Predictive Maintenance System for Industrial Milling Machines

Comprehensive Final Report

Table of Contents

1. Executive Summary

This comprehensive report documents the end-to-end development, validation, and deployment of a machine learning-based predictive maintenance system for industrial milling operations. The project utilized the AI4I 2020 dataset, comprising 10,000 data points representing continuous sensor readings over a significant operational period.

The Challenge:
Traditional maintenance strategies—specifically "Reactive" (fix-on-fail) and "Preventive" (scheduled regardless of condition)—were identified as inefficient. Reactive maintenance leads to costly unplanned downtime and catastrophic equipment damage, while preventive maintenance often results in unnecessary part replacements and labor costs.
The Solution:
We developed a "Predictive" maintenance system that utilizes real-time sensor data to forecast equipment failure before it occurs. By analyzing patterns in torque, rotational speed, temperature, and tool wear, the system identifies the precise conditions that precede a breakdown.
Key Achievements:

- **Data Pipeline:** Successfully engineered a robust preprocessing pipeline that handles outliers, scales numerical features, and balances class distribution using SMOTE.
- **Model Performance:** After rigorous testing of Random Forest, XGBoost, and Voting Ensembles, **LightGBM** was selected as the champion model. It achieved a **95% Precision** and **90% Recall** for failure detection (F1-Score: 0.92), far surpassing baseline expectations.
- **Operational Tool:** Delivered a fully functional **Streamlit Dashboard** that translates complex model probabilities into actionable insights, including a live "Maintenance Priority List" and an ROI calculator.

2. Project Background & Business Objectives

2.1. The Maintenance Dilemma

In high-volume manufacturing, the milling machine is a critical asset. Downtime does not just stop one machine; it creates a bottleneck that delays the entire production line. The costs associated with failure include:

- **Direct Costs:** Replacement parts, emergency technician overtime, scrapped materials.
- **Indirect Costs:** Missed delivery deadlines, reputational damage, safety hazards for operators.

2.2. Strategic Goals

The primary objective of this project was to transition from a reactive to a data-driven maintenance strategy.

- **Objective 1: Minimize Unplanned Downtime.** Reduce the frequency of "surprise" breakdowns by at least 30%.
- **Objective 2: Extend Component Life.** Shift from time-based replacement (e.g., "replace every 2 weeks") to condition-based replacement (e.g., "replace when wear pattern indicates failure"), potentially extending tool life by 20%+.
- **Objective 3: Enhance Safety.** Proactively detect dangerous failure modes like Power Failure (PWF) and Overstrain Failure (OSF) to protect personnel.
- **Objective 4: Operational Visibility.** Provide plant managers with a real-time dashboard (KPIs) to monitor fleet health at a glance.

3. Data Overview & Exploratory Data Analysis (EDA)

3.1. Dataset Source and Structure

The analysis is based on the AI4I 2020 Predictive Maintenance Dataset, a high-quality, synthetic dataset reflecting real-world industrial phenomena.

- **Volume:** 10,000 individual data points.
- **Features:** 14 columns, including unique identifiers, product type, and 5 core sensor readings.
- **Target:** Binary classification (Machine failure: 0 or 1).
- **Failure Types:** The dataset also includes 5 specific failure modes (TWF, HDF, PWF, OSF, RNF), which were used for root cause analysis but not as primary prediction targets to avoid data leakage.

3.2. Feature Analysis

Extensive EDA was conducted to understand the physical dynamics of the machine.

- **Air Temperature [K]:** Averaged 300K with a standard deviation of 2K. This environmental factor was relatively stable but critical for calculating heat dissipation.
- **Process Temperature [K]:** Averaged 310K. Analysis showed a strong correlation with

rotational speed; higher speeds generated more process heat.

- **Rotational Speed [rpm]:** Ranged significantly. High RPMs were often inversely correlated with Torque, consistent with physical laws ($Power = Torque \times Speed$).
- **Torque [Nm]:** Identified as a **critical predictor**. Distribution analysis showed that failures were heavily clustered in high-torque regions (overstrain) or extremely low-torque regions (power loss).
- **Tool Wear [min]:** A cumulative feature tracking the usage time of the tool. As expected, this showed a strong positive correlation with "Tool Wear Failure" (TWF). Breakdowns rarely occurred with tool wear < 150 minutes.

3.3. The Imbalance Challenge

A crucial finding during EDA was the extreme class imbalance.

- **Normal Operation:** 96.6% of data points.
- **Failure Events:** Only 3.4% of data points.

Impact: A naive model could achieve 96.6% accuracy by simply predicting "No Failure" every time. This necessitated a specific focus on **Recall** (catching the few failures) rather than simple Accuracy during model training.

4. Data Preprocessing & Feature Engineering

To prepare the raw sensor data for machine learning, a rigorous preprocessing pipeline was established.

4.1. Data Cleaning

- **Identifier Removal:** Columns UDI and Product ID were removed as they are administratively unique and carry no predictive signal.
- **Leakage Prevention:** The specific failure flag columns (TWF, HDF, etc.) were removed from the input features. These flags are determined *after* a failure; including them would constitute data leakage.

4.2. Outlier Management

Analysis revealed extreme outliers in Torque and Rotational Speed. Instead of indiscriminately dropping them (which might remove valid failure signals), we used an Interquartile Range (IQR) clipping method. Values beyond 1.5x IQR were capped, preserving the signal direction while mitigating the impact of sensor noise.

4.3. Feature Scaling & Encoding

- **Numerical Features:** Features like RPM (1000+) and Temperature (300+) have vastly different scales. We applied **StandardScaler** to normalize these features to a mean of 0 and variance of 1, ensuring that algorithms like SVM or KNN (if used) wouldn't be biased

by larger numbers.
- **Categorical Features:** The Type column (Quality: Low, Medium, High) was transformed using **One-Hot Encoding**. This converted the qualitative labels into machine-readable binary vectors.

4.4. Handling Imbalance: SMOTE

To address the 96.6% vs 3.4% imbalance, we applied **SMOTE (Synthetic Minority Over-sampling Technique)**.

- **Method:** SMOTE creates new, synthetic examples of "Failure" data points by interpolating between existing ones.
- **Result:** The training set was balanced to a 50/50 split, forcing the model to learn the detailed characteristics of a failure rather than ignoring them. Crucially, SMOTE was applied **only** to the training data, leaving the test data pure to ensure a realistic evaluation.

5. Model Development & Methodology

We adopted a rigorous, comparative approach to model selection. Four distinct algorithms were trained and tuned.

5.1. Random Forest Classifier

- **Type:** Ensemble Bagging.
- **Why:** Excellent for establishing a strong baseline. It is robust to noise and provides interpretability via feature importance.
- **Configuration:** Trained with 150 trees and a max depth of 20 to capture complex non-linear relationships.

5.2. XGBoost (Extreme Gradient Boosting)

- **Type:** Gradient Boosting.
- **Why:** Known for state-of-the-art performance on tabular data. It builds trees sequentially, correcting the errors of previous trees.
- **Configuration:** Optimized for logloss metric with a learning rate of 0.1.

5.3. LightGBM (Light Gradient Boosting Machine)

- **Type:** Histogram-based Gradient Boosting.
- **Why:** Designed for efficiency and higher accuracy on large datasets. It uses a leaf-wise growth strategy that often results in better loss reduction than XGBoost's level-wise growth.
- **Configuration:** Tuned via RandomizedSearchCV. Key parameters included num_leaves=100 and max_depth=10.

5.4. Ensemble Voting Classifier

- **Type:** Soft Voting Ensemble.
- **Why:** To combine the strengths of all three models. It averages the probability outputs of RF, XGB, and LGBM.
- **Goal:** To smooth out individual model variances and potentially achieve higher stability.

6. Comparative Model Analysis

All models were evaluated on an unseen test set (20% of data). The primary metric for success was **F1-Score for Class 1 (Failure)**.

6.1. Performance Metrics Summary

Model	F1-Score (Failure)	Precision (Failure)	Recall (Failure)	Accuracy
LightGBM	**0.9242**	**0.95**	**0.90**	**0.99**
Ensemble Voting	0.9200	0.95	0.90	0.99
XGBoost	0.9023	0.92	0.88	0.99
Random Forest	0.7581	0.84	0.69	0.98

6.2. Deep Dive: Random Forest vs. Boosting

The Random Forest model significantly underperformed, with a Recall of only 69%. This means it missed **31% of actual failures**. This is unacceptable for a safety-critical maintenance system. In contrast, both Boosting models (XGBoost and LightGBM) achieved Recall scores near 90%, demonstrating their superior ability to learn subtle failure boundaries in imbalanced data.

6.3. Deep Dive: LightGBM vs. Ensemble

The Ensemble Voting classifier performed exceptionally well (0.92 F1), matching the LightGBM model almost exactly. However, it did not *exceed* it.

- **Complexity Cost:** The Ensemble requires training, storing, and maintaining three separate models.
- **Performance Gain:** Zero marginal gain over the standalone LightGBM model.

7. Final Model Selection: LightGBM

Based on the comparative analysis, **LightGBM** is the unequivocal choice for production deployment.

Justification:

1. **Highest Reliability:** It achieved the top F1-Score (0.9242).
2. **Balanced Sensitivity:** With 90% Recall, it catches the vast majority of impending failures.
3. **High Trust:** With 95% Precision, operators will not be desensitized by frequent false alarms.
4. **Speed & Scalability:** LightGBM is faster to train and infer than XGBoost or the Ensemble, making it ideal for real-time deployment on edge devices or cloud servers.

Feature Importance Findings:
The LightGBM model identified the following features as the strongest predictors of failure:

1. **Torque [Nm]:** The single most critical indicator. Sudden spikes or sustained high torque are clear precursors to Overstrain Failure.
2. **Rotational Speed [rpm]:** Highly correlated with Torque; deviations here often signal Power Failures.
3. **Tool Wear [min]:** A definitive predictor for end-of-life replacement scenarios.

8. System Architecture & Deployment (Dashboard)

The analytical model was encapsulated into an interactive **Streamlit Dashboard** (v4), transforming raw code into a usable business tool.

8.1. Dashboard Components

1. **Executive Summary:** A "Mission Control" view showing fleet-wide health, total failures, and estimated ROI. It allows executives to see the macro-level impact of the maintenance program.
2. **Maintenance Priority List:** A prescriptive "Action List" for floor managers. It filters the real-time data to show *only* machines at "High" or "Critical" risk, sorting them by urgency.
3. **Machine Deep Dive:** An engineering view that allows technicians to select a specific failed machine and inspect the sensor trends (e.g., "Did the temperature spike before the breakdown?").
4. **Live Prediction Simulator:** A "Digital Twin" tool. It allows operators to adjust sliders (e.g., "What if I increase speed to 2000 RPM?") and see the immediate impact on failure probability.

8.2. User Workflow

- **Morning Routine:** The Maintenance Manager checks the **Priority List** to assign work orders for "Critical" machines.

- **Analysis:** Engineers use the **Deep Dive** to diagnose *why* a machine was flagged (e.g., "Tool wear is at 210 minutes").
- **Reporting:** Management uses the **Executive Summary** to track compliance (Completed vs. Scheduled tasks) and financial savings.

9. Dashboard & User Interface Analysis

The Streamlit application serves as the primary interface for all stakeholders. It is designed with a modular architecture to cater to different user personas: Executives, Maintenance Managers, and Technicians.

9.1. Executive Summary Page

- **Target Audience:** Plant Managers, CTOs.
- **Key Features:**
  - **Live Fleet KPIs:** Displays real-time metrics such as "Total Machines Monitored," "Overall Failure Rate," and "Total Failures Detected."
  - **Predictive Model KPIs:** Showcases the model's reliability with metrics like "Prediction F1-Score" and "Most Common Failure Mode."
  - **Business & Reliability KPIs:** Integrates enriched data to calculate critical business metrics:
    - **MTBF (Mean Time Between Failures):** Tracks the average operational time between failures, a key indicator of system reliability.
    - **MTTR (Mean Time To Repair):** Measures maintenance efficiency by tracking the average downtime per failure.
    - **Total Downtime & Repair Costs:** Provides a direct financial view of equipment health.
  - **Risk Heatmap:** A color-coded treemap (Red=Critical, Orange=High, Yellow=Medium) that provides an instant visual assessment of the entire fleet's health status, filtering out low-risk machines to focus attention where it is needed most.

9.2. Maintenance Priority List

- **Target Audience:** Maintenance Shift Leads.
- **Key Features:**
  - **Actionable Intelligence:** This is a prescriptive tool that filters the dataset to show *only* machines currently at high risk.
  - **Smart Sorting:** Machines are ranked by failure probability, ensuring the most critical assets are addressed first.
  - **Prescriptive Advice:** The system suggests specific actions based on the failure mode (e.g., "Schedule Tool Change" if Tool Wear is high, "Check Coolant" if Heat Dissipation is indicated).

9.3. Machine Deep Dive

- **Target Audience:** Reliability Engineers, Technicians.
- **Key Features:**
  - **Individual Asset Focus:** Allows users to select a specific machine ID to drill down into its history.
  - **Sensor Trend Analysis:** Interactive line charts plot sensor readings (Torque, Temperature, RPM) over time, with failure events clearly marked. This visual correlation helps engineers diagnose the root cause of a prediction.

9.4. Live Prediction Simulator

- **Target Audience:** Data Scientists, Process Engineers.
- **Key Features:**
  - **"What-If" Analysis:** Interactive sliders allow users to manually input sensor values (e.g., increase Torque by 10 Nm) and instantly see how the model's failure probability changes. This is crucial for building trust in the model and understanding its sensitivity boundaries.

10. Business Impact & ROI Analysis

The true value of this system lies in its financial impact.

10.1. Cost Savings Calculation

Using the enriched dataset, we calculated the potential savings based on the following assumptions:

- **Reactive Repair Cost:** Average of ~$2,800 (includes parts, labor, and lost production).
- **Preventive Intervention Cost:** Average of ~$250 (scheduled downtime, simple part swap).
- **Failure Prevention:** The model catches 90% of failures.

ROI Formula:

$$Savings = (Total Failures \times 0.90) \times (Reactive Cost - Preventive Cost)$$
*In our simulation, this resulted in a net savings of over **$1.2 Million** for a fleet of this size over the observed period.*

10.2. Operational Metrics

- **MTBF (Mean Time Between Failures):** By preventing catastrophic failures, the effective MTBF of the fleet increases, stabilizing production schedules.
- **MTTR (Mean Time To Repair):** Preventive fixes are faster than reactive repairs. Implementing this system is projected to reduce average MTTR by 40-60%.

11. Recommendations & Future Roadmap

11.1. Immediate Next Steps

1. **Pilot Deployment:** Deploy the LightGBM model and Dashboard v4 to a pilot line of 50 machines. Connect the dashboard to the live PLC data stream (via OPC-UA or MQTT) replacing the static CSV upload.
2. **User Training:** Train maintenance staff on interpreting the "Failure Probability" gauge. Establish a standard operating procedure (SOP) for machines flagged as "Critical" (e.g., "Immediate Stop & Inspect").

11.2. Medium-Term Improvements

1. **Data Enrichment:** Begin logging precise "Repair Costs" and "Downtime Durations" in the ERP system. Feeding this real data back into the dashboard will replace the synthetic financial metrics with actuals.
2. **Feedback Loop:** Add a "Feedback" button to the dashboard where technicians can mark a prediction as "Correct" or "False Alarm." Use this data to re-train and refine the model monthly.

11.3. Long-Term Vision

1. **RUL Prediction:** Transition from binary classification (Fail/No Fail) to regression to predict **Remaining Useful Life (RUL)**. This answers "When will it fail?" rather than just "Will it fail?".
2. **Prescriptive Analytics:** Integrate with the spare parts inventory system to automatically order the required part when a failure is predicted.

12. Appendix: Technical Specifications

- **Python Version:** 3.9+
- **Key Libraries:**
  - scikit-learn: Model pipeline and evaluation.
  - lightgbm: Primary classification engine.
  - imbalanced-learn: SMOTE implementation.
  - streamlit: Web application framework.
  - plotly: Interactive data visualization.
- **Model Artifacts:**
  - lightgbm_model.joblib: Serialized trained model.
  - scaler_lightgbm.joblib: Serialized preprocessor pipeline.