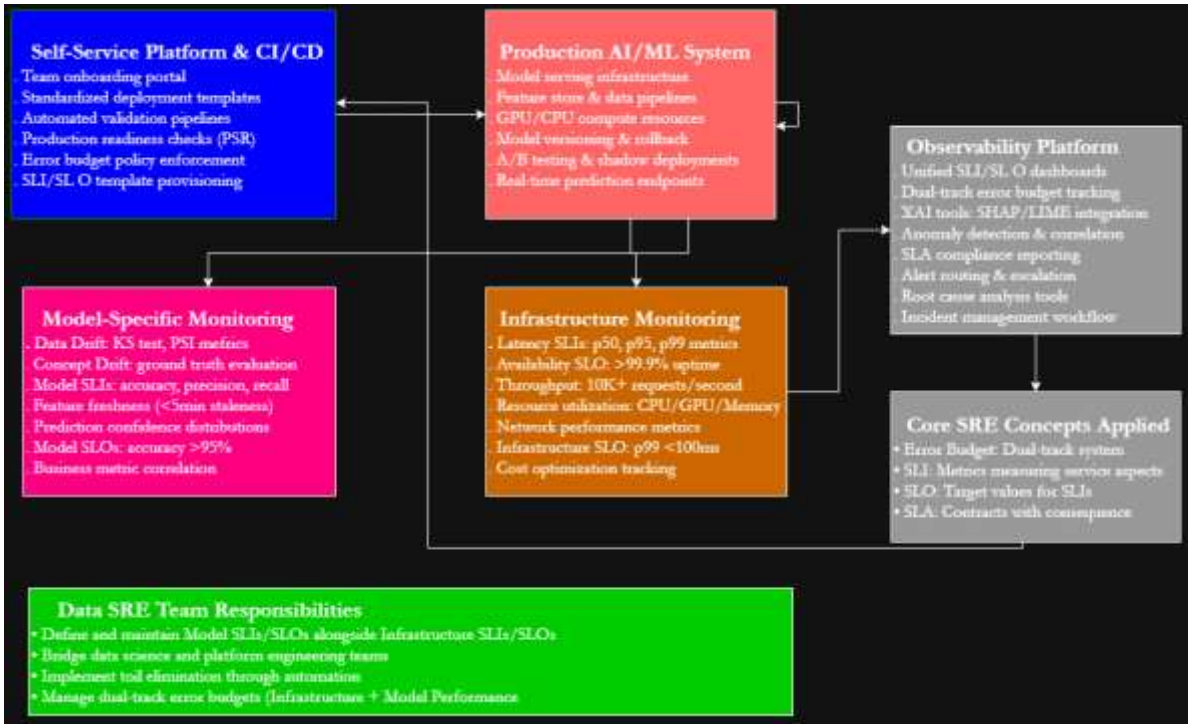

Figure 1: AI/ML Operations: SRE Architecture Framework [7,8]

The adoption of Explainable AI mechanisms turns debugging from black-box tasks into systematic investigation procedures aligned with SLO violations and business impact analysis. When Model SLIs indicate degraded performance below established thresholds, SHAP and LIME tools for feature-level attribution analysis identify precise inputs that are most responsible for prediction errors and SLO breaches [8]. Ogeti et al. observe cloud environments complicate things with multi-tenant sharing of resources and fluctuating network characteristics that can mask or accentuate model performance issues, requiring advanced correlation analysis between infrastructure statistics and patterns of model behavior [7]. This layer of explainability allows Data SREs to move beyond identifying SLI threshold violations to understanding precisely why SLOs are being missed and which feature engineering or data quality issues require remediation to restore SLA compliance.

Monitoring Component	Target Specification
Model accuracy SLO (pragmatic)	95% threshold
p99 prediction latency	Under 100ms
Data scientist accuracy proposal	99% model accuracy
Drift detection methods	KS tests, chi-square, PSI
XAI tools	SHAP and LIME

Table 3: Model Monitoring SLO Implementation Requirements [7,8]

5. Operationalizing AI Reliability Through Systematic Solutions

This shift from reactive incident response to proactive reliability engineering in AI systems calls for the deployment of systematic solutions, setting and mandating thorough SLI/SLO frameworks across the model development lifecycle. Vadavalasa points out that data validation routines are the backbone of ML reliability, with systematic validation pipelines identifying around 70% of possible production failures before deployment if implemented correctly across data ingestion, transformation, and feature engineering phases [9]. The development of Production Service Readiness frameworks specifically designed for ML workloads is a significant operational shift to guarantee adequate SLI definition before deployment. Unlike traditional PSR checklists, ML-PSR must verify that teams have defined both Infrastructure SLIs (latency, availability, throughput) and Model SLIs (accuracy, drift, confidence), established corresponding SLOs with clear business justification, and documented SLAs with defined escalation

paths and remediation procedures. This framework functions as a quality gate ensuring that models meet minimum reliability standards—expressed through SLOs—before production deployment.

The use of self-service onboarding platforms removes configuration entropy anti-patterns by automating the creation of baseline SLIs and SLOs for new models by using standardized templates and automated provisioning. Empirical research proves that validation frameworks with schema checks, statistical distribution monitoring, and anomaly detection decrease incidents related to data by large proportions when part of continuous deployment pipelines [9]. These systems offer certified deployment templates that provision SLI collection infrastructure automatically, configure default SLO thresholds based on model type (classification achieving 94% accuracy, regression maintaining RMSE <0.15, ranking preserving NDCG >0.8), establish Error Budget policies which balance innovation with reliability needs, and produce SLA documentation templates which stakeholders can review to ensure that all deployed models have appropriately defined metrics from the very beginning.

The development towards Data SRE as a niche field is an acknowledgment in the industry that setting up and maintaining Model SLIs calls for specialized knowledge across the statistical analysis and operational excellence spheres. Schröder and Schulz note four main categories of monitoring challenges in production ML systems: data quality deterioration, model performance degradation, optimization of computational resources, and alignment of business metrics, which all need expert monitoring methods and skills to manage appropriately [10]. Data SREs serve as crucial bridges, translating model requirements into measurable SLIs, negotiating SLOs that balance model performance with infrastructure constraints, and establishing SLAs that align technical capabilities with business expectations. They define composite SLIs that capture complex model behaviors—for instance, combining accuracy, latency, and confidence into a single "prediction quality" metric. They establish tiered SLOs that recognize different criticality levels—a payment fraud model might have stricter SLOs than a recommendation system.

The practical implementation of these solutions requires sophisticated tooling that continuously calculates Error Budget consumption based on both Infrastructure and Model SLO violations. When a model's accuracy drops below its SLO of 95%, the system must automatically deduct from the Model Error Budget while alerting the Data SRE team [9]. Vadavalasa mentions that organizations that apply end-to-end validation frameworks see time-to-detection for data quality bugs decrease from days to hours, with automated validation detecting schema errors, nulls, and statistical outliers before the downstream model effect is noticed. Schröder and Schulz emphasize that successful monitoring demands multi-layered solutions integrating rule-based detection for established failure patterns with statistical analysis for anomaly detection and machine learning algorithms for predictive maintenance, establishing all-encompassing safety nets against silent failures [10]. If Infrastructure SLOs are met but Model SLOs are consistently violated, the platform should prevent new feature deployments until model health is restored—enforcing the fundamental SRE principle that reliability must be maintained before new functionality is added.

Implementation Aspect	Specification/ Outcome
Failure prevention rate	70%
Model accuracy SLO	95% requirement
Classification accuracy	94% achievement
Regression RMSE	< 0.15
Ranking NDCG	Greater than 0.8

Table 4: Systematic reliability engineering deployment [9,10]

Conclusion

The integration of site reliability engineering principles with artificial intelligence operations represents a paradigm shift in organizational models for production machine learning systems, demanding a comprehensive reconceptualization of traditional monitoring and reliability practices. The framework presented demonstrates that sustainable AI operations require transcending infrastructure-focused observability to embrace holistic monitoring spanning both computational resources and model performance dimensions through carefully defined Service Level Indicators, Objectives, and Agreements. The systematic identification and avoidance of critical anti-patterns—from configuration entropy generating exponential operational overhead to monitoring frameworks producing noise without actionable signal—provides organizations with a roadmap for scaling AI deployments without proportionally scaling operational complexity. The development of Data SRE as a professional field recognizes that successful AI reliability needs professionals who can realize the statistical nature of machine learning algorithms and the operational features of production systems as imperative intermediaries between data science fantasies and infrastructure realities. By deploying dual-track error budgets, defining model-specific performance measurements with their goals, automated validation systems, and incorporating explainable AI methods to systematize the debugging process, organizations convert reactive firefighting into proactive excellence in engineering. The way forward must acknowledge that AI trustworthiness goes beyond infrastructure issues to the probabilistic reality of machine learning and that parallel adjustments in organizational structure, monitoring ideology, as well as engineering custom are needed to present systems that keep reliability, scalability, and trustworthiness at scale.

Disclaimer

The views expressed in this article are my own and do not represent the views or positions of C3.AI. I am not representing C3.AI by submitting or publishing the contents in this article.

Sreejith Kaimal is a Principal Site Reliability Engineer at C3.AI

References

[1] D Sculley et al., "Hidden Technical Debt in Machine Learning Systems", ResearchGate, 2015. [Online]. Available: https://www.researchgate.net/publication/319769912_Hidden_Technical_Debt_in_Machine_Learning_Systems
[2] Carter Happer, "Evaluating Technical Debt Reduction Through AI-Assisted Automated Code Refactoring Tools", ResearchGate, January 2025. [Online]. Available: https://www.researchgate.net/publication/394456362_Evaluating_Technical_Debt_Reduction_Through_AI-Assisted_Automated_Code_Refactoring_Tools
[3] Elife Ozturk Kiyak et al., "High-Level K-Nearest Neighbors (HLKNN): A Supervised Machine Learning Model for Classification Analysis", MDPI, 2023. [Online]. Available: https://www.mdpi.com/2079-9292/12/18/3828
[4] Betsy Beyer et al., "The Site Reliability Workbook", O'Reilly - Irita.github.io. [Online]. Available: https://lrita.github.io/images/posts/com/the-site-reliability-workbook-next18.pdf
[5] Neoklis Polyzotis et al., "Data Management Challenges in Production Machine Learning", Google. [Online]. Available: https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/46178.pdf
[6] Rajat Suvra Das, "TensorFlow: Revolutionizing Large-Scale Machine Learning in Complex Semiconductor Design", ResearchGate, 2024. [Online]. Available: https://www.researchgate.net/publication/379951674_TensorFlow_Revolutionizing_Large-Scale_Machine_Learning_in_Complex_Semiconductor_Design
[7] Pavan Ogeti et al., "Benefits and Challenges of Deploying Machine Learning Models in the Cloud", IJISAE, 2024. [Online]. Available: https://ijisae.org/index.php/IJISAE/article/view/6409/5236
[8] Sandeep Bharadwaj Mannapur, "Understanding Data Drift and Concept Drift in Machine Learning Systems", IJSRCSEIT, January 2025. [Online]. Available: https://ijsrcseit.com/index.php/home/article/view/CSEIT25111239/CSEIT25111239
[9] Rammohan Vadavalasa, "Data Validation Process in Machine Learning Pipeline", ResearchGate, 2021. [Online]. Available: https://www.researchgate.net/publication/351022721_Data_Validation_Process_in_Machine_Learning_Pipeline
[10] Tim Schröder and Michael Schulz, "Monitoring machine learning models: a categorization of challenges and methods", ScienceDirect, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2666764922000303