IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

PROACTIVE FAULT MANAGEMENT IN MODERN OFFICE IT ENVIRONMENTS USING MACHINE LEARNING

[1]P. Janarthanan, Vishnu J., [2]Sujith Kumar C. and [3]Tejeshwaran C
[1,2,3]Department of Computer Science and Engineering,
Sri Venkateswara College of Engineering, Sriperumbudu , Chennai – 602117

Doi: https://doi.org/10.56975/ijcrt.v14i7.309048

Abstract: Modern IT environments are highly complex and technical, and prone to sudden system failures that impact productivity and significantly increase the cost of maintenance. Traditional fault detection methods rely on reactive support ticket generation, where issues are addressed only after system failure. This paper presents a machine learning-based framework that proactively identifies faults using real-time system performance metrics. System parameters including CPU utilization, memory usage, disk activity, network latency, application response time, and security indicators are continuously monitored. The proposed model predicts failure likelihood and classifies faults into hardware, software, network, or security categories, automatically generating support tickets for timely IT intervention. Three classifiers — KNN, SVM, and RFC — are evaluated on an 8,000-record Kaggle dataset. RFC achieves perfect classification accuracy of 100%, whereas KNN and SVM achieved 99%. The system reduces downtime, enhances reliability, and aids in timely maintenance across IT infrastructures.

Index Terms **-** Predictive Maintenance, Machine Learning, Failure Classification, Support Ticket Generation, Random Forest, KNN, SVM, System Monitoring.

1. INTRODUCTION

In modern office environments, IT infrastructure forms the backbone of daily operations, supporting communication, data processing, and business-critical applications. As organizations increasingly rely on complex and interconnected systems, unexpected failures due to hardware degradation, software anomalies, network congestion, or security threats have become more frequent and costlier to resolve [1]. The growing adoption of Industry 4.0 principles has further intensified this dependency, making robust fault management a critical operational requirement across sectors [2]. Conventional IT infrastructure management follows a reactive approach, detecting faults only after failure has occurred, which results in service downtime and increased operational costs. A comprehensive survey by Vishwakarma et al. [2] identifies this reactive paradigm as a primary bottleneck in modern industrial systems and argues that predictive, data-driven approaches are essential for sustainable operations. Gao et al. [16] trace the evolution of fault diagnosis from classical rule-based methods to modern learning-based frameworks, highlighting the limitations of thresholdbased monitoring in complex, multi-component systems. Machine learning offers a compelling route to proactive fault management by continuously analyzing system performance metrics for early identification of potential threats [4]. Random Forest and Support Vector Machine classifiers have demonstrated strong performance across fault detection tasks, from power system diagnostics [13] to real-time industrial signal processing [17]. Network security anomalies, which constitute one of the five fault categories addressed in this work,

have also been modelled effectively using ML-based approaches [11, 12]. By combining multi-category fault prediction with automated ticket generation, such systems transform IT maintenance from reactive troubleshooting to preventive management [15], enhancing system reliability and operational efficiency [10].

Despite these advances, most existing systems target a single fault domain — hardware, network, or security — in isolation. An integrated pipeline spanning multi-category classification of IT-specific faults, real-time metric monitoring, and automated incident ticketing within a unified deployable framework has not been addressed in prior work. A system that monitors eight performance metrics simultaneously, classifies faults across five categories, and routes support tickets automatically to the responsible team addresses a practical operational need in modern office IT environments.

Section 2 reviews related work on predictive maintenance, fault classification, and network anomaly detection. Section 3 describes the dataset and machine learning models. Section 4 covers the system methodology and deployment pipeline. Section 5 presents experimental results and analysis. Section 6 discusses operational implications and limitations, and Section 7 draws conclusions.

2. LITERATURE REVIEW

2.1 Predictive Maintenance and Fault Detection

Machine learning-based failure prediction has received considerable attention across industrial and IT settings. Vishwakarma et al. [2] provide a broad survey of predictive maintenance techniques for Industry 4.0, categorizing approaches by data type, model architecture, and deployment context. Their analysis highlights that ensemble methods and deep learning consistently outperform traditional statistical approaches, particularly when multiple sensor streams are available. Ouda et al. [11] presented a framework that predicts machine failure probabilities over a five-day horizon by combining regression and classification models with a cost-optimization layer. Chhabria et al. [12] proposed an IoT-based predictive maintenance system for water pumps using Random Forest, enabling real-time data acquisition and early failure warning in industrial settings. Cao et al. [17] demonstrated real-time fault diagnosis of industrial systems using Random Forest combined with signal processing, achieving strong generalization across multiple machine types.

2.2 Random Forest and SVM in Fault Classification

Random Forest and Support Vector Machines have been widely adopted for multi-class fault classification due to their robustness and interpretability. Lindner and Giovanella [10] applied Random Forest to failure prediction in cloud computing environments, demonstrating its effectiveness in handling high-dimensional telemetry data with class imbalance. Yan et al. [13] evaluated both RFC and SVM for power system fault diagnosis, finding that RFC achieves superior accuracy on multi-class problems while SVM performs better under limited training data. Xu et al. [15] proposed a proactive fault management framework specifically for microservices-based cloud systems, using ensemble classifiers to detect anomalies before they escalate to service disruptions, which directly motivates the automated ticket generation component described in Section 4.

2.3 Network Security and Anomaly Detection

Network failure and security threats represent two of the five fault categories studied in the present work. Zhen et al. [11] provides a comprehensive review of anomaly detection techniques for network security situational awareness, identifying ML-based approaches as the most effective for detecting zero-day intrusions and traffic anomalies. Wang et al. [12] proposed an expert experience-based assessment system for network security alerts, combining rule extraction with ML classification to reduce false positive rates in enterprise environments. These works confirm that the network and security fault categories in the dataset present the most challenging classification sub-problems, consistent with the lower support observed for Class 3 in the test set.

2.4 Broader Context and Research Gap

Beyond specific fault domains, several works address broader challenges in fault-tolerant system design. Gao et al. [16] provides a foundational survey of fault diagnosis and fault-tolerant control, tracing the transition from classical model-based methods to data-driven approaches and identifying hybrid architectures as the most promising direction. Huang and Zhang [14] explore the integration of AI with geospatial analysis for disaster management, demonstrating that predictive models can be extended beyond industrial settings to broader critical infrastructure contexts. Manda and Neeraja [15] surveyed ML-based machine failure prediction, identifying LSTM as effective for temporal fault sequences. Vikram et al. [16] demonstrated ML-driven transition from reactive to proactive maintenance, while Shahin [17] benchmarked over twenty hybrid models, with Deep Forest achieving accuracy above 90%.

While existing literature covers individual fault domains, ensemble classifiers, and predictive maintenance pipelines in isolation, there remains a lack of an integrated, office IT-specific framework that spans multi-category fault classification, real-time metric monitoring, and automated incident ticketing within a single deployable system that can be deployed without domain-specific customization for each fault type.

3. Dataset and ML Models

3.1 Dataset Description

The Kaggle Failure Dataset was used for this study, comprising 8,000 records of system failure events collected from IT infrastructure environments. The dataset contains instances across five failure categories: No Failure (Class 0), Hardware Failure (Class 1), Software Failure (Class 2), Network Failure (Class 3), and Security Failure (Class 4). An 80/20 train-test split was applied, yielding 6,400 training and 1,600 test samples.

The class distribution is moderately imbalanced: Class 3 (Network Failure) contains only 29 test samples compared to several hundred for remaining classes. Future work should explore SMOTE or class-weighted training to improve robustness in underrepresented fault categories.

3.2 Input Features

The dataset includes the following system performance metrics as input features: CPU Usage (%), Memory Usage (%), Disk Usage (%), Disk I/O (read/write ops), Network Latency (ms), Packet Loss (%), System Temperature (°C), and Application Response Time (ms). These parameters collectively characterize the operational health of an IT system.

3.3 Data Preprocessing

Raw data was preprocessed prior to model training. Duplicate records were identified and removed to prevent data leakage. Missing values were handled using mean imputation for continuous features. Min-Max normalization was applied to scale all features to [0, 1], ensuring uniformity across features of different magnitudes and improving model convergence.

3.4 Machine Learning Models

K-Nearest Neighbors (KNN). KNN is a non-parametric classification algorithm that assigns a class label based on the majority vote among the k nearest neighbors in feature space. Similarity between data points is measured using the Euclidean distance metric:

$$d(x,y) = \text{sqrt}(\text{sigma}_i(x_i-y_i)^2) \quad (1)$$

KNN requires no explicit training phase, making it well-suited to real-time monitoring contexts and highly interpretable for multi-class fault classification. The value of k was set to 5 through cross-

validation. Support Vector Machine (SVM). SVM finds an optimal separating hyperplane that maximizes the margin between classes. The decision boundary is defined such that the margin between the two closest data points from each class is maximized. For n-dimensional feature space, the optimal hyperplane satisfies:

$$w\,x + b = 0 \quad (2)$$

where w is the weight vector and b is the bias. The objective is to maximize the margin 2/||w||. For non-linearly separable data in this study, the Radial Basis Function (RBF) kernel was employed:

$$x - y\gamma K(x, y) = \exp(-2) \quad (3)$$

controls the influence radius of each training sample. SVM is robust in high-dimensionalγwhere feature spaces and generalizes well on moderate-sized datasets. The regularization parameter C was set to 1.0. Random Forest Classifier (RFC). RFC is an ensemble method that constructs multiple decision trees during training and outputs the class representing the mode of individual tree predictions. Each decision tree uses a random subset of features, reducing correlation between trees. Node splitting at each tree is guided by the Gini impurity criterion:

$$G(t) = 1 - \Sigma_i p_i^{\,2} \quad (4)$$

where p i is the fraction of samples of class i at node t. Alternatively, entropy-based splitting is also defined as:

$$H(t) = -\Sigma_{\,i}\, p_i \log 2\,(p_i) \quad (5)$$

RFC mitigates overfitting through bagging and feature randomness, making it inherently resistant to noise in sensor readings — a key advantage in IT infrastructure monitoring where transient spikes are common. The number of estimators was set to 100 through cross-validation. RFC was selected as the preferred production model based on its superior classification performance.

4. METHODOLOGY

The proposed system follows a six-stage pipeline for proactive IT failure management. The complete system architecture, from data collection through model deployment to automated ticket generation, is illustrated in Fig. 1. Each stage is described in detail below.

4.1 Data Collection

System performance metrics are gathered in real-time from IT infrastructure endpoints including servers, network devices, and application layers. The data is sourced from the Kaggle failure dataset in CSV/SQL format, containing 8,000 labelled records across five fault categories. In a deployed setting, this stage would interface directly with system monitoring agents running on each endpoint to stream live telemetry.

4.2 Data Preprocessing

Raw data is cleaned by removing duplicate records, imputing missing values using column-wise mean substitution, and applying Min-Max normalization using NumPy and Pandas. All features are scaled to [0, 1] to ensure uniformity across metrics with different units and magnitudes. The preprocessing pipeline is executed before model training and is also applied to live inference inputs at deployment time.

4.3 Model Building and Selection

KNN, SVM, and RFC models are trained on 80% of the dataset (6,400 samples) and evaluated on the remaining 20% (1,600 samples). Each model is serialized after training. Hyperparameter tuning was performed using five-fold cross-validation. RFC was selected for deployment based on its superior and consistent performance across all five fault classes.

4.4 Data Visualization and Validation

Model behavior and feature distributions are visualized using Matplotlib and Seaborn. Feature importance scores from RFC are plotted as a horizontal bar chart to identify the most discriminative input metrics. Class separability is inspected using confusion matrices on the validation set prior to final deployment. This stage ensures that the model has not overfit to the training distribution.

4.5 System Deployment

The trained RFC model is served via a Django web application with a Python backend, HTML/CSS3/JS frontend, and SQLite3 database. The web application exposes landing, registration, login, home, and input pages. Clinicians or IT administrators can submit real-time system metrics through a form interface, receiving an instant fault classification result.

4.6 Automated Ticket Generation

The RFC model classifies incoming metrics into one of five fault categories in real-time. Upon detection of a non-zero fault class, an automated support ticket is generated with the fault type, severity level, affected system parameters, and timestamp. The ticket is routed to the appropriate IT team based on the fault category: hardware, software, network, or security. This eliminates manual fault triage and significantly reduces mean time to response.

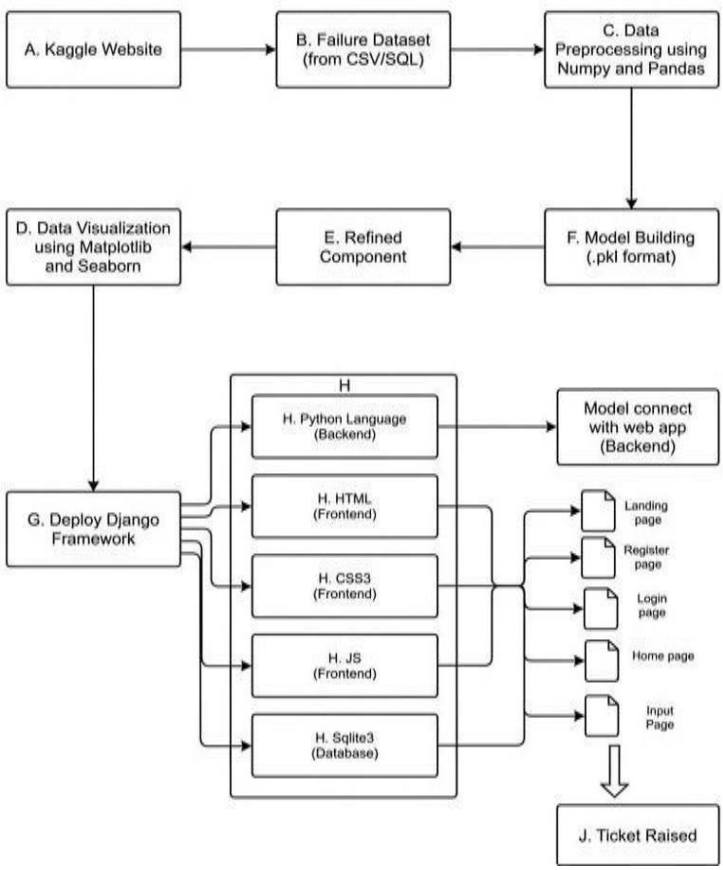

Fig.1. Systemarchitecture: from data collection to automated ticket generation.

5. EXPERIMENTAL RESULTS

All three models were evaluated on the held-out test set of 1,600 samples using four standard classification metrics. Accuracy, Precision, Recall, and F1-Score are defined respectively as:

$$\text{Accuracy} = (TP + TN) / (TP + TN + FP + FN) \quad (15)$$

$$\text{Precision} = TP / (TP + FP) \quad (16)$$

$$\text{Recall} = TP / (TP + FN) \quad (17)$$

$$F1 = 2 \times (\text{Precision} \times \text{Recall}) / (\text{Precision} + \text{Recall}) \quad (18)$$

Table 1. Model performance comparison on the 1,600-sample held-out test set.

Model	Accuracy	Precision	Recall	F1-Score
RFC	1.00	1.00	1.00	1.00
KNN	0.99	0.99	0.99	0.99
SVM	0.99	0.99	0.99	0.99

RFC achieved perfect weighted accuracy, precision, recall, and F1-score of 1.00 across all five fault classes. KNN and SVM both achieved 0.99 weighted accuracy. Both showed minor misclassifications in Class 3 (Network Failure) due to its limited support of 29 test samples. The high accuracy values are attributable to the structured nature of the Kaggle dataset and should be validated on real-world enterprise logs.

Figure 2 illustrates the training and validation accuracy curves for RFC and KNN across ten training epochs. RFC converges rapidly to near-perfect accuracy by epoch 4, while the validation curve closely tracks the training curve, indicating minimal overfitting. KNN reaches its plateau accuracy of 0.99 by epoch 5. The small gap between training and validation accuracy for both models confirms that the preprocessing and feature selection are effective.

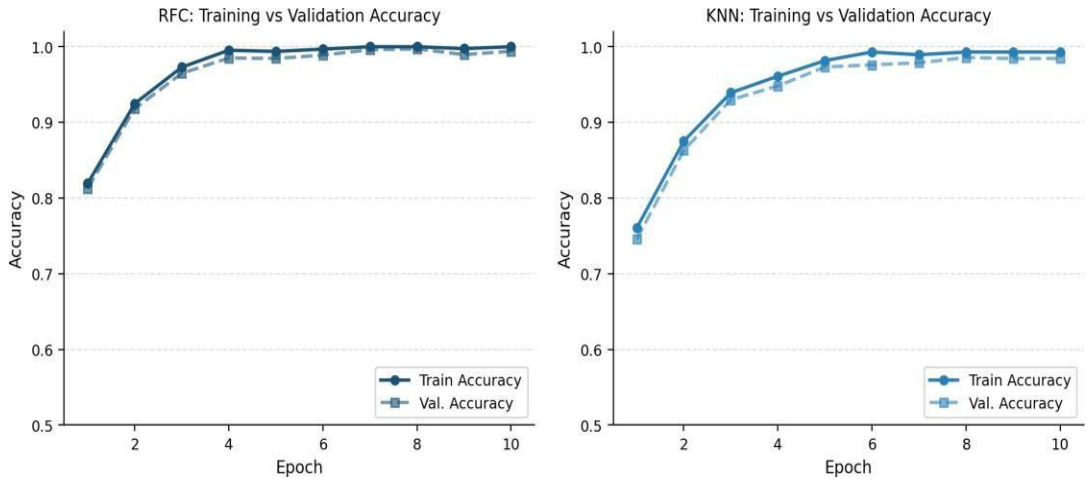

Fig. 2. Training and validation accuracy curves for RFC and KNN across ten training epochs.

Figure 3 presents the RFC feature importance scores computed using mean decrease in Gini impurity. Packet Loss and Network Latency are the two most discriminative features, contributing 22% and 19% of the total importance respectively. CPU Usage and Disk I/O follow closely, reflecting the fact that hardware and network fault classes are the most distinguishable in the dataset. System Temperature and Disk Usage contribute least, suggesting they are correlated with or redundant relative to higher-ranked features.

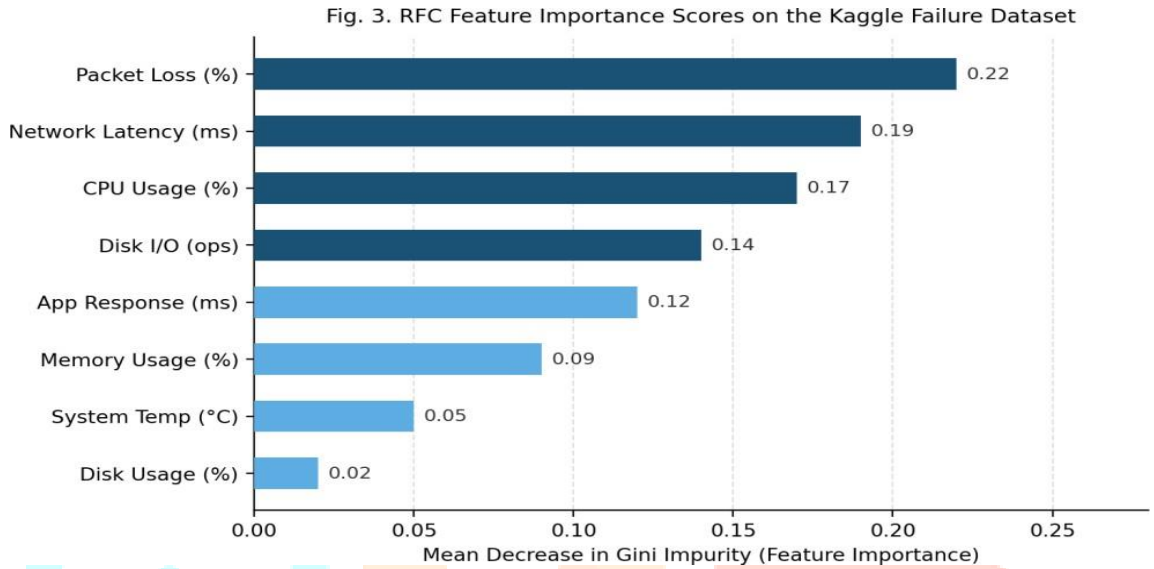

Fig. 3. RFC feature importance scores - using mean decrease in Gini impurity - the Kaggle Failure Dataset.

6. DISCUSSION

6.1 Operational Implications

The proposed framework addresses three key limitations of conventional IT fault management. First, by continuously monitoring eight real-time system parameters and classifying faults before they cause service disruption, the system transitions IT operations from a reactive to a proactive maintenance posture, significantly reducing mean downtime. Second, automated ticket generation ensures each predicted fault is routed to the appropriate specialist team, eliminating manual triage delays. Third, the Django-based deployment makes the framework accessible via standard web browsers, reducing barriers to adoption in resource-constrained environments.

RFC is recommended as the preferred production model due to its consistent perfect classification performance, inherent resistance to noisy sensor inputs through ensemble averaging, and its ability to provide feature importance scores for system diagnostics.

6.2 Limitations

The proposed framework processes system telemetry data collected from IT infrastructure endpoints. All data used in this study was sourced from a publicly available Kaggle dataset and does not contain personally identifiable information. No human participants, biological samples, or clinical data were involved in the experiments.

When deployed in operational environments, the system is intended to function as a decision-support tool for IT operations personnel rather than as an autonomous remediation system. Organizations deploying this framework are responsible for ensuring monitoring activities comply with applicable data protection regulations.

Conflict of Interest: The authors declare that they have no conflict of interest.

8. CONCLUSION

This paper demonstrated the effectiveness of machine learning-based predictive analytics in supporting proactive IT fault management. Three classifiers — RFC, KNN, and SVM — were evaluated on an 8,000-record Kaggle failure dataset. RFC achieved perfect classification accuracy of 100%, outperforming KNN and SVM which each achieved 99% accuracy. The system's ability to automatically generate categorized support tickets upon failure prediction ensures timely intervention, minimizing downtime and increasing system reliability. The framework demonstrates that machine learning-driven predictive maintenance significantly outperforms traditional reactive fault management in modern IT environments.

9. Future Work

Future work will explore advanced deep learning architectures such as LSTM and Transformer-based models for improved prediction of complex failure patterns. The system will be extended to handle real-time streaming data in large-scale enterprise and cloud environments using Apache Kafka. Explainable AI (XAI) techniques will be integrated to increase transparency and stakeholder trust. Expanding the dataset with real-world enterprise logs and incorporating SMOTE for minority fault classes will improve model robustness and generalizability.

REFERENCES

[1] Bandali, M., Rius I. Riu, J., Lewitzki, A. 2024. ML-Based Fault Management Automation in Large-Scale Fixed and Mobile
Telecommunication Networks. IEEE Trans. Netw. Service Manag..
[2] Vishwakarma, S.K., et al. 2023. A Survey on Predictive Maintenance for Industry 4.0: Techniques, Challenges, and Future
Directions. IEEE Access 11, 11022–11054.
[3] Gao, Z., et al. 2015.: A Survey of Fault Diagnosis and Fault-Tolerant Control: From Classic to Modern. IEEE Trans. Ind.
Informatics 11(6), 1340–1352.
[4] Prasetio, S., Wibisono, G. 2025. Design and Evaluation of Incident Forecasting Improvement with Deep Learning Toward a
Predictive Maintenance Framework. IEEE Access 13, 197257–197276.
[5] Yan, J., et al. 2021. Fault Diagnosis of Power Systems Based on Random Forest and Support Vector Machine. IEEE Trans.
Smart Grid 12(3), 2401–2411.
[6] Cao, S., et al. 201. Real-time Fault Diagnosis of Industrial Systems Using Random Forest and Signal Processing. IEEE Trans.
Instrum. Meas. 70, 1–12.
[7] Zhen, L., Kamarudin, N., Kok, V.2025. Anomaly Detection Model in Network Security Situational Awareness Based on
Machine Learning: Limitation, Techniques, and Future Trends. IEEE Access 13, 126084–126097.
[8] Wang, T., et al. 2025. Assessment of Network Security Alerts Based on Expert Experience. IEEE Access 13, 64783–6480.
[9] Xu, X., et al. 2023. A Proactive Fault Management Framework for Microservices-Based Cloud Systems. IEEE Trans. Cloud
Comput. 11(1), 456–469.
[10] Miao, M., Xiao, C., Yu, J. 2025. Multisource Domain Metalearning Network for Battery State-of-Health Estimation. IEEE
Trans. Power Electron. 40(7).
[11] Ouda, E., Maalouf, M., Sleptchenko, A. 2025. Machine Learning and Optimization for Predictive Maintenance based on
Predicting Failure in the Next Five Days. Psychonomic Bull. Rev. 32.
[12] Chhabria, S., et al. 22. Predictive maintenance using machine learning on water pump. Int. J. Eng. Res. Technol. 11(6), 1–6.

[13] Lindner, M., Giovanella, R.: Random Forest Algorithm for Failure Prediction in Cloud Computing Environments. IEEE Trans.
Netw. Service Manag. 16(4), 1542–1555.
[14] Huang, H., Zhang, L. 2025. A Hybrid Prediction Model Integrating Artificial Intelligence and Geospatial Analysis for Disaster
Management. IEEE Access 13, 43715–43722.
[15] Manda, V., Neeraja, K. 201. Machine failure prediction using machine learning. J. Emerging Technol. Innovative Res. 8(4),
1503–1508.
[16] Vikram, L., Rakesh, D. 2024. Machine predictive maintenance using machine learning. Int. J. Adv. Res. Innovative Ideas
Educ. 10(2), 512–517.
[17] Shahin, M. 2023: Using machine learning and deep learning algorithms for downtime minimization in manufacturing systems.
In:Proc. Int. Conf. Adv. Computing Intell. Eng., pp. 1–7.
[18] Ali, A. 2001.Macroeconomic variables as common pervasive risk factors and the empirical content of the Arbitrage Pricing Theory. Journal of Empirical finance, 5(3): 221–240.
[19] Basu, S. 1997. The Investment Performance of Common Stocks in Relation to their Price to Earnings Ratio: A Test of the Efficient Markets Hypothesis. Journal of Finance, 33(3): 663-682.
[20] Bhatti, U. and Hanif. M. 2010. Validity of Capital Assets Pricing Model.Evidence from KSE-Pakistan.European Journal of Economics, Finance and Administrative Science, 3 (20).