Applying Machine Learning to Performance Optimization in High-Volume Transaction Systems

CONF42 MACHINE LEARNING 2026

VITECH SYSTEMS GROUP

Speaker

Vivek Kumar

Conference Introduction

Vitech Systems Group

Good afternoon, everyone. My name is Vivek Kumar, and I’m a Staff Developer at Vitech Systems Group with over 16 years of experience building scalable, cloud-native enterprise platforms.
I specialize in designing high-throughput microservices architectures capable of handling massive concurrency and optimizing system performance.
Modern cloud platforms like AWS and Azure use predictive autoscaling powered by telemetry and machine learning to scale infrastructure automatically based on workload patterns.
In this session, I’ll share how machine learning can leverage system telemetry to predict bottlenecks, optimize resource allocation, and enable transaction platforms to scale intelligently with improved latency, stability, and efficiency.
Earlier at Tata Consultancy Services, Vivek contributed to India’s national tax systems and global commerce platforms for Cisco, delivering secure, scalable, and high-impact solutions.

The Challenge: When Static Rules Fail

The Reality of Modern Transaction Systems

High-volume transaction platforms generate massive amounts of operational data every second. Yet most enterprises still depend on predetermined thresholds and manual intervention to manage performance issues.

As workloads grow more volatile and architectures become increasingly distributed, these legacy approaches struggle to maintain consistent latency, throughput, and resilience under pressure.

Static Thresholds

Fixed rules can't adapt to changing patterns

Reactive Scaling

Always one step behind actual demand

Manual Tuning

Too slow for real-time optimization needs

SESSION OVERVIEW

What You'll Learn Today

01

Modern Architecture Patterns

Spring Boot microservices, RESTful APIs, and Kafka-based event streaming as the foundation for ML-driven optimization

02

Telemetry as Training Data

How operational metrics become valuable features for predictive performance models

03

ML-Driven Optimization Techniques

Adaptive autoscaling, intelligent caching, and dynamic configuration tuning across distributed services

04

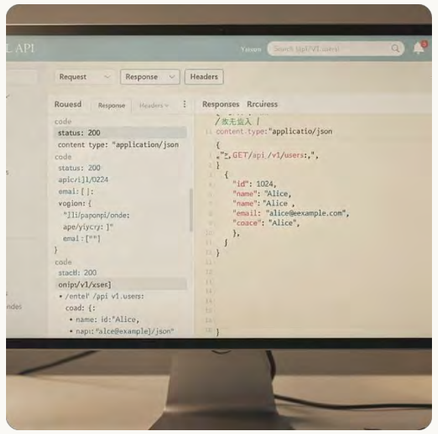
Real-World Implementation

Practical guidance on model selection, feedback loops, and production deployment strategies

Foundation: Modern Transaction Platform Architecture

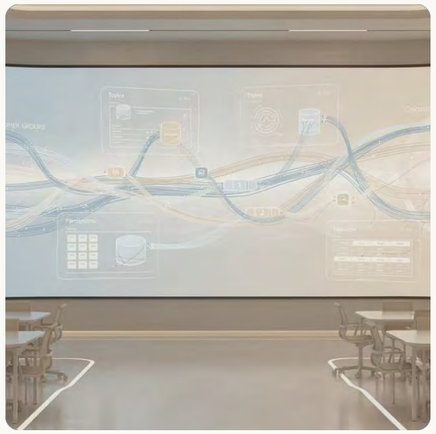
Spring Boot Microservices

Modular, independently deployable services handling discrete business functions with clear boundaries



RESTful API Layer

Standardized communication protocols enabling seamless integration and service orchestration



Kafka Event Streaming

High-throughput message backbone for asynchronous processing and event-driven workflows

From Telemetry to Training Data

Operational Intelligence

Every transaction, every API call, every queue operation generates telemetry. This continuous stream of operational data contains patterns that reveal system behavior under varying conditions.

Machine learning transforms raw metrics into predictive insights by identifying correlations between system states and performance outcomes. The key is selecting features that truly influence transaction processing efficiency.

Key Performance Indicators as ML Features

Transaction Velocity

Request rates, transaction batch sizes, and temporal patterns reveal demand trends and help forecast capacity requirements

Concurrency Patterns

Thread pool utilization, active connections, and parallel execution metrics indicate resource contention points

Queue Depth Analysis

Message backlogs, processing lag, and throughput rates signal bottlenecks before they impact end users

Resource Utilization

CPU, memory, network bandwidth, and disk I/O patterns correlate with performance degradation thresholds

Forecasting Load Spikes and Bottlenecks

Machine learning models trained on historical telemetry can predict performance issues minutes or hours before they occur. By analyzing patterns in transaction velocity, resource consumption, and queue behavior, ML algorithms identify leading indicators of system stress.

This predictive capability enables proactive intervention scaling resources, adjusting cache policies, or rerouting traffic before latency degrades or throughput drops. The shift from reactive to predictive fundamentally changes how systems maintain performance under pressure.

ML-DRIVEN STRATEGIES

Three Pillars of Intelligent Optimization

Adaptive Autoscaling

ML models predict demand curves and scale infrastructure ahead of load spikes, minimizing latency while optimizing cost. Unlike reactive threshold-based scaling, predictive models consider multiple signals and temporal patterns.

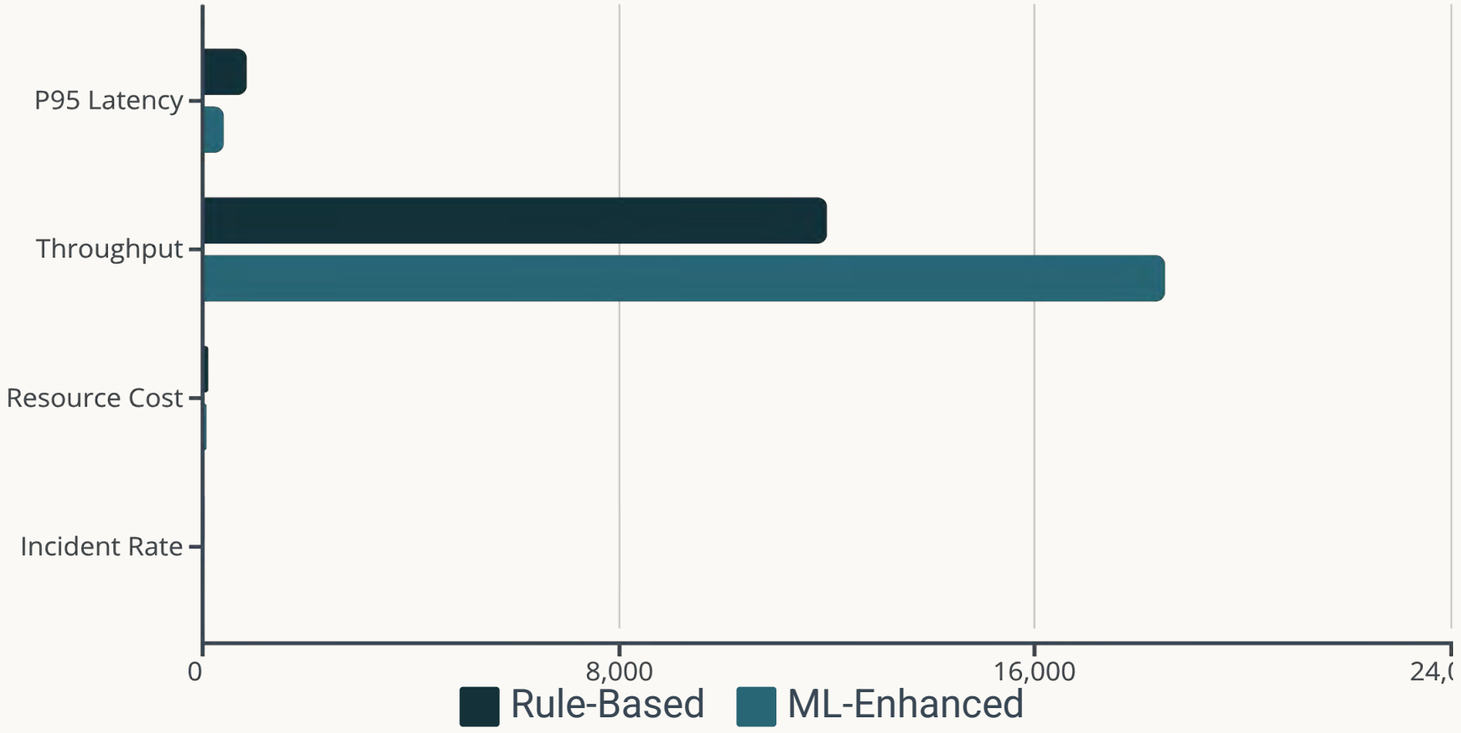
Intelligent Caching

Dynamic cache policies adjust based on access patterns, hit rates, and resource constraints. ML determines optimal cache sizes, eviction strategies, and pre-warming schedules for each service tier.

Dynamic Configuration

Real-time tuning of connection pools, timeout values, batch sizes, and retry policies across services. ML continuously optimizes parameters based on current workload characteristics and performance goals.

Performance Comparison: ML vs. Rule-Based Systems



Measurable Improvements

Comparative analysis shows ML-enhanced systems significantly outperform reactive rule-based approaches across critical dimensions.

- **Responsiveness:** 50% reduction in P95 latency through predictive scaling
- **Stability:** 73% fewer performance incidents with proactive optimization
- **Efficiency:** 28% lower infrastructure costs via intelligent resource allocation

Model Selection for Performance Engineering

Time Series Forecasting

LSTM and ARIMA models excel at predicting load patterns from historical telemetry sequences, capturing seasonal and cyclical trends

Anomaly Detection

Isolation forests and autoencoders identify unusual patterns that signal emerging bottlenecks or degradation before thresholds breach

Regression Models

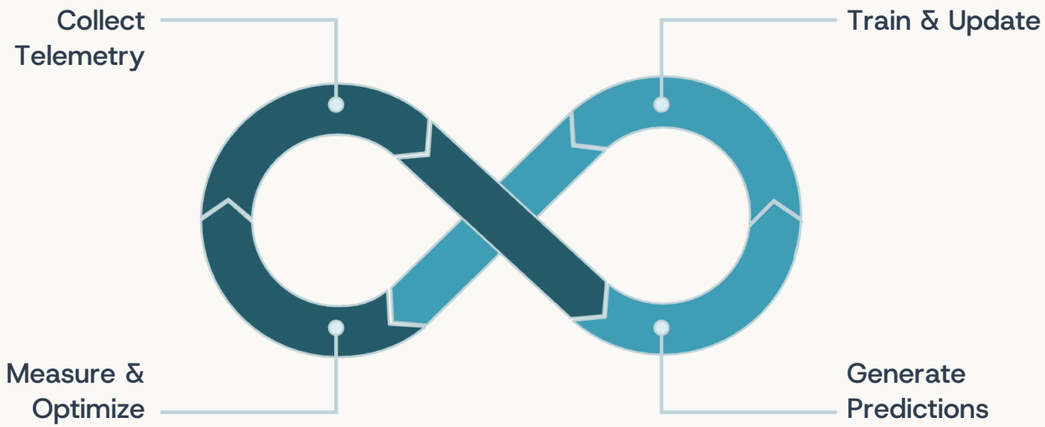
Gradient boosting and random forests predict resource requirements based on workload characteristics, enabling precise capacity planning

Reinforcement Learning

Policy-based agents learn optimal configuration tuning strategies through trial and exploration in controlled environments

Building Effective Feedback Loops

Successful ML-driven performance optimization requires closed-loop systems where predictions drive actions, outcomes are measured, and results feed back into model improvement.



This iterative process ensures models adapt to changing workload patterns, infrastructure updates, and evolving business requirements. Regular retraining with fresh data maintains prediction accuracy as system behavior shifts over time.

Deployment Considerations for Production

Model Serving Infrastructure

Low-latency inference endpoints with sub-100ms response times for real-time optimization decisions

Monitoring and Observability

Track model performance metrics alongside system KPIs to detect prediction drift and accuracy degradation

Gradual Rollout Strategy

Canary deployments and A/B testing validate ML-driven changes before full-scale implementation

Fallback Mechanisms

Maintain rule-based safeguards to handle edge cases where ML predictions may be unreliable

Production Readiness

Deploying ML models in critical transaction paths requires careful planning and risk mitigation. Models must deliver predictions with consistent latency and high availability.

Comprehensive monitoring ensures rapid detection of issues. Feature engineering pipelines need robust validation to prevent data quality problems from degrading model performance in production.

Ready to Optimize with Intelligence?

Machine learning isn't just for data scientists anymore. Platform engineers can leverage these techniques to build transaction systems that adapt, predict, and optimize autonomously.

The roadmap is clear: instrument your systems thoroughly, select models aligned with your optimization goals, deploy with proper safeguards, and iterate continuously based on real-world feedback.

Key Takeaways

Shift from Reactive to Predictive

ML transforms performance optimization from firefighting to strategic forecasting and proactive intervention

Telemetry is Your Foundation

Rich operational data from modern architectures provides the training ground for intelligent optimization models

Measurable Business Impact

ML-enhanced systems deliver superior responsiveness, stability, and resource efficiency compared to static rules

Start Small, Iterate Fast

Begin with focused use cases, establish feedback loops, and expand ML capabilities as you validate results

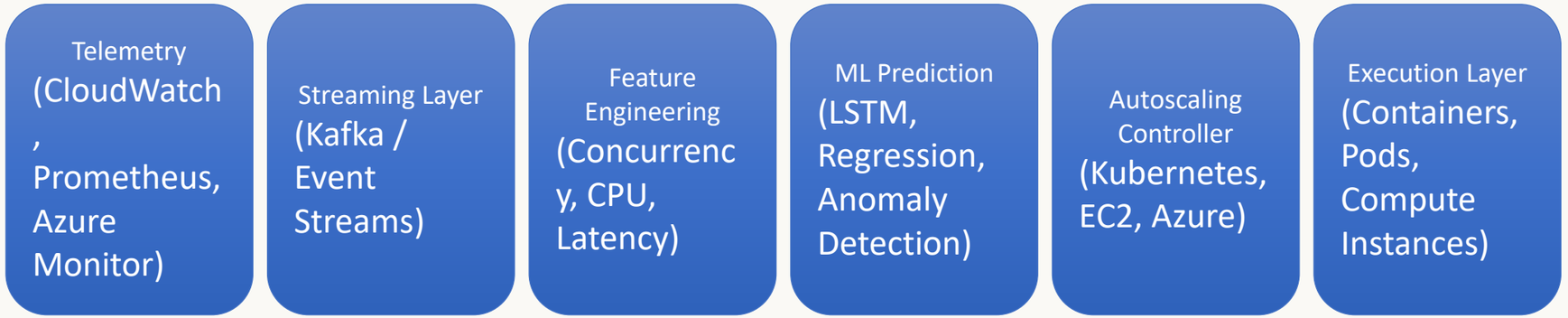
Real-World Cloud Scaling Examples: AWS & Azure

AWS EC2 Predictive Auto Scaling: Uses machine learning to forecast CPU utilization, request rates, and concurrency patterns to proactively launch instances before traffic spikes occur.
AWS Lambda Concurrency Scaling: Automatically scales from a few executions to tens of thousands of concurrent threads within seconds without manual provisioning.
Azure Kubernetes Service (AKS): Dynamically scales pods and nodes based on thread utilization, queue depth, CPU pressure, and memory consumption.
Azure App Service Autoscaling: Automatically adds or removes compute instances based on HTTP queue length, CPU utilization, and request throughput.
Real-world impact: Enables systems to handle millions of transactions with minimal latency, improved stability, and optimized infrastructure cost.

ML-Driven Performance Optimization Architecture

Telemetry Collection Layer: Application metrics, thread pool utilization, API latency, CPU, memory, and queue depth collected via monitoring tools (CloudWatch, Azure Monitor, Prometheus).
Streaming & Storage Layer: Kafka or event streaming pipelines capture real-time telemetry and store historical data in data lakes or time-series databases.
Feature Engineering Layer: Extract key features such as concurrency patterns, workload trends, and resource utilization for ML model training.
Machine Learning Prediction Layer: ML models (LSTM, regression, anomaly detection) forecast load spikes, bottlenecks, and optimal resource requirements.
Decision & Autoscaling Layer: Autoscaling controllers trigger scaling actions in AWS EC2, Kubernetes, or Azure App Services based on ML predictions.
Execution Layer: Infrastructure dynamically scales pods, containers, or compute instances to maintain optimal performance and latency.
Feedback Loop: Continuous monitoring feeds new telemetry back into ML models for retraining and continuous optimization.

Visual Architecture: ML-Driven Performance Optimization



LinkedIn URL: https://www.linkedin.com/in/vivek-kumar-68384616/ Gmail : vivekimp574@gmail.com

Thank You!

Vivek Kumar || Conf42 Machine Learning 2026 || Vitech Systems Group