Volume 11, Issue 11, November – 2026 International Journal of Innovative Science and Research Technology
ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26nov055

Explainable Machine Learning for Predictive Maintenance in Industry 4.0

Tirunagiri Lokesh[1]

[1]Department of Computer Science and Engineering TKR College of Engineering and Technology Hyderabad, Telangana, India

Publication Date: 2026/11/27

Abstract: Predictive maintenance identifies potential equipment failures before they cause unplanned downtime. However, industrial applications require accurate and interpretable predictions. This study develops an explainable machine-learning prototype for binary machine-failure prediction using the AI4I 2020 Predictive Maintenance Dataset, containing 10,000 records, including 339 failures (3.39%) and 9,661 non-failures (96.61%). After removing identifiers and failure-mode columns, five numerical measurements and two one-hot-encoded product-type indicators were retained. The dataset was divided using a stratified 80:20 training-testing split. Four classifiers—Logistic Regression, Random Forest, Decision Tree, and Gradient Boosting—were evaluated. Gradient Boosting achieved the highest accuracy (98.60%), failure-class F1-score (0.7667), and ROC-AUC (0.9698). Its recall was 0.6765, detecting 46 of 68 failures while missing 22. Random Forest achieved 98.05% accuracy but lower recall (0.4706), demonstrating the limitations of accuracy for imbalanced datasets. SHAP explanations and tree-based feature importance highlighted torque, rotational speed, and tool wear as influential features. A Streamlit dashboard integrates sensor inputs, predictions, configurable thresholds, explanations, evaluation reports, and prediction history. Since the dataset is synthetic and sensor inputs are simulated, external validation is necessary before real-world deployment.

Keywords: *Predictive Maintenance; Explainable Artificial Intelligence; Machine Learning; Machine Failure Prediction; SHAP; AI4I 2020; Streamlit Dashboard.*

How to Cite: Tirunagiri Lokesh (2026) Explainable Machine Learning for Predictive Maintenance in Industry 4.0. *International Journal of Innovative Science and Research Technology*, 11(11), 1914-1927. https://doi.org/10.38124/ijisrt/26nov055

I. INTRODUCTION

➢ *Background*

Manufacturing equipment can experience wear, thermal stress, power-related problems, and other operating conditions that lead to failure. Reactive maintenance responds after a failure occurs, while preventive maintenance schedules interventions in advance. Predictive maintenance adds a data-driven layer: models analyse operating measurements and estimate whether a failure may occur, potentially helping maintenance teams prioritize inspection and reduce avoidable downtime.

Machine-learning models can represent nonlinear interactions among operating variables, but more complex models may be difficult to interpret. In an industrial setting, a prediction is more useful when a technician can inspect the factors associated with it rather than receiving only a binary "failure" or "normal" label. Explainable AI methods such as SHAP can help communicate how features contribute to a model output.

➢ *Project Motivation*

Modern industrial systems rely heavily on rotating machinery and other critical equipment, where unexpected failures can lead to production downtime, increased maintenance costs, equipment damage, and safety risks. Traditional maintenance approaches, such as reactive maintenance and fixed-schedule preventive maintenance, often fail to identify the actual condition of machinery accurately, resulting in unnecessary maintenance activities or unexpected breakdowns. Although machine learning techniques have demonstrated considerable potential for predicting equipment failures, their adoption in industrial environments remains challenging because many predictive models provide limited insight into the reasoning behind their predictions.

This project is motivated by the need to develop an intelligent, reliable, and interpretable predictive maintenance system that not only identifies potential machine failures but also explains the factors contributing to those predictions. By applying and comparing multiple machine learning algorithms, the proposed approach aims to evaluate failure prediction performance using appropriate classification metrics, with particular emphasis on detecting failure cases in an imbalanced dataset. Explainable AI techniques, particularly SHAP (Shapely Additive explanations), are incorporated to identify influential operational parameters and provide insights into individual predictions, helping

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26nov055

bridge the gap between predictive accuracy and human understanding.

Furthermore, the integration of the trained models with an interactive dashboard enables users to examine sensor measurements, obtain failure predictions, understand contributing factors, and receive visual indications of potential risks. By combining predictive analytics with explainability and an accessible monitoring interface, this research aims to support more informed maintenance decisions, improve the interpretability of machine learning outputs, and contribute to reducing avoidable equipment downtime. Ultimately, the project seeks to demonstrate how explainable machine learning can make predictive maintenance systems more transparent, practical, and useful for data-driven industrial decision-making.

II. RELATED WORK AND RESEARCH GAP

➢ *Predictive Maintenance Data and Benchmark*

Matzka introduced the AI4I 2020 predictive-maintenance dataset alongside an explainable-AI application for predictive maintenance. The dataset was designed to resemble industrial operating data while remaining shareable; the UCI repository explicitly identifies it as synthetic and documents 10,000 instances. This makes it useful for reproducible experiments, although it cannot by itself establish performance on physical equipment.

➢ *Explainable Machine Learning*

Lundberg and Lee introduced SHAP as a unified framework for attributing model output to input features. SHAP values provide a consistent way to examine the direction and magnitude of feature contributions, while global summary plots aggregate explanations across observations. Tree-based models also provide built-in feature-importance scores, but these are not identical to SHAP values: impurity-based importance summarises split-based contributions, whereas SHAP estimates feature contributions to individual predictions and can be aggregated globally.

➢ *Model Families Used in this Project*

Logistic Regression provides a linear classification baseline; standardisation is applied in the supplied pipeline. Decision Trees learn rule-like partitions of the feature space. Random Forest aggregates multiple decision trees to reduce variance. Gradient Boosting builds an additive ensemble sequentially, fitting new learners to improve the current model. Comparing models is appropriate because their bias, variance, interpretability, and sensitivity to nonlinear interactions differ.

➢ *Research Gap and Project Position*

The project is not claiming a new learning algorithm. Its practical contribution is an integrated educational prototype that compares common classifiers, exposes the failure-class trade-offs, adds SHAP-based global interpretation, and presents outputs through a dashboard. A further research gap for a production-oriented extension is validation on real machine telemetry, including temporal generalisation, sensor drift, calibrated probabilities, and operational cost analysis.

Although machine learning has demonstrated potential for predictive maintenance, several practical challenges remain. Machine failures are relatively uncommon compared with normal operating conditions, making it difficult to evaluate models using accuracy alone. A model may achieve high overall accuracy while failing to identify a substantial proportion of actual failures. Furthermore, complex machine learning models may provide predictions without sufficiently explaining the factors influencing their decisions. These limitations highlight the need for a predictive maintenance framework that combines reliable failure detection, comprehensive performance evaluation, model explainability, and an accessible interface for inspecting predictions.

A. Problem Definition and Objectives

➢ *Problem Statement*

Unexpected machine failures can cause production downtime, increased maintenance costs, and reduced industrial efficiency. Although machine learning techniques offer promising solutions for predictive maintenance, imbalanced datasets can lead to high classification accuracy while failing to detect actual failures. Furthermore, the limited interpretability of complex models makes it difficult for maintenance personnel to understand the factors influencing predictions and use them confidently in decision-making.

This study addresses these challenges by developing an explainable machine learning framework using the AI4I 2020 Predictive Maintenance dataset. It compares Logistic Regression, Decision Tree, Random Forest, and Gradient Boosting using multiple evaluation metrics, incorporates feature-importance analysis and SHAP explanations, and integrates predictions into a dashboard. The objective is to combine failure detection, model interpretability, and accessible prediction outputs to support more informed predictive maintenance decisions.

➢ *Objectives*

- Dataset Inspection and Preprocessing: To analyze the AI4I 2020 Predictive Maintenance dataset for missing values, duplicate records, class imbalance, outliers, and feature relationships, while preparing the data through appropriate feature selection, categorical encoding, and a consistent training and testing split.
- Machine Learning Model Development and Comparison: To train and compare four supervised binary classification algorithms—Logistic Regression, Decision Tree, Random Forest, and Gradient Boosting—to predict machine failures based on operational sensor measurements.
- Comprehensive Model Evaluation: To assess model performance using accuracy, precision, recall, F1-score, ROC-AUC, and other relevant metrics, with particular

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26nov055

emphasis on minority-class failure detection and balancing false alarms against missed failures.

- Explainable AI and Model Interpretation: To employ SHAP and feature importance techniques to identify influential operational parameters, explain their contributions to individual predictions, and interpret overall model behaviour, thereby improving prediction transparency and supporting informed maintenance decisions.
- Interactive Dashboard and Decision Support: To develop a Streamlit-based dashboard integrating trained models, simulated sensor inputs, failure predictions, explainability insights, evaluation reports, and prediction history, demonstrating a unified and interpretable predictive maintenance system for supporting proactive equipment monitoring.

III. DATASET AND EXPLORATORY DATA ANALYSIS

➢ *Dataset Overview*

The project uses the AI4I 2020 Predictive Maintenance Dataset from the UCI Machine Learning Repository. The supplied CSV contains 10,000 rows and 14 columns. Initial inspection reports no missing values and no duplicate rows. The dataset contains identifiers, product type, five operational measurements, the composite machine-failure label, and five failure-mode indicators. The overall dataset profile and split distribution are summarized in Table 1.

Table 1 Dataset Profile and Train/Test Split Summary.

Property	Observed value
Dataset rows	10,000
Original columns	14
Missing values	0
Duplicate rows	0
No-failure records	9,661 (96.61%)
Failure records	339 (3.39%)
Train/test split	80% / 20%, stratified
Training/test rows	8,000 / 2,000

➢ *Variables*

The 14 original dataset variables, their physical meanings, and their handling during preprocessing are detailed in Table 2.

Table 2. Dataset Variables, their Meanings, and their use in the Final Model.

Variable	Meaning	Use in final model
UDI	Unique record identifier	Removed
Product ID	Product identifier	Removed
Type	Product type: H, L, M	Encoded as Type_L and Type_M; H is reference category
Air temperature [K]	Ambient air temperature in kelvin	Retained
Process temperature [K]	Process temperature in kelvin	Retained
Rotational speed [rpm]	Rotational speed in revolutions per minute	Retained
Torque [Nm]	Torque in newton-metres	Retained
Tool wear [min]	Tool wear in minutes	Retained
Machine failure	Composite binary target	Prediction target
TWF, HDF, PWF, OSF, RNF	Failure-mode indicators	Excluded from predictors to reduce target leakage

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26nov055

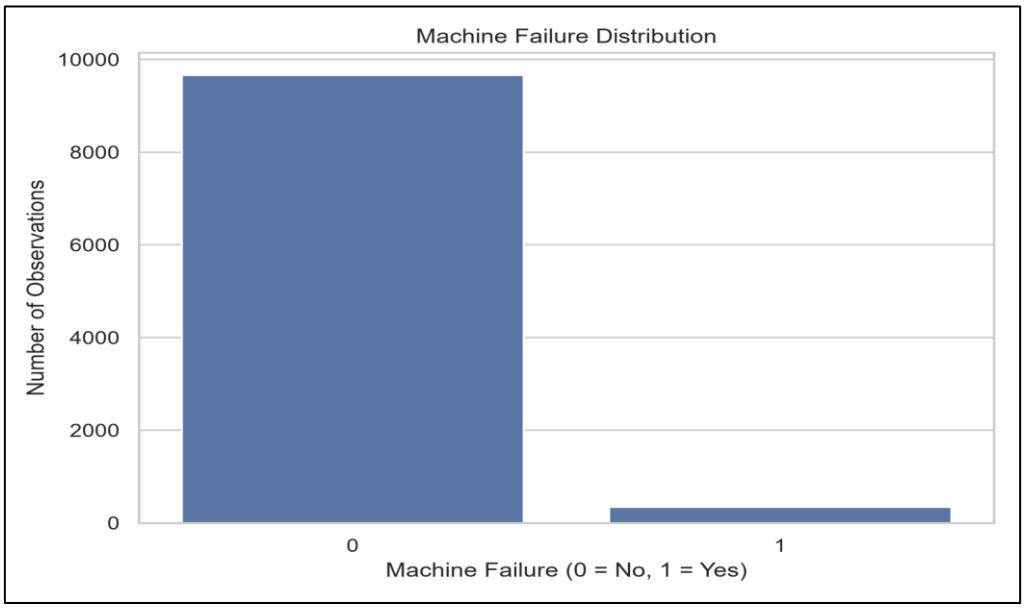


Fig 1 Distribution of the Binary Machine-Failure Target.

Figure 1 shows the distribution of the binary machine-failure target, which is strongly imbalanced: failures account for 3.39% of observations, while 96.61% are normal. A classifier that always predicts "no failure" would achieve 96.61% accuracy while detecting no failures at all. This is why failure-class recall, precision, F1-score, and confusion-matrix counts are necessary alongside accuracy.

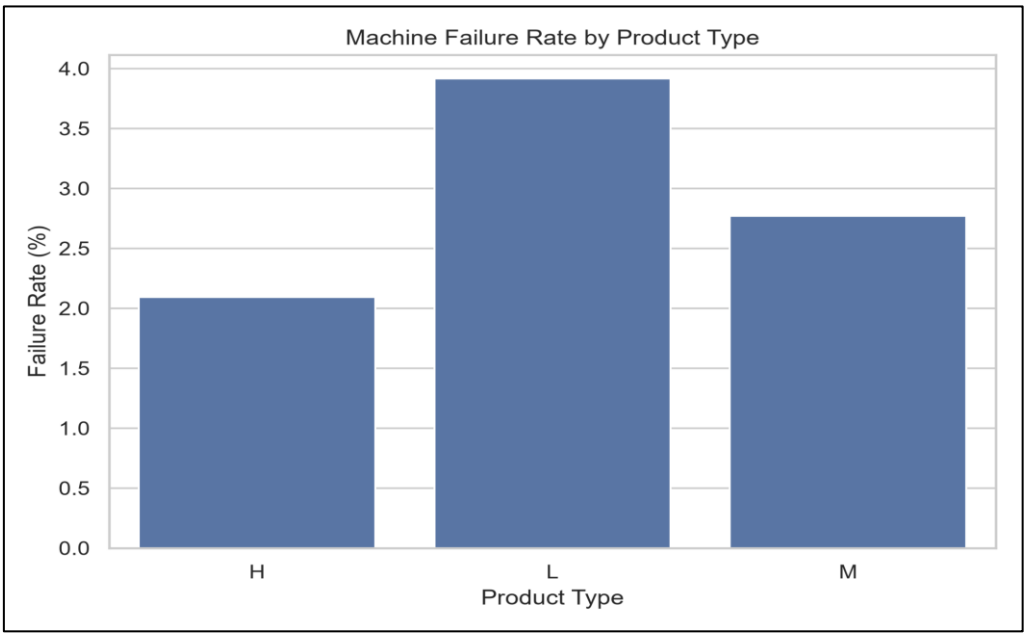


Fig 2 Observed Failure Rate by Product Type.

Figure 2 shows the observed failure rate for each product type (H, L and M). The EDA report gives failure rates of 2.09% for type H, 3.92% for type L, and 2.77% for type M. These are descriptive rates in this synthetic dataset; they should not be interpreted as evidence that product type itself causes failure.

➢ *Operating Measurements by Failure Class*

Figures 3 to 7 compare the distribution of each operating measurement for records labelled as normal and failure: torque (Figure 3), rotational speed (Figure 4), tool wear (Figure 5), air temperature (Figure 6) and process temperature (Figure 7). They show that torque and tool wear differ visibly between the two classes, while temperature distributions overlap substantially. Rotational speed also differs between the classes. These are descriptive patterns and do not prove causation.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26nov055

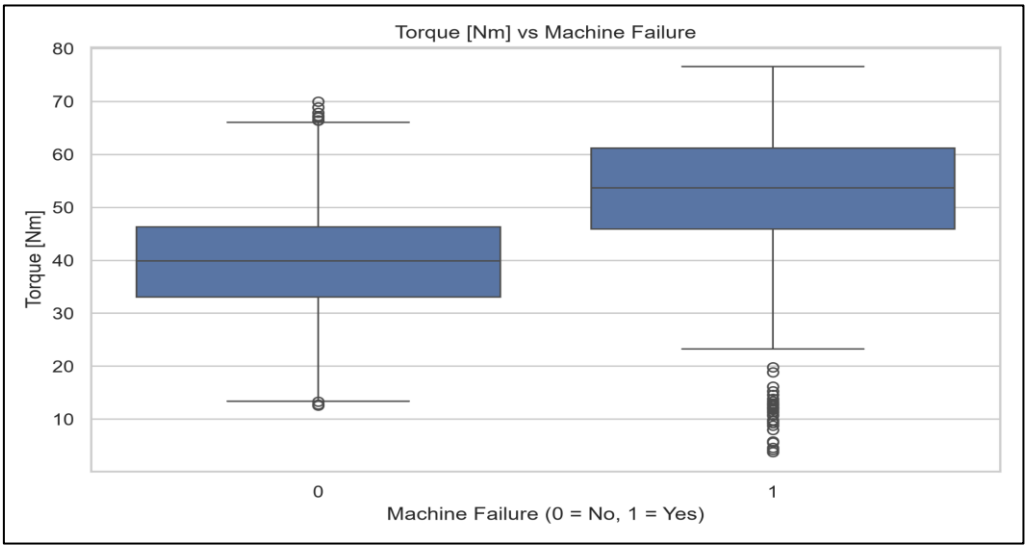


Fig 3 Torque Distribution by Machine-Failure Label.

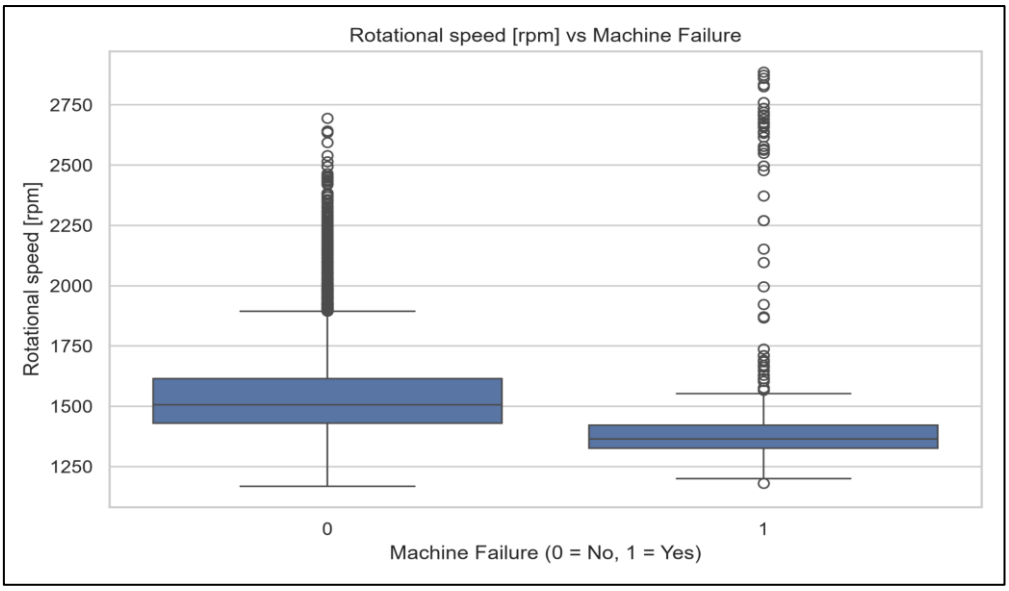


Fig 4 Rotational Speed Distribution by Machine-Failure Label.

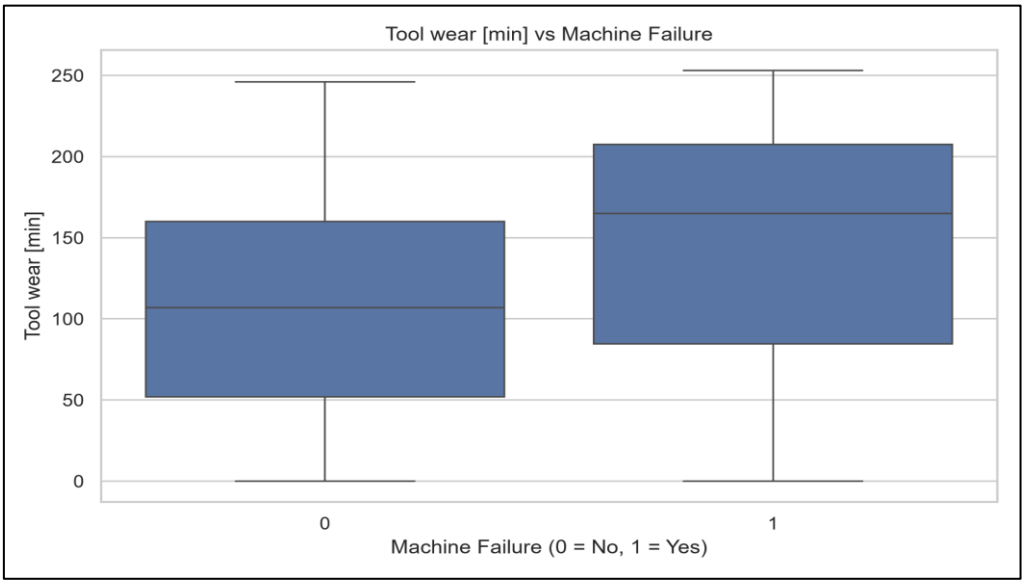


Fig 5 Tool Wear Distribution by Machine-Failure Label.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26nov055

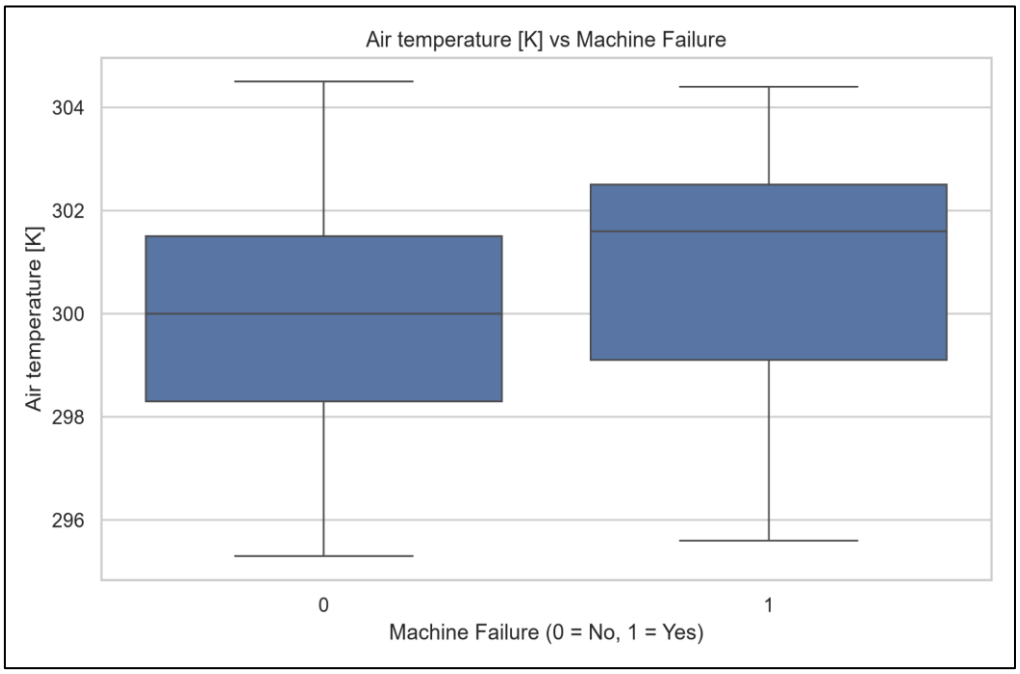


Fig 6 Air Temperature Distribution by Machine-Failure Label.

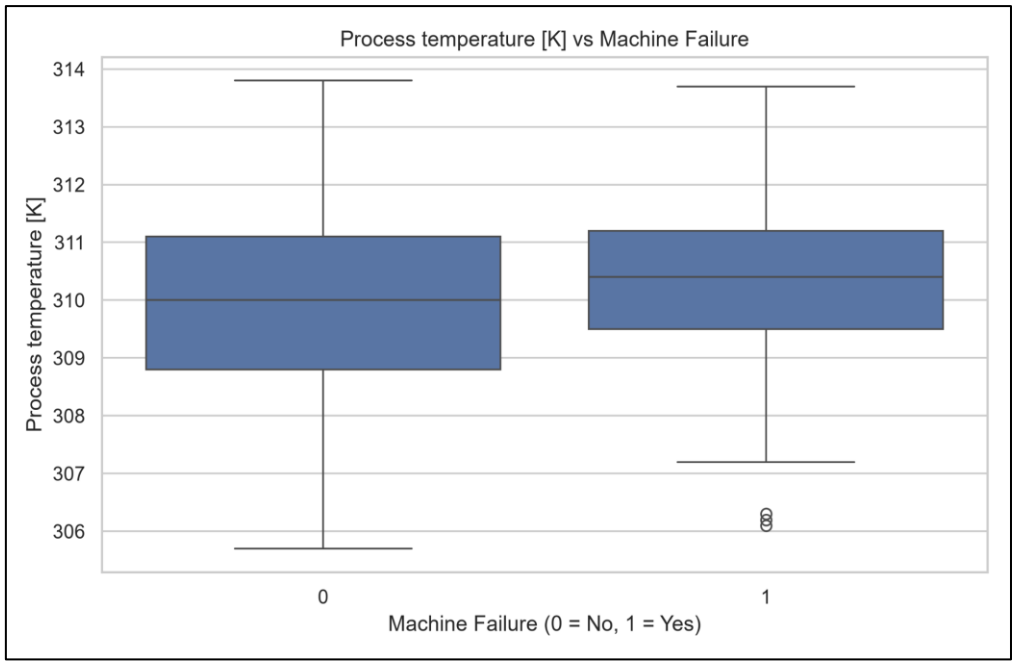


Fig 7 Process Temperature Distribution by Machine-Failure Label.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26nov055

➢ *Correlation Analysis*

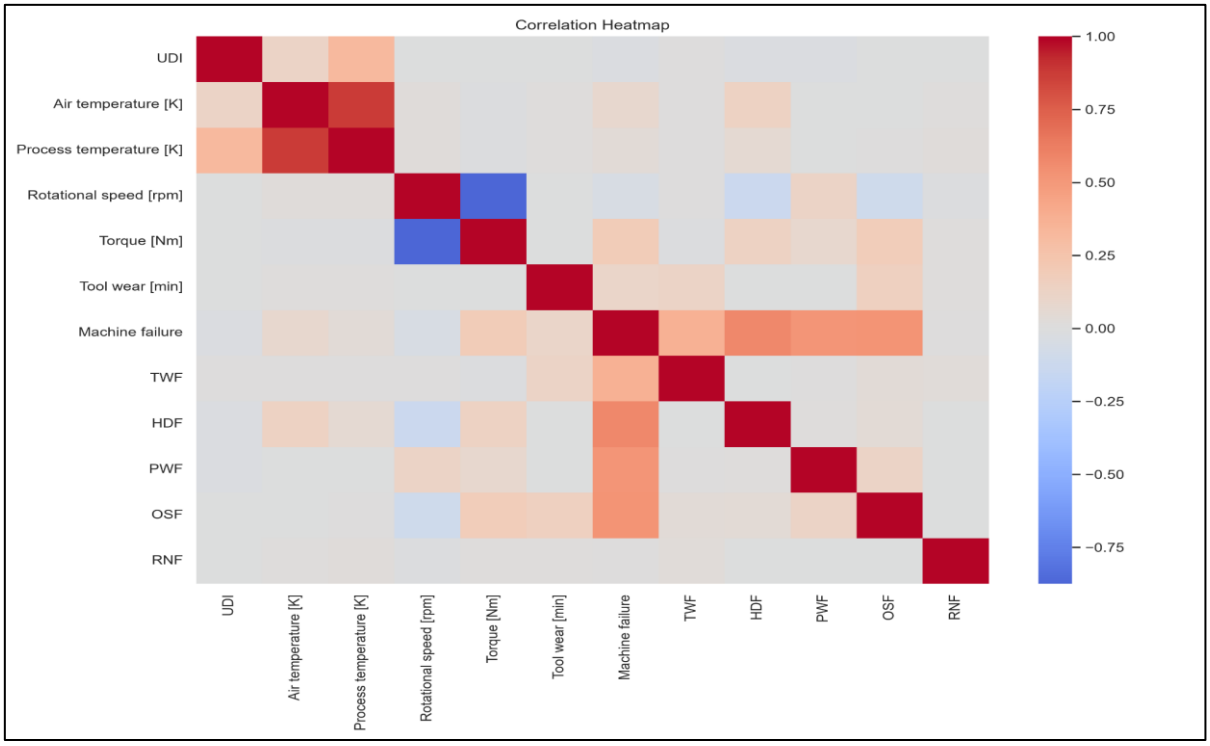


Fig 8 Correlation Heatmap for Numeric Dataset Columns.

Figure 8 presents the correlation heatmap for the numeric dataset columns. The EDA text report lists correlations with the composite failure label of 0.191 for torque, 0.105 for tool wear, 0.083 for air temperature, 0.036 for process temperature, and -0.044 for rotational speed. These pairwise linear correlations are modest and do not capture nonlinear interactions. The failure-mode columns have stronger correlations with the composite label because the composite target is defined using those failure indicators. For that reason, the mode labels are excluded from the predictive feature set; using them as inputs would leak target-related information.

➢ *EDA Summary*

- The data are complete according to the supplied checks, with no missing values or duplicate rows.
- The failure class is rare, making class-sensitive evaluation essential.
- Torque, tool wear, and rotational speed appear informative in the visual analysis and later model explanations.
- Product type L has the highest observed failure rate among the three types in this dataset.
- Correlation alone is insufficient to rank features for a nonlinear classifier.

IV. DATA PREPROCESSING

➢ *Identifier Removal*

UDI and Product ID are removed because they identify records or products rather than representing the selected operational condition measurements. Retaining arbitrary identifiers can let a model exploit non-generalising patterns.

➢ *Target and Leakage Control*

The target is Machine failure. The five columns TWF, HDF, PWF, OSF, and RNF describe specific failure modes associated with the composite failure label. They are excluded from X so that the model predicts failure from operating variables and product type rather than reading related failure labels directly. This is a key design decision for a meaningful predictive experiment.

➢ *Categorical Encoding*

The Type column has categories H, L, and M. The preprocessing script applies one-hot encoding with drop_first=True, resulting in Type_L and Type_M. Type H is represented when both indicators are zero. This gives seven predictors: five numerical measurements and two binary product-type indicators.

➢ *Train/Test Partition*

The preprocessing script uses train_test_split with test_size=0.20, random_state=42, and stratify=y. The resulting shapes are 8,000 training rows and 2,000 testing rows, with the target ratio preserved approximately across partitions. The saved test set contains 68 failures and 1,932 non-failures. A summary of the complete preprocessing pipeline and its rationale is provided in Table 3.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26nov055

Table 3 Preprocessing Pipeline and Rationale for Each Step.

Preprocessing step	Implementation	Rationale
Drop identifiers	Remove UDI and Product ID	Avoid relying on record identity
Define target	Machine failure	Binary prediction objective
Exclude failure-mode columns	TWF, HDF, PWF, OSF, RNF excluded from X	Reduce target leakage
Encode product type	One-hot encoding; drop_first=True	Convert category into numeric inputs
Split data	80:20, stratified, random_state=42	Hold-out evaluation and class-ratio preservation
Feature scaling	StandardScaler in Logistic Regression pipeline only	Support scale-sensitive linear baseline

V. PROPOSED METHODOLOGY AND SYSTEM ARCHITECTURE

➢ *End-to-end Workflow*

- Load the AI4I 2020 CSV and inspect data quality.
- Perform exploratory analysis and save plots and findings.
- Remove identifiers and failure-mode labels; encode product type.
- Create stratified training and test partitions.
- Train four supervised classifiers and save model artifacts.
- Evaluate predictions and compare class-sensitive metrics.
- Generate Random Forest feature importance and SHAP plots.
- Load a saved model in the Streamlit application and present predictions, reports, and history.

➢ *Logical Architecture*

Table 4 Logical System Architecture Across the Five Functional Layers.

Layer	Components	Output
Data layer	AI4I 2020 CSV; processed train/test CSVs	Source and prepared data
Preprocessing layer	Identifier removal, target separation, one-hot encoding, stratified split	Seven model features and binary target
Model layer	Logistic Regression, Random Forest, Decision Tree, Gradient Boosting	Trained classifiers saved as joblib files
Evaluation layer	Accuracy, precision, recall, F1, ROC-AUC, confusion matrix, threshold comparisons	CSV reports and metrics
Explainability layer	Random Forest feature_importances_ and SHAP TreeExplainer	Global importance CSV and figures
Presentation layer	Streamlit pages, simulated inputs, threshold slider, reports and history	Interactive prototype for users

The logical system architecture across its five functional layers is outlined in Table 4. The workflow is a batch-trained, locally served prototype. The model is trained by separate scripts; the dashboard loads saved artefacts. The supplied application includes simulated sensor input/refresh functionality, a configurable alert threshold, an explanation page, report downloads, and history review. It should not be described as a deployed IoT pipeline because the supplied files do not establish live hardware connectivity.

A. Model Development and Evaluation

➢ *Algorithms*

- *Logistic Regression*

Logistic Regression is used as a baseline. The supplied training script wraps StandardScaler and LogisticRegression in a pipeline, sets class_weight="balanced", max_iter=1000, and random_state=42. The balanced class weights increase the penalty associated with minority-class errors. The model has high failure recall but low precision in the saved hold-out evaluation, producing many false alarms.

- *Random Forest*

Random Forest combines predictions from multiple decision trees. The supplied main training script uses 200 trees, class_weight="balanced", random_state=42, and n_jobs=-1. Its default-threshold results show very high precision and specificity but miss more than half of the actual failures, so its high accuracy does not make it the best option for every maintenance objective.

- *Decision Tree*

A Decision Tree creates feature-based decision rules. The saved comparison shows an intermediate trade-off between precision and recall. The tree is potentially easier to inspect than a large ensemble, although the saved metrics alone do not establish that its explanations are more useful to technicians.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26nov055

- *Gradient Boosting*

Gradient Boosting adds weak learners sequentially to improve predictions. In the supplied four-model comparison, it gives the strongest default-threshold F1-score and ROC-AUC, with recall higher than Random Forest and precision still high. This makes it the strongest overall candidate among the compared models for the selected hold-out metrics, although the operational threshold should be chosen using the relative costs of false negatives and false positives.

➢ *Evaluation Metrics*

The evaluation metrics used to assess imbalanced classification performance and their interpretation cautions are listed in Table 5.

Table 5 Evaluation Metrics for Imbalanced Classification and Interpretation Cautions.

Metric	Meaning for this task	Interpretation caution
Accuracy	Fraction of all records classified correctly	Can look high when normal records dominate
Precision	Fraction of predicted failures that are actual failures	Low precision means more false alarms
Recall / sensitivity	Fraction of actual failures detected	Low recall means more missed failures
F1-score	Harmonic mean of precision and recall	Balances precision and recall but not business cost
ROC-AUC	Ranking ability across thresholds	Does not select an operational threshold
Specificity	Fraction of normal records correctly identified	High specificity can coexist with poor failure recall
MCC	Correlation between binary predictions and truth	Uses all cells of the confusion matrix

➢ *Main Hold-Out Results*

Table 6. Hold-Out Test Performance of the Four Classifiers.

Model	Accuracy	Failure precision	Failure recall	Failure F1	ROC-AUC	False alarms	Missed failures	Detected failures
Gradient Boosting	98.60%	88.46%	67.65%	76.67%	0.9698	6	22	46
Decision Tree	97.90%	72.41%	61.76%	66.67%	0.8047	16	26	42
Random Forest	98.05%	91.43%	47.06%	62.14%	0.9619	3	36	32
Logistic Regression	82.45%	14.18%	82.35%	24.19%	0.9070	339	12	56

The primary hold-out test results across all four classifiers are presented in Table 6. The primary test set contains 68 failure cases and 1,932 normal cases. Gradient Boosting detects 46 failures and misses 22, while raising 6 false alarms. Random Forest raises only 3 false alarms but detects 32 failures and misses 36. Logistic Regression detects 56 failures but generates 339 false alarms, producing poor precision. The preferred model therefore depends on the maintenance team's relative tolerance for missed failures and unnecessary inspections.

➢ *Confusion matrix for Random Forest*

Table 7 Confusion Matrix for the Random Forest Classifier at the Default 0.50 Threshold.

Actual class	Predicted normal	Predicted failure
Actual normal	1,929	3
Actual failure	36	32

The detailed confusion matrix for the Random Forest classifier at the default 0.50 threshold is shown in Table 7. The Random Forest confusion matrix confirms a conservative positive prediction pattern: only three normal records are incorrectly flagged, but 36 of 68 actual failures are missed. In predictive maintenance, false negatives can be costly, so this trade-off should be explicitly considered.

➢ *Threshold Comparison*

The supplied threshold reports demonstrate that the probability threshold controls the precision–recall trade-off. Lowering the threshold tends to detect more failures but also increases false alarms. The Random Forest threshold comparison is shown below. These results are based on the saved threshold report and should not be conflated with the separate internal-validation experiment. The Random Forest threshold comparison is shown in Table 8.

Table 8 Random Forest Performance at Different Decision Thresholds.

Threshold	Accuracy	Precision	Recall	F1-score	False alarms	Missed failures	Detected failures
0.50	98.05%	91.43%	47.06%	62.14%	3	36	32
0.40	98.35%	84.31%	63.24%	72.27%	8	25	43
0.30	98.40%	79.03%	72.06%	75.38%	13	19	49
0.20	97.45%	59.55%	77.94%	67.52%	36	15	53
0.10	95.30%	40.97%	86.76%	55.66%	85	9	59

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26nov055

At threshold 0.30, the Random Forest F1-score rises from 0.6214 to 0.7538 and recall rises from 0.4706 to 0.7206, while false alarms increase from 3 to 13. At threshold 0.10, recall reaches 0.8676 but false alarms rise to 85 and precision falls to 0.4097.

B. Explainable AI Analysis

➢ *Random Forest Feature Importance*

Figure 9 visualizes these Random Forest importance values as a bar chart. The project calculates the fitted Random Forest model's feature importance values and sorts the features by importance. The three largest values are torque (0.3111), rotational speed (0.2908), and tool wear (0.2079). These are relative model-based importance scores, not causal effects and not probabilities of failure. The full feature-importance rankings are listed in Table 9.

Table 9 Random Forest Feature-Importance Rankings.

Rank	Feature	Random Forest importance
1	Torque [Nm]	0.3111
2	Rotational speed [rpm]	0.2908
3	Tool wear [min]	0.2079
4	Air temperature [K]	0.1051
5	Process temperature [K]	0.0678
6	Type_L	0.0097
7	Type_M	0.0076

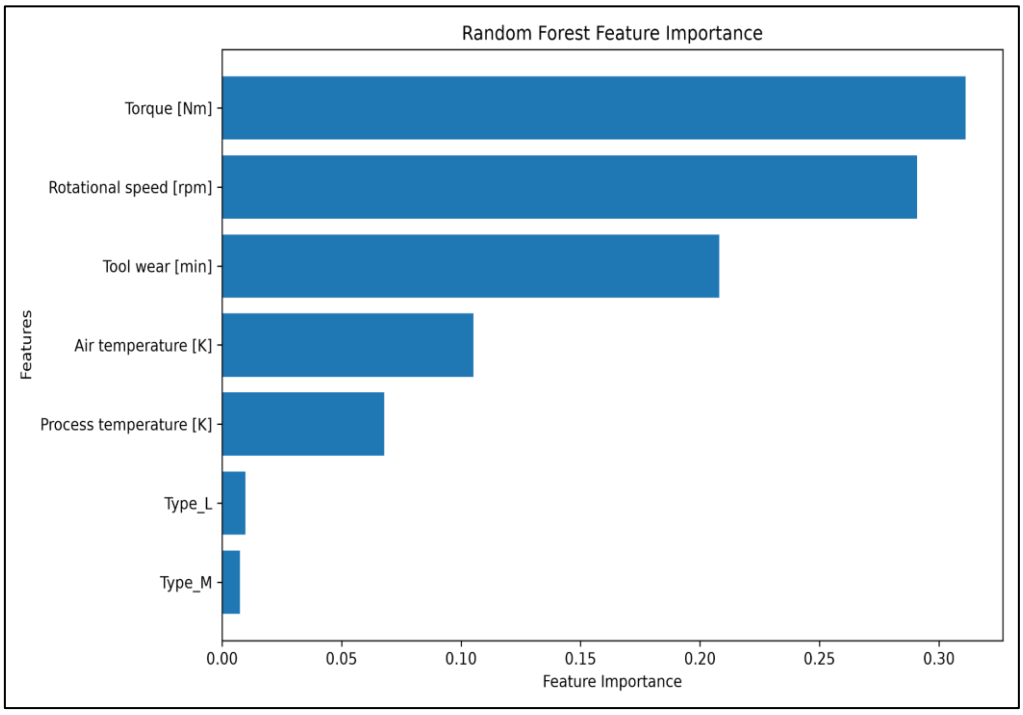


Fig 9 Random Forest Feature-Importance Bar Chart.

➢ *SHAP Global Importance*

Figure 10 ranks the features by mean absolute SHAP value. The supplied SHAP importance plot ranks features by mean absolute SHAP value, which summarises the average magnitude of each feature's contribution to the model output across the evaluated records. Torque ranks first, followed by rotational speed, tool wear, air temperature, process temperature, Type_L, and Type_M. This broadly agrees with the built-in Random Forest importance ranking.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26nov055

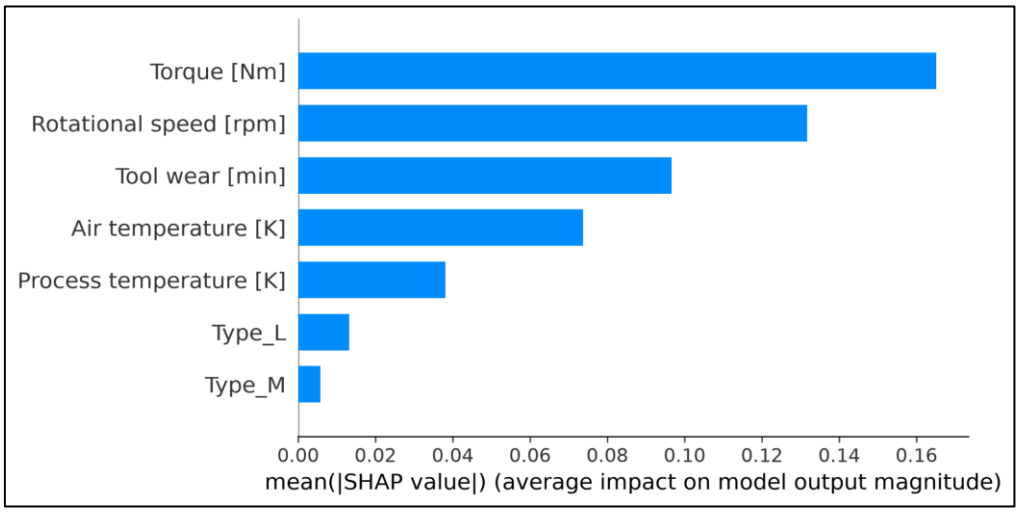


Fig 10 Global SHAP Importance Using Mean Absolute SHAP Values.

➢ *SHAP Summary Plot*

Figure 11 shows the SHAP summary (beeswarm) plot for the Random Forest model. The SHAP beeswarm plot shows each observation's SHAP value for each feature. Points to the right of zero increase the model output for the explained class, while points to the left decrease it. Point colour encodes the feature value from low (blue) to high (pink/red). The spread indicates that the direction and magnitude of a feature's contribution vary across records; this is more informative than treating a feature as having one fixed effect.

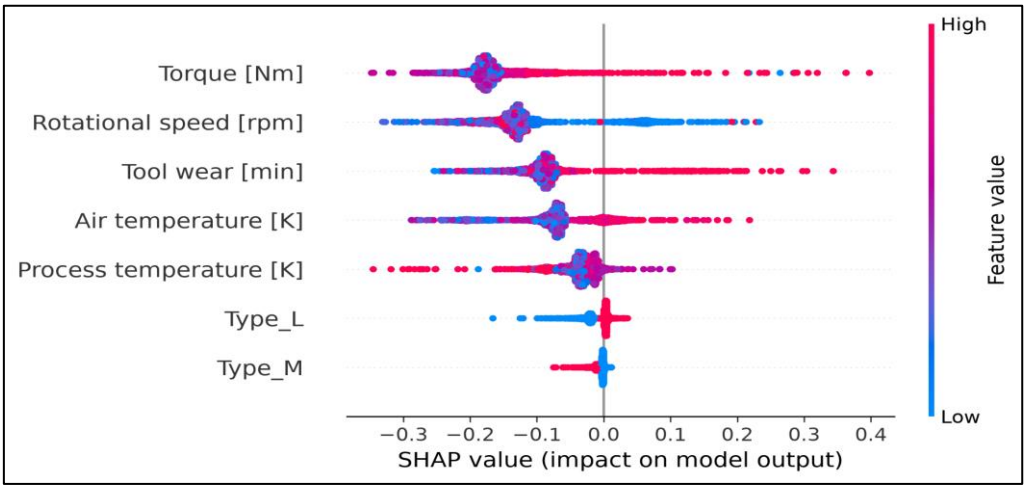


Fig 11 SHAP Summary (Beeswarm) Plot for the Random Forest Model.

In the supplied plot, high torque values are associated with positive SHAP values for many observations, while low torque values often appear on the negative side. Tool wear also shows a broad positive tail for high values, indicating that higher wear can raise the model output for some records. Rotational speed displays both positive and negative contributions depending on the value and context. These are associations learned by the model on this dataset, not causal conclusions. SHAP values depend on the fitted model, background/distribution assumptions, and class-output conventions.

➢ *Comparing Feature-Importance Approaches*

A comparative overview of the feature-importance and explainability methods used in this study is presented in Table 10.

Table 10 Comparison of Feature-Importance and Explainability Methods.

Approach	What it summarises	Main strength	Limitation
Random Forest feature_importances_	Relative split-based importance across the forest	Fast, directly available from tree ensemble	Can favour features with certain distributions/cardinalities; no per-row direction
Mean absolute SHAP	Average magnitude of feature contributions	Global ranking grounded in local explanations	Magnitude alone does not show direction

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26nov055

SHAP beeswarm	Per-row contribution direction and spread	Shows heterogeneity and high/low feature patterns	Requires careful class/output interpretation; not causal evidence

C. Dashboard Implementation

➢ *Purpose and Pages*

The supplied Streamlit application is organised into five pages: Live Dashboard, Model Evaluation, Explainable AI, Reports & History, and About Project. The sidebar includes a toggle for auto-refreshing simulated sensor values, a refresh-interval control, and a failure-alert threshold slider. The dashboard uses a saved model artefact to generate predictions from the seven model inputs. The purpose and operational scope of each dashboard component are summarized in Table 11.

Table 11 Purpose and Operational Scope of the Dashboard Components.

Dashboard component	Purpose	Evidence/qualification
Live Dashboard	Collect or refresh example operating readings; display predicted class and risk score	Sensor values are simulated, not received from physical hardware
Threshold control	Adjust cutoff applied to the predicted failure probability	Changing threshold changes labels and trade-offs
Model Evaluation	Display saved model evaluation outputs	Results depend on the saved CSV/model version
Explainable AI	Display feature-importance and SHAP visualisations	The supplied plots are global explanations; not necessarily a fresh local explanation per input
Reports & History	Review/download reports and prediction history	History relies on local CSV storage
Email alert option	Send notification using configured SMTP credentials	Requires correct SMTP settings and successful connectivity

➢ *Prediction Flow*

Read the user-entered or simulated values for the five numerical features and product type.

- Convert product type to the same Type_L and Type_M representation used during training.
- Pass the seven-feature row to the loaded model.
- Obtain a predicted class and, where supported, the model-estimated probability for class 1.
- Apply the selected threshold when the application's prediction logic uses probability thresholding.
- Display the result and append a record to local prediction history; optional email alerting requires configured SMTP settings.

➢ *Dashboard Safety and Usability*

A failure probability should be described as a model-estimated score, not as a calibrated real-world probability unless calibration has been tested. Likewise, a dashboard alert should prompt inspection rather than automatically shut down machinery. For real deployment, the application would need authenticated access, validated input ranges, logging and data-integrity checks, secure credential management, alert delivery monitoring, and human review procedures.

VI. RESULTS AND DISCUSSIONS

➢ *Main Findings*

- Gradient Boosting is the strongest overall model in the supplied default-threshold hold-out comparison by accuracy (98.60%), failure F1-score (0.7667), and ROC-AUC (0.9698).
- Random Forest has higher failure precision (91.43%) and fewer false alarms than Gradient Boosting, but lower recall and more missed failures.
- Logistic Regression has relatively high recall (82.35%) but very low precision (14.18%) because it flags many normal records as failures.
- Threshold changes materially affect the operating trade-off; no single threshold is optimal without knowing the cost of missed failures and false alarms.
- Torque, rotational speed, and tool wear dominate both Random Forest feature importance and SHAP global importance.

➢ *Why Accuracy is not Enough*

The majority class comprises 96.61% of the full dataset. A model can obtain a high accuracy by predicting normal for most observations and still fail to detect a substantial proportion of actual failures. Gradient Boosting's failure recall of 67.65% means that approximately one-third of failures in the test set were missed at the default threshold. This is an important residual risk even though the aggregate accuracy is high.

➢ *Threshold and Maintenance Trade-offs*

If missed failures are especially costly, the operating threshold may need to be lowered, accepting additional false alarms. If inspection resources are scarce, a higher threshold may be preferred to prioritise only high-scoring cases. A defensible deployment threshold should be selected using a validation set, explicit costs, and consultation with maintenance specialists. The threshold table demonstrates trade-offs; it does not identify a universally correct threshold.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26nov055

➢ *Interpretation and Feature Relationships*

Agreement between the model's feature_importances_ ranking and SHAP global ranking provides a useful consistency check, particularly for torque, rotational speed, and tool wear. However, importance is not proof of causation. Features may be correlated, and their contributions can depend on combinations of measurements. Further validation should examine stability across repeated splits and independent data.

➢ *Limitations and Threats to Validity*

- Synthetic dataset: AI4I 2020 is generated to resemble industrial data and may not reflect the full complexity, sensor noise, failure prevalence, or maintenance practices of a real plant.
- No physical sensor integration: the current dashboard simulates sensor readings; live hardware communication has not been demonstrated by the supplied code.
- Class imbalance: only 3.39% of the records are failures, and false-negative performance remains important.
- Single hold-out evaluation: the primary reported test results are from one stratified 80:20 split. They do not establish stability across time, machines, factories, or datasets.
- Threshold selection: the threshold experiments show trade-offs, but an operational threshold needs to be chosen using validation data and a defined cost model.
- Probability calibration: the reported model probabilities have not been shown to be calibrated; dashboard scores should not be interpreted as verified real-world risk probabilities.
- Explainability limits: global SHAP plots do not automatically explain each live prediction, and SHAP associations do not establish causal relationships.
- Model/version consistency: the uploaded files include results from different project stages and model configurations. The main table uses the supplied consolidated four-model comparison and Day 8 report; other CSVs should be labelled by experiment and not combined blindly.
- Operational readiness: SMTP delivery, cybersecurity, data governance, human factors, model monitoring, and safe maintenance workflows require separate validation.

➢ *Future Work*

- Connect validated industrial sensors or a historian/API feed and document sampling frequency, units, timestamps, and missing-data handling.
- Validate the model on real machine telemetry and use time-based or machine-based splits where appropriate.
- Use repeated stratified cross-validation or nested validation to estimate variability and tune model parameters without using the final test set for decisions.
- Calibrate probability outputs and select thresholds from an explicit cost matrix for false negatives, false positives, downtime, and inspection.
- Add precision–recall curves, calibration plots, confidence intervals, and subgroup analysis by product type.
- Investigate additional methods such as XGBoost or other gradient-boosting libraries only if the environment and evaluation protocol support them.
- Develop per-prediction explanations, clearly separating local explanations from the existing global SHAP plots.
- Improve dashboard persistence using a database, add access control, validate inputs, log model version and timestamp, and test alert delivery.
- Monitor data drift and performance over time, with a documented retraining and approval process.

VII. CONCLUSION

This project implements an end-to-end explainable predictive-maintenance prototype using the AI4I 2020 dataset. The workflow includes data inspection, exploratory visualisation, preprocessing, four-model comparison, threshold analysis, SHAP-based explanation, and a Streamlit dashboard. On the supplied 2,000-record hold-out test set, Gradient Boosting provides the strongest overall default-threshold balance by F1-score and ROC-AUC, while Random Forest produces fewer false alarms but misses more failures. The threshold experiments demonstrate that sensitivity can be increased at the cost of additional false alarms.

The explanation outputs consistently identify torque, rotational speed, and tool wear as important predictors in the trained tree models. The dashboard makes model outputs and reports easier to inspect, but it currently demonstrates simulated sensor input rather than live industrial connectivity. Because the dataset is synthetic and the test results are based on one split, the system should be considered a prototype for further research. Real-world deployment requires independent data validation, cost-aware threshold selection, probability calibration, monitoring, secure integration, and human oversight.

REFERENCES

[1.] S. Matzka, "Explainable Artificial Intelligence for Predictive Maintenance Applications," in Proceedings of the Third International Conference on Artificial Intelligence for Industries (AI4I), 2020, pp. 69–74, doi: 10.1109/AI4I49448.2020.00023.

[2.] S. M. Lundberg and S.-I. Lee, "A Unified Approach to Interpreting Model Predictions," in Advances in Neural Information Processing Systems 30, 2017. Available:https://papers.neurips.cc/paper/2017/hash/8a20a8621978632d76c43dfd28b67767-Abstract.html

[3.] UCI Machine Learning Repository, "AI4I 2020 Predictive Maintenance Dataset," 2020, doi: 10.24432/C5HS5C. Available:

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26nov055

https://archive.ics.uci.edu/dataset/601/ai4i+2020+predictive+maintenance+dataset
[4.] L. Breiman, “Random Forests,” Machine Learning, vol. 45, pp. 5–32, 2001, doi: 10.1023/A:1010933404324.
[5.] J. H. Friedman, “Greedy Function Approximation: A Gradient Boosting Machine,” The Annals of Statistics, vol. 29, no. 5, pp. 1189–1232, 2001, doi: 10.1214/aos/1013203451.
[6.] F. Pedregosa et al., “Scikit-learn: Machine Learning in Python,” Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.
[7.] C. Molnar, Interpretable Machine Learning, 2nd ed., 2022. Available:https://christophm.github.io/interpretable-ml-book/
[8.] S. Lundberg et al., “From local explanations to global understanding with explainable AI for trees,” Nature Machine Intelligence, vol. 2, pp. 56–67, 2020, doi: 10.1038/s42256-019-0138-9.