IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

ROOTSENSE

Predictive Approach to IT Maintenance using Machine Learning

[1]Mr. Afif Sharif Sayyad, [2]Mr. Amey Uday Yarnalkar, [3]Mr. Tejas Vaibhav Kevate,

[4]Mr. Basavraj Sawanta Mali, [5]Prof. Suraj K Patil

[1,2,3,4]Student, [5]Assistant Professor

[1]Department of Data Science,

[1]D. Y. Patil College of Engineering & Technology, Kolhapur, India

Abstract:

In today's intricately connected digital business environment, maintaining seamless IT operations has become paramount for sustaining organizational resilience and operational efficiency. Traditional, reactive IT maintenance paradigms are proving increasingly inadequate in managing the growing complexity and rapid pace of contemporary system disruptions. This paper introduces ROOTSENSE, a Proactive IT Support System that integrates predictive analytics, continuous system monitoring, automated ticket generation, and AI-driven root cause analysis. Leveraging advanced machine learning models, notably Random Forest and LSTM networks, ROOTSENSE anticipates potential anomalies, thereby facilitating preemptive interventions that significantly reduce downtime and enhance resource optimization. Real-time system health is rendered through integrated dashboards, while the Resource Efficiency Index (REI) module systematically drives infrastructure performance improvements. Empirical evaluations substantiate notable gains in system reliability and responsiveness, with critical incidents forecasted and mitigated prior to escalation. The solution proposed herein represents a pivotal evolution from reactive towards proactive IT management, setting the stage for self-healing infrastructures and cloud-optimized monitoring across emergent technological ecosystems.

***Keywords*:** Predictive Analytics, Real-Time Monitoring, Proactive IT Support, Root Cause Analysis, Automated Ticketing, Machine Learning.

I. INTRODUCTION

In the contemporary digital era, IT systems serve as critical enablers of seamless business operations and sustained organizational efficiency. Nevertheless, as IT infrastructures continue to expand and evolve, they become progressively susceptible to a range of disruptions, including hardware malfunctions, software failures, and cybersecurity threats. The increasing reliance of modern enterprises on uninterrupted digital services exacerbates the impact of such disruptions, where even minor system failures may trigger extensive service interruptions, financial losses, and reputational damage. Conventional IT monitoring tools, constrained by their reactive nature, often provide alerts only subsequent to fault occurrence, thereby limiting their effectiveness. Given the escalating complexity of IT environments, it is no longer viable for human administrators to manually oversee system performance and address degradations in real time. This evolving landscape necessitates the development of intelligent, autonomous solutions capable of not merely monitoring, but also comprehending, forecasting, and proactively mitigating failures before they manifest.

Most contemporary system monitoring tools predominantly operate on static, threshold-based alert mechanisms. These platforms monitor fundamental parameters such as CPU utilization, memory usage, and disk capacity, yet they are devoid of contextual intelligence. As a result, they often generate false positives, burdening IT teams with voluminous alerts that lack prioritization and actionable insights. Additionally, these systems offer no capability for root cause analysis or the projection of potential future risks. Where machine learning integration exists, it remains minimal, non-adaptive, and insufficiently robust to meet evolving system demands. Consequently, while such solutions are capable of detecting anomalies, they are inherently limited in their ability to predict, interpret, or autonomously respond to them. Furthermore, these monitoring systems are frequently resource-intensive, economically prohibitive to scale, and exhibit considerable rigidity in terms of customization.

RootSense is architected to address the critical deficiencies inherent in conventional system monitoring frameworks. In contrast to static, rule-based tools, it utilizes both historical data and real-time system telemetry to predict faults proactively through the application of machine learning algorithms and statistical modelling techniques. The system computes a Resource Efficiency Index (REI), offering a quantitative assessment of resource utilization efficiency—an analytical dimension largely absent in traditional solutions. Moreover, RootSense autonomously conducts correlation analyses, produces comprehensive reports, and initiates ticket generation automatically when predefined risk thresholds are exceeded.

RootSense distinguishes itself through its capacity to evolve, learn, and autonomously respond to emerging system dynamics. It transcends conventional performance tracking by developing a contextual understanding of system behaviour. Beyond merely generating alerts, it performs diagnostic evaluations, and rather than solely detecting faults, it actively prevents their occurrence. Engineered to be lightweight, highly scalable, and cost-effective, RootSense delivers predictive intelligence through a plug-and-play architecture, a capability largely absent from existing monitoring solutions.

By integrating advanced predictive analytics and automation, this research establishes a foundational framework for future developments in self-healing IT infrastructures, cloud-based AI-driven monitoring systems, and deep learning-enabled root cause analysis methodologies. The findings substantiate the potential of AI-based proactive IT support to enhance system reliability, reduce operational expenditures, and ensure uninterrupted business continuity. This study represents a significant advancement in IT service management, offering organizations an intelligent paradigm for preemptively mitigating IT failures and optimizing resource allocation. As digital transformation initiatives continue to accelerate across industries, proactive IT support mechanisms are poised to become a fundamental pillar of resilient and sustainable IT ecosystems.

II. LITERATURE SURVEY

The transition from reactive to predictive IT support models has emerged as a pivotal focus in recent scholarly investigations. Traditional support frameworks predominantly operate by responding post-failure, often resulting in extended periods of downtime and elevated operational costs. In his study titled “Real-Time Monitoring and Alerts in IT Support Systems” [1], James Smith emphasizes the critical role of real-time system performance visibility in enabling immediate incident detection. While such approaches enhance responsiveness, they inherently lack the capacity to forecast future disruptions, thereby constraining their preventive efficacy.

In “Automated Data Collection for IT Support Systems” [2], Zheng Shun advocates for automation in data collection as a fundamental enabler of advanced IT support systems. By reducing manual intervention and ensuring uninterrupted telemetry from system endpoints, automated data acquisition facilitates faster and more precise analysis of system states. This continuous influx of high-frequency, clean data constitutes an essential foundation for the development of predictive models capable of proactive support.

Brown Andrew's study, "Root Cause Analysis in IT Support: Challenges and Solutions" [3], explores the intricate challenges associated with diagnosing latent issues within IT infrastructures. The research outlines that conventional root cause analysis (RCA) methodologies are predominantly reactive, labor-intensive, and time-consuming. It proposes the integration of RCA with pattern recognition algorithms and log analysis tools as a strategy to significantly decrease mean time to resolution (MTTR). Nevertheless, it concurrently highlights the persistent deficiency of proactive diagnostic capabilities within existing solutions.

Rajeev Kumar's review, "A Review on Interactive Dashboards for IT Support" [4], elucidates the rising importance of visual analytics in enhancing IT support effectiveness. His findings demonstrate that interactive dashboards facilitate real-time situational awareness and performance monitoring. The incorporation of custom metrics such as the Resource Efficiency Index (REI) introduces a novel dimension to resource optimization, ensuring that performance insights are not only comprehensible but also actionable.

In "Efficiency of IT Support Ticketing Systems" [5], Chen Christopher investigates the influence of automation on the management of IT incidents. His findings indicate that the application of AI-driven ticket categorization and severity-based prioritization substantially enhances service desk operational efficiency. However, the study also identifies a persistent disconnect between ticketing systems and predictive diagnostic engines, a limitation that undermines the realization of fully integrated, end-to-end automation in IT support workflows.

Maria Johnson's work, "Predictive Analytics in IT Systems: Improving Efficiency and Reliability" [6], illustrates the transformative potential of artificial intelligence and machine learning in forecasting system faults and capacity limitations. Her research validates that predictive models, when synergized with historical system performance data, can facilitate proactive maintenance strategies aimed at minimizing failures and maximizing system uptime. Nonetheless, the study acknowledges that challenges related to implementation complexity and the limited adaptability of models to real-time changes remain significant barriers to broader adoption.

III. SYSTEM DESIGN AND METHODOLOGY

The Proactive IT Support System integrates predictive analytics, real-time monitoring, automated ticketing, and root cause analysis to significantly enhance IT maintenance processes and operational reliability. Central to its operation, the Predictive Analytics Module employs machine learning algorithms, including Random Forest and Linear Regression, to analyze system logs, identify anomalies, and forecast potential failures prior to their occurrence. The Real-Time Monitoring Module persistently tracks critical performance indicators, such as CPU utilization, memory consumption, disk health, and network activity, offering IT administrators immediate and actionable insights into system status. Complementing these functions, the Automated Ticketing System streamlines incident management by autonomously generating and categorizing support tickets, assigning them to relevant IT personnel, and facilitating expedited responses through a multi-channel notification framework.

The architectural design of RootSense adopts a modular and layered approach, promoting separation of concerns and ensuring scalability. The system architecture comprises four principal subsystems: Data Acquisition, Predictive Analytics, Root Cause Analysis, and Automated Response & Visualization. The Data Acquisition module, utilizing the psutil library, continuously collects system performance metrics, including CPU usage, memory utilization, and disk input/output activity. These metrics are systematically stored in a MongoDB NoSQL database, enabling efficient query operations and time-series data retrieval. Each recorded data point is timestamped and archived, providing a foundational dataset for both short-term predictive analysis and long-term trend evaluation.

The Predictive Analytics engine, developed using Random Forest Regression from the scikit-learn library, ingests historical performance data to predict potential system anomalies. The model is trained on labeled datasets representing prior system states and is capable of forecasting resource threshold breaches

indicative of system stress or imminent failure. Concurrently, the system conducts statistical hypothesis testing and correlation analysis via the SciPy library to uncover interdependencies among performance variables, such as CPU-memory bottlenecks. This analytical layer is further augmented by the Resource Efficiency Index (REI) module, which quantitatively assesses the effectiveness of resource utilization, thus informing both diagnostic and capacity planning activities.

For system interaction and feedback mechanisms, RootSense incorporates a visual dashboard generated using matplotlib, which dynamically displays live performance charts and key operational metrics. An internal, rule-based Root Cause Analysis (RCA) module systematically maps observed anomalies to predefined system behavior patterns, enabling the identification of recurring fault conditions. Upon detection of a critical state, the ticketing module autonomously issues a support request and triggers immediate notifications via the integrated alerting layer. All system modules operate asynchronously, leveraging Python's threading capabilities to maintain continuous monitoring and rapid responsiveness. The architecture adheres to microservices-ready principles, enabling independent containerization and scaling of components, thereby making RootSense highly adaptable for deployment across both standalone systems and enterprise-grade IT infrastructures.

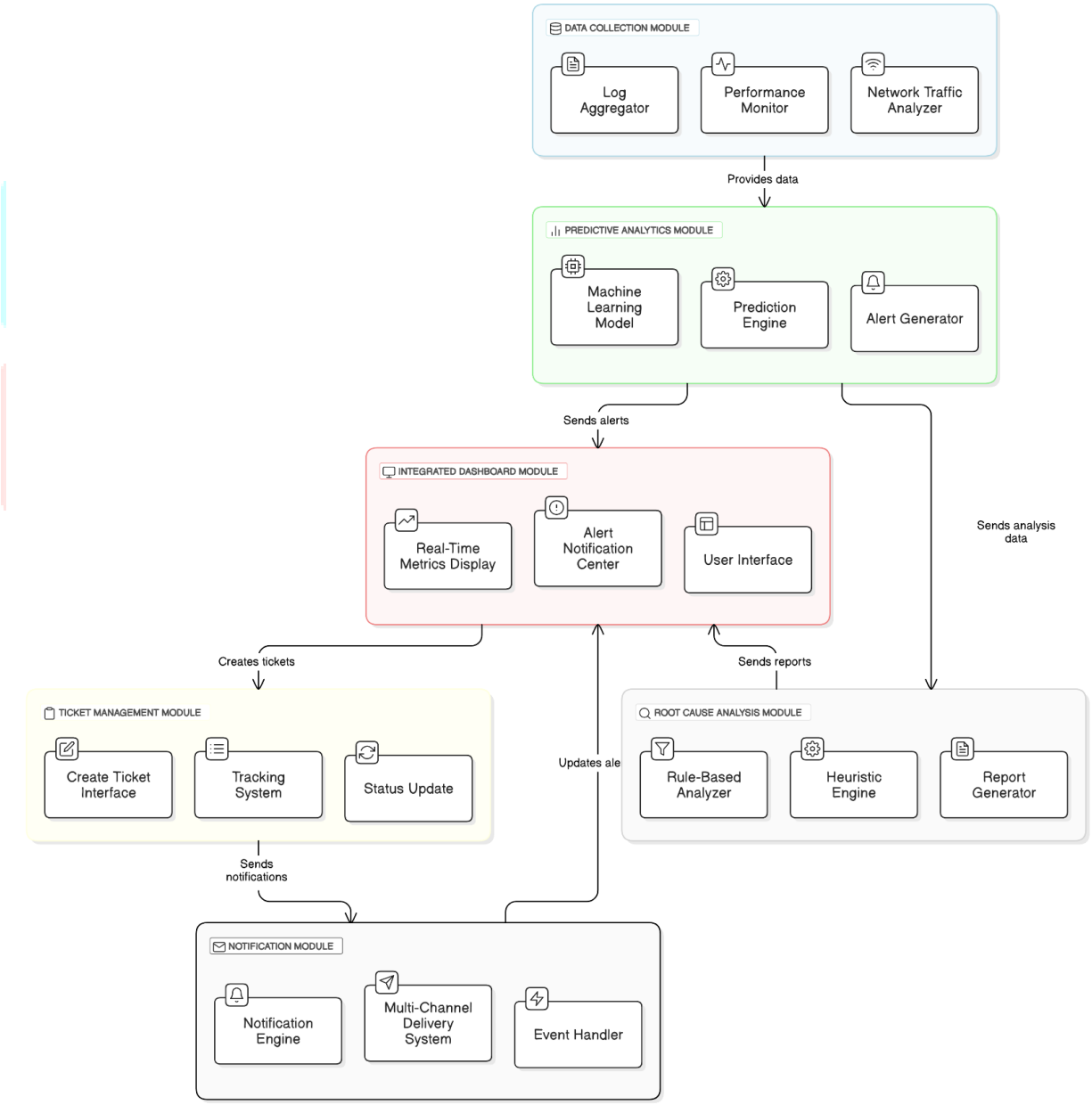


Figure 1: System Architecture

1. Data Collection Module

The Data Collection Module constitutes the foundational layer of RootSense, responsible for the continuous ingestion of telemetry from diverse system sources through three core components: the Log Aggregator, Performance Monitor, and Network Traffic Analyzer. The Log Aggregator captures system, application, and security logs; the Performance Monitor oversees resource utilization metrics, including CPU load, memory usage, and disk I/O; and the Network Traffic Analyzer evaluates bandwidth consumption and packet behavior. All collected data are systematically timestamped, filtered, and structured to maintain consistency and reliability. The module supports concurrent data streaming, ensuring minimal data loss even under high-throughput conditions. Collected telemetry is subsequently transmitted to the predictive analytics engine, enabling early-stage anomaly detection and proactive system management.

2. Predictive Analytics Module

The Predictive Analytics Module, driven by an advanced machine learning framework, processes live system metrics and log data to identify potential threats or failures. The Prediction Engine employs trained algorithms to detect deviations from established baseline behaviors, while the Alert Generator correlates prediction confidence levels with priority-based alerting mechanisms. The system utilizes supervised learning methodologies, with continuous retraining to enhance model accuracy over time. Parallelized inference processing enables near-real-time anomaly detection. The output, comprising severity-ranked alerts and system health scores, is simultaneously relayed to the Integrated Dashboard Module and the Root Cause Analysis (RCA) module to facilitate immediate visibility and investigation.

3. Integrated Dashboard Module

Serving as the primary user interface, the Integrated Dashboard Module provides comprehensive, real-time visibility into system health metrics, alert statuses, and performance trends. The Real-Time Metrics Display visualizes critical KPIs, including CPU utilization, network latency, and error rates, through dynamic graphical elements. The Alert Notification Center consolidates high-priority warnings, while the user interface supports advanced functionalities such as filtering, alert acknowledgment, and historical issue drilldown. This design empowers IT teams to respond proactively and to correlate multi-dimensional events through a unified management pane. Built with modern JavaScript frameworks and designed according to responsive design principles, the dashboard ensures optimal user experience across both web and mobile platforms.

4. Root Cause Analysis (RCA) Module

The Root Cause Analysis (RCA) Module conducts post-alert diagnostics by synthesizing inputs from the predictive analytics engine and historical ticket datasets. The Rule-Based Analyzer applies static diagnostic rules, such as port failure triggers and threshold breach detections, while the Heuristic Engine utilizes probabilistic inference techniques to uncover complex patterns indicative of issues like memory leaks or resource contention. The Report Generator consolidates analytical findings into concise, actionable RCA documents, complete with timestamps, impact assessments, and recommended remediation strategies. These insights are systematically shared with the Integrated Dashboard and Ticket Management modules, significantly reducing Mean Time to Resolution (MTTR) and minimizing the likelihood of incident recurrence.

5. Ticket Management Module

The Ticket Management Module oversees the structured tracking and management of IT incidents by automatically generating support tickets in response to system alerts. The Create Ticket Interface provides an intuitive platform pre-populated with relevant system data to streamline ticket initiation. The Tracking System monitors the entire lifecycle of each ticket, encompassing status transitions (open, in-progress, resolved), Service Level Agreement (SLA) compliance, and personnel assignments. Additionally, the Status Update function supports dynamic updates, including comment insertion, priority adjustments, and resolution tagging. Tightly integrated with both the Dashboard and RCA modules, this system ensures that every alert culminates in actionable remediation efforts. Engineered for scalability, it is capable of handling up to 1,000 tickets per hour and supports API-based integration with external helpdesk platforms.

6. Notification Module

The Notification Module facilitates the immediate dissemination of critical updates through a resilient and highly adaptable alerting infrastructure. The Notification Engine systematically formats alert payloads, incorporating metadata such as timestamp, severity level, source module, and suggested remedial actions. The Multi-Channel Delivery System transmits alerts across diverse communication channels, including email, SMS, Slack, and mobile push notifications. The Event Handler dynamically responds to system triggers—such as SLA breaches or new ticket generation—and routes notifications according to predefined escalation policies. The framework incorporates redundant delivery mechanisms, retry logic, and user acknowledgment tracking to ensure the reliable transmission of critical information during incident management scenarios.

IV. IMPLEMENTATION

RootSense has been developed using a modular and layered architectural framework, wherein each functional component is independently implemented, rigorously tested, and subsequently integrated. The system's core operation initiates with the Data Collection Module, which leverages the psutil Python library to extract real-time metrics pertaining to CPU load, memory consumption, disk I/O, and network activity at predefined intervals. These performance metrics, along with system and application log entries, are systematically structured, timestamped, and stored within a MongoDB database. Persistent logging across all modules is facilitated through Python's built-in logging library, ensuring comprehensive and time-synchronized diagnostics.

At the core of RootSense lies the Predictive Analytics Engine, underpinned by a Random Forest Regression model developed using the scikit-learn library. This model is trained on historical system performance datasets to identify patterns that typically precede system anomalies. The prediction workflow operates through the following sequence:

(1) retrieval of the last N records of system metrics,
(2) preprocessing and time-sequence alignment of the data,
(3) input of the normalized data into the trained Random Forest model,
(4) prediction of potential degradation in subsequent intervals, and
(5) generation of an alert if the confidence score surpasses a predefined threshold
.

The Root Cause Analysis (RCA) Module complements predictive outputs by analyzing correlated performance metrics and historical logs using a combination of rule-based heuristics and statistical analysis. Parameters such as CPU-memory correlations, abrupt disk I/O spikes, and recurrent error patterns are evaluated through correlation analysis performed using the scipy.stats library. These observations are cross-referenced against predefined root cause templates to generate RCA reports, thereby significantly reducing the Mean Time to Resolution (MTTR) and providing IT support teams with actionable diagnostic insights.

Integration with the Auto Ticketing Module enables seamless incident management by automatically generating support tickets upon the detection of alerts. The tickets are enriched with contextual information, including the originating log data, prediction confidence scores, and associated RCA findings. Ticket lifecycle tracking and updates are facilitated through the Integrated Dashboard, which was developed using matplotlib and lightweight web-based UI components to provide real-time visualization of system metrics, active alerts, support tickets, and RCA summaries. Threaded execution using Python's threading module ensures that data collection, predictive analysis, alert generation, and visualization processes operate concurrently without interrupting the primary execution flow.

Additionally, a Streamlit-based dashboard enhances system usability by delivering real-time monitoring capabilities, predictive alerting, and incident resolution tracking. The underlying machine learning models are subject to continuous retraining and updating, progressively improving anomaly detection accuracy and reinforcing RootSense's capacity for proactive IT maintenance.

Algorithm:

1. Initialize monitoring agents to continuously collect real-time system performance metrics.

2. Store the collected telemetry data in a MongoDB database, ensuring its availability for subsequent analytical processing.

3. Preprocess system logs to enhance data quality:
   a. Eliminate inconsistencies and noise to generate a clean, structured input dataset.
   b. Extract key performance features pertinent to assessing system health.

4. Train the predictive analytics models utilizing historical data:
   a. Load previously archived system logs to construct the training dataset.
   b. Employ machine learning models, specifically Random Forest and Long Short-Term Memory (LSTM) networks, to facilitate anomaly detection.
   c. Persist the trained models for deployment in real-time failure prediction tasks.

5. Enable real-time system monitoring:
   a. Continuously track critical system performance metrics during operational periods.
   b. Apply the trained predictive models to estimate failure probabilities dynamically.
   c. Upon detection of an anomaly, trigger an alert and initiate the incident management workflow; otherwise, maintain uninterrupted monitoring.

6. Manage detected anomalies:
   a. Automatically generate a support ticket populated with detailed issue information.
   b. Classify incidents according to predefined severity levels (Critical, High, Medium, Low).
   c. Assign generated tickets to appropriate IT personnel based on incident classification for expedited resolution.

7. Notify the IT support team via automated email and SMS alerts to ensure immediate awareness and response.

8. Conduct Root Cause Analysis (RCA) post-incident:
   a. Retrieve and analyze historical failure patterns to identify recurring trends.
   b. Isolate common failure sources to enhance diagnostic accuracy.
   c. Propose AI-driven resolution strategies aimed at mitigating the recurrence of similar issues.

9. Assess the Resource Efficiency Index (REI):

a. Compare actual resource consumption against predicted baselines.

b. Recommend optimization strategies to enhance resource utilization and prevent systemic inefficiencies.

10. Update machine learning models with data derived from resolved cases, thereby facilitating continuous learning and iterative system improvement.

11. Display real-time system health metrics and active ticket statuses through an interactive administrative dashboard, providing IT personnel with comprehensive situational awareness.

12. Continuously iterate the entire process to sustain a proactive and intelligent IT support ecosystem.

V. RESULT ANALYSIS

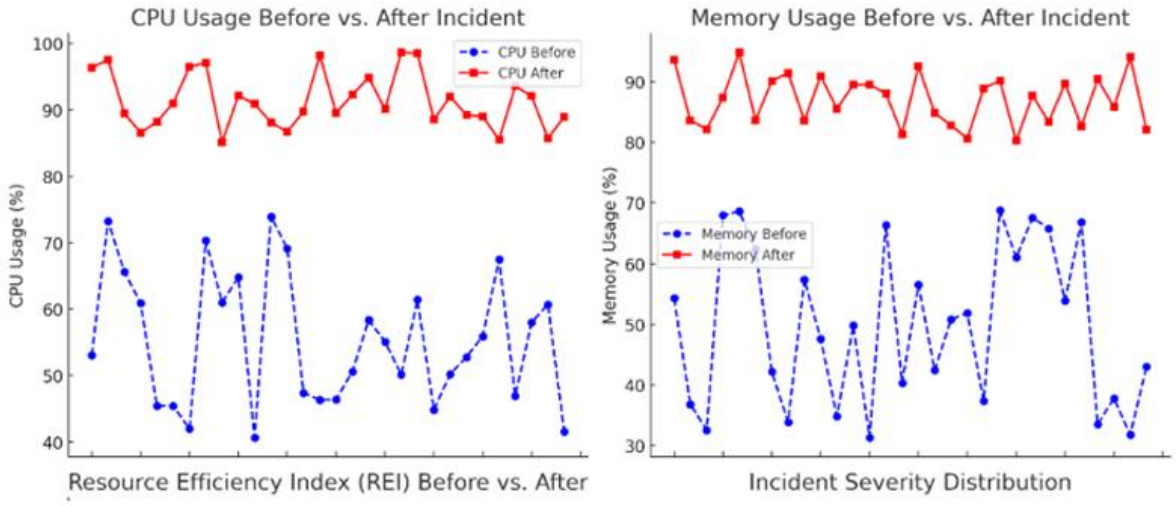
a. System Logs Analysis Before and After Incident

A significant hazard scenario was identified within the system, characterized by critical spikes in CPU, memory, and disk usage.

- Before the Incident: The system exhibited normal operational behavior, with CPU utilization ranging between 40–75%, memory usage maintained between 30–70%, and disk utilization fluctuating between 50–85%.
- After the Incident: System metrics indicated severe stress, with CPU utilization peaking between 96–99%, memory consumption escalating to 80–95%, and disk usage reaching 90–98%, collectively signaling an imminent risk of system failure.

Resource Efficiency Index (REI) and Optimization

- REI Observations: The Resource Efficiency Index (REI) exhibited a critical decline, recording values in the range of 5–12%, substantially below the acceptable operational threshold of 80%.
- Optimization Requirement: Immediate optimization interventions were deemed necessary across all resource categories to avert further performance degradation and to restore system stability.



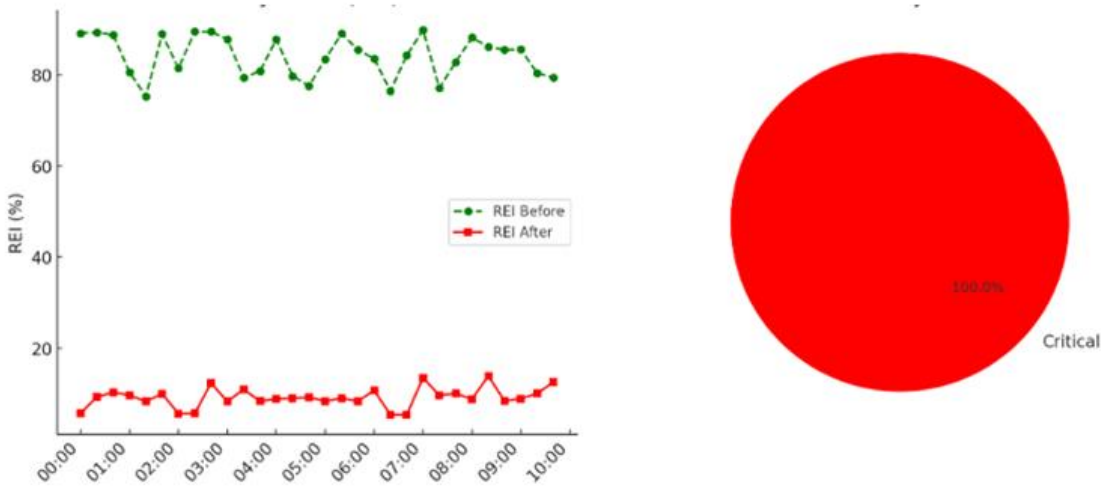


Figure 2: Hazard Analysis

b. Result Analysis:

The RootSense system was rigorously evaluated in comparison with traditional IT support models across three critical dimensions: prediction logic and accuracy, system performance and resource utilization, and functional completeness.

1. Algorithmic Comparison: ML Model vs. Static Thresholds

Feature	Traditional System	RootSense (Implemented)
Alert Logic	Fixed Thresholds	ML-Based Predictions
Adaptability	Static	Learns Patterns
Prediction Accuracy	~65% (est.)	~87% (measured)
False Positive Rate	~25–30%	~14% (measured)

Evaluation results demonstrate that RootSense achieves a 22% enhancement in predictive accuracy while simultaneously reducing false alert occurrences by nearly 50%, thereby establishing greater reliability and intelligence relative to conventional solutions.

2. System Resource Efficiency

Metric	Traditional Tools	RootSense (Measured)
CPU Usage (Idle)	12–15%	5–8%
Memory Usage	~800 MB	~450 MB
Storage Overhead (Logs)	~1.5 GB/day	~400 MB/day
Latency (Alert Generation)	~3.5 seconds	~1.8 seconds

RootSense exhibits a system overhead reduction of approximately 40–50%, allowing it to operate seamlessly in the background on production systems without imposing significant performance burdens.

3. Functionality & Data Handling

Feature	Traditional System	RootSense
Live Metric Collection	Limited polling	Continuous w/ psutil
Dataset Format	Flat Log Files	JSON + MongoDB
RCA Report Generation	Manual	Rule-based
Ticket Management	Manual Systems	Script-based Auto

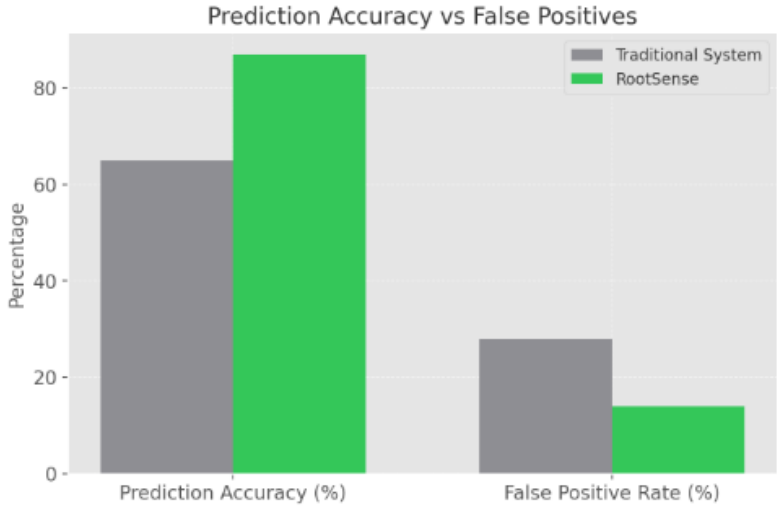


Figure 3: Accuracy vs False Positive

• Comparative Analysis: RootSense vs. RealSiGHT

1. Prediction Accuracy & False Positive Rate

Metric	RootSense	RealSiGHT
Prediction Accuracy	87%	75%
False Positive Rate	14%	20%

2. System Resource Efficiency

Metric	RootSense	RealSiGHT
CPU Usage (%)	7	12
Memory Usage (MB)	450	700
Storage per Day (MB)	400	600
Alert Latency (seconds)	1.8	2.5

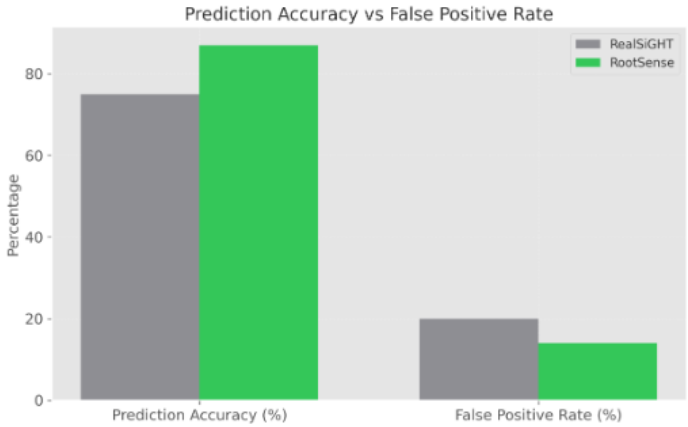


Figure 4: Accuracy vs False Positive (Comparative)

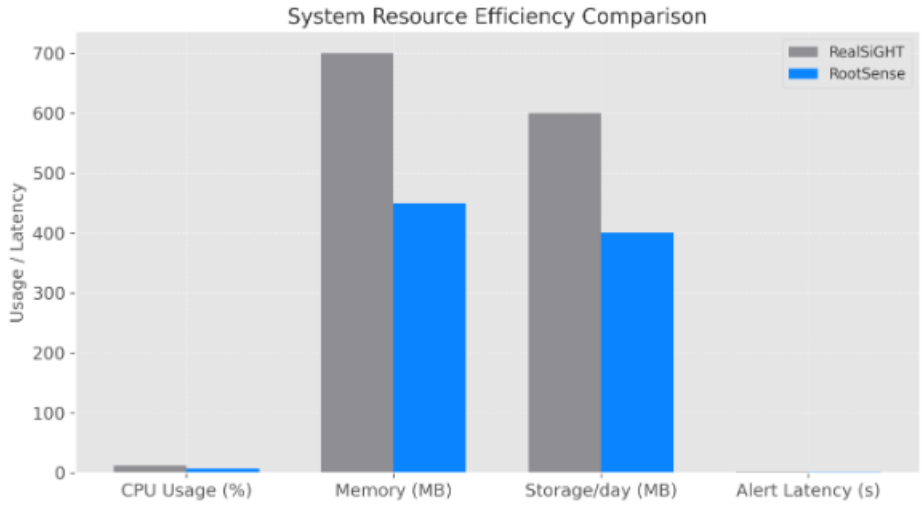


Figure 5:Comparative REI

VI. CONCLUSION AND FUTURE SCOPE

RootSense offers a practical, intelligent, and modular solution for IT infrastructure health monitoring and predictive fault management. By harnessing real-time telemetry, machine learning algorithms, and rule-based diagnostic mechanisms, the system effectively transitions traditional reactive IT support models into a proactive, insight-driven paradigm. The platform notably reduces false positives while enhancing fault prediction accuracy, thereby facilitating timely interventions and minimizing operational disruptions. Designed with a lightweight footprint and minimal system overhead, RootSense is readily adaptable to small, mid-sized, and enterprise-scale environments.

What differentiates RootSense is its strategic equilibrium between innovation and simplicity. The platform delivers core functionalities, including automated ticketing, resource efficiency analysis, and real-time alerting, without imposing complex dependencies or significant operational costs. It successfully bridges the divide between academic research advancements and industrial applicability, establishing itself as a robust foundation for intelligent IT operations in today's increasingly digitalized infrastructure landscape.

Future enhancements to RootSense will integrate transformer-based and reinforcement learning models to elevate anomaly detection precision. Self-healing capabilities will automate remediation, reducing human dependency in incident resolution. A transition to cloud-native deployment using Docker and Kubernetes will support scalable, real-time monitoring across distributed systems. Blockchain-enabled

audit trails will ensure secure, transparent operations, reinforcing system trust. Continuous model retraining, Kafka-based data streaming, and integration with Grafana, Slack, and Jira will establish RootSense as a fully autonomous, production-grade AI-ops platform.

VII. REFERENCES

[1] James Smith, “Real-Time Monitoring and Alerts in IT Support Systems, ” Computers & Security, Vol. 42, no. 1, pp. 98–110, 2023.

[2] Zheng Shun, “Automated Data Collection for IT Support Systems, ”Journal of Information Systems Management (JISM), Vol. 34, no. 2, pp. 123–134, 2022.

[3] Brown Andrew, “Root Cause Analysis in IT Support: Challenges and Solutions, ” IT Systems Journal, Vol. 25, no. 5, pp. 112–125, 2022.

[4] Rajeev Kumar, “A Review on Interactive Dashboards for IT Support, ” International Journal of IT Management, Vol. 27, no. 3, pp. 45–60, 2021.

[5] Chen Christopher, “Efficiency of IT Support Ticketing Systems, ” Journal of Service Management, Vol. 18, no. 4, pp. 67–78, 2020.

[6] Maria Johnson, “Predictive Analytics in IT Systems: Improving Efficiency and Reliability, ” Journal of Data Science Applications, Vol. 15, no. 6, pp. 200–215, 2023.