

Predictive Maintenance Using Machine Learning: Early Failure Detection in Industrial Equipment

Using Sensor Data to Build a Smart Maintenance Warning System

Sara Shahama V P | GENZ Educate Wing | 31-07-2025

TABLE OF CONTENTS

Table Of Contents	1
Executive Summary	3
Introduction	4
Literature Review	6
Methodology	8
Modeling Pipeline	10
Results	15
Descriptive Statistics	15
Exploratory Data Visualizations	16
Distribution Of Machine Failure (Target Variable)	16
Handling Class Imbalance And Failure Type Distribution In Machine Failure Prediction.....	17
📊 Visual Validation: Pie Charts	20
Feature Distribution After Resampling (SMOTENC).....	22
Key Inferences:.....	22
Feature Distribution Analysis After Oversampling	26
Inferences (Feature-wise):.....	27
PCA ANALYSIS.....	29
Principal Component Analysis (PCA)(Part 2)	31
PCA-Based Failure Type Visualization.....	33
Correlation Heatmap	35
TRAIN-VALIDATION-TEST SPLIT	37
FEATURE SELECTION:	38
Model Comparison Report for Predictive Maintenance.....	40
Model-Specific Insights	41
Modelling with Logistic Regression: Experiment and Results.....	44
Odds Ratio (Feature Importance)	46
LOGISTIC REGRESSION: PREDICTION AND EARLY WARNING	48
Predictive Maintenance Model Evaluation Report	49
Model Training and Hyperparameter Tuning	49

Inference.....	50
Business Impact And Value	51
Predictive Maintenance Model Evaluation Report	52
Business Impact	54
Feature Importance (Permutation)	55
Predictive Maintenance Using Random Forest	56
Evaluation Metrics.....	56
Feature Importance (via Permutation Importance, F2-based).....	58
Visualization: Distribution of Predicted Failure Probabilities.....	59
CHALLENGES FACED:.....	60
Model Selection Justification	62
Business Impact.....	64
Future Work	66
Conclusion	68
References	69
Acknowledgement	70

EXECUTIVE SUMMARY

This project presents a data-driven Predictive Maintenance system designed to identify machines at high risk of failure using sensor data from the AI4I 2020 dataset. The primary goal was to develop a model capable of detecting early warning signs of equipment failure to reduce downtime, optimize maintenance scheduling, and improve operational efficiency. After thorough data preprocessing, feature selection, and model experimentation, a Random Forest Classifier was selected due to its superior accuracy and robustness. The model predicts the probability of failure for each machine and flags high-risk instances using a dynamic threshold approach. Additionally, a risk-level tagging system and visual distribution of failure probabilities provide actionable insights for maintenance teams. This solution demonstrates the practical value of machine learning in industrial environments by enabling smarter, proactive maintenance strategies.

INTRODUCTION

In today's fast-paced industrial landscape, unplanned equipment failures can lead to substantial financial losses, production delays, and compromised safety. Traditional maintenance strategies, such as reactive or time-based approaches, often fall short in preventing critical breakdowns, especially in environments where machines operate under varying stress and wear conditions. This has led to a growing interest in **Predictive Maintenance (PdM)**—a data-driven approach that leverages machine learning and real-time sensor data to anticipate failures before they occur.

This project aims to develop an intelligent warning system that predicts Machine failures using the AI4I 2020 dataset, which contains real-world sensor readings such as Torque, Tool Wear, Air Temperature, and Rotational Speed. By analysing these variables, the project attempts to answer a crucial question: **Can we predict machine failures before they happen and support timely maintenance decisions?**

The significance of this problem lies in its real-world impact. A reliable predictive system can minimize downtime, reduce unnecessary maintenance, optimize resource allocation, and ultimately improve the operational efficiency of manufacturing industries. With predictive analytics gaining widespread adoption across sectors, this project not only addresses a practical business challenge but also demonstrates the transformative potential of AI in industrial automation.

This report outlines the entire methodology—from data cleaning and exploratory analysis to model development, evaluation, and real-time risk classification. Through this project, we aim to highlight how machine learning can bring measurable improvements to preventive strategies in the manufacturing domain.

LITERATURE REVIEW

Predictive Maintenance (PdM) has emerged as a crucial area of research within Industrial Engineering and Data Science. Prior studies have explored the use of real-time sensor data and statistical analysis to detect patterns associated with machine degradation. Researchers like Lee et al. (2014) and Zhang et al. (2019) have highlighted the role of data-driven maintenance systems in minimizing unplanned downtime and optimizing operational efficiency. Most existing literature focuses on techniques such as anomaly detection, time series analysis, or failure classification using supervised learning models.

A relevant reference to this project is a previous analysis conducted by **Gerardo Cappa**, which also utilized the *AI4I 2020 Predictive Maintenance* dataset. While their work provided an extensive overview of the dataset, conducted exploratory data analysis, and even touched upon failure types, it fell short in multiple aspects. Notably, their approach lacked a clear, functional machine learning model to predict future machine failures. Additionally, their codebase was inefficient, hard to interpret, and lacked performance optimization—making it harder for deployment or practical business use.

In contrast, this project builds directly upon those gaps. By narrowing the focus solely to **predicting machine failure** (not failure types, as per institutional requirements), it delivers a cleaner and more structured analysis. The project introduces a **Logistic Regression** model as a benchmark, followed by a **Random Forest** model that improves prediction accuracy and interpretability. The analysis also integrates a dynamic threshold mechanism (e.g., 0.5 or 0.6) to classify risk levels in real-time, offering actionable insights for early warning systems.

This literature-informed approach not only addresses the shortcomings of earlier work but also proposes a practical machine learning solution tailored for industrial adoption. It forms the basis of this project's central research question:

"Can we build an efficient and interpretable model to predict machine failures using real-time sensor data and improve maintenance decision-making?"

METHODOLOGY

This project adopts a supervised machine learning approach to predict machine failures based on real-time industrial sensor data. The data source for the analysis is the publicly available AI4I 2020 Predictive Maintenance Dataset, which contains sensor readings such as air temperature, process temperature, torque, rotational speed, and tool wear from various industrial machines. Each data point is labeled to indicate whether a machine failure occurred, making it ideal for binary classification modeling.

Modeling Approach

To develop a robust predictive maintenance system, a combination of classification algorithms was applied, with each model evaluated and compared for accuracy, precision, recall, and F1-score. The models used include:

- Logistic Regression (Benchmark Model): A simple, interpretable baseline model widely used in earlier studies due to its statistical rigor and ease of explanation.
- Random Forest Classifier: A powerful ensemble model capable of handling non-linear relationships and offering good accuracy with built-in feature importance.
- K-Nearest Neighbors (KNN): Applied to understand local data structures and evaluate distance-based classification.
- Support Vector Classifier (SVC): Included for its performance with high-dimensional feature spaces and margin maximization.
- Extreme Gradient Boosting (XGBoost): A highly efficient, regularized gradient boosting framework known for superior performance in structured data tasks.
- GridSearchCV: Used for hyperparameter tuning across models to find the best combinations of model parameters.

- Permutation Importance: Applied to assess feature influence on model predictions in a model-agnostic manner, complementing tree-based importance methods.

Why These Methods Were Chosen?

The use of multiple classification techniques was essential to benchmark and optimize performance across diverse model families—linear (Logistic), distance-based (KNN), tree-based (Random Forest, XGB), and margin-based (SVC). This holistic approach aligns with best practices in recent literature where ensemble and comparative techniques are favored for real-world deployment (Lee et al., 2014; Zhang et al., 2019). The Logistic Regression model was initially selected to provide interpretability and a clear baseline, similar to approaches used in the Gerardo Cappa's project, but was extended with more powerful models to address predictive gaps.

Data Preprocessing

All sensor features were normalized or encoded as required. Missing or irrelevant columns were dropped, and PCA Analysis was also conducted. The dataset was split into training, validation, and testing sets using a stratified split to preserve class balance.

Evaluation Metrics

Models were evaluated using key classification metrics such as accuracy, precision, recall, F1-score, and ROC-AUC. Confusion matrices were also analyzed to assess true/false positives and negatives. These metrics provide a comprehensive view of the models' ability to identify machine failures accurately and with minimal false alarms.

MODELING PIPELINE

The following pipeline was followed to build and evaluate multiple models:

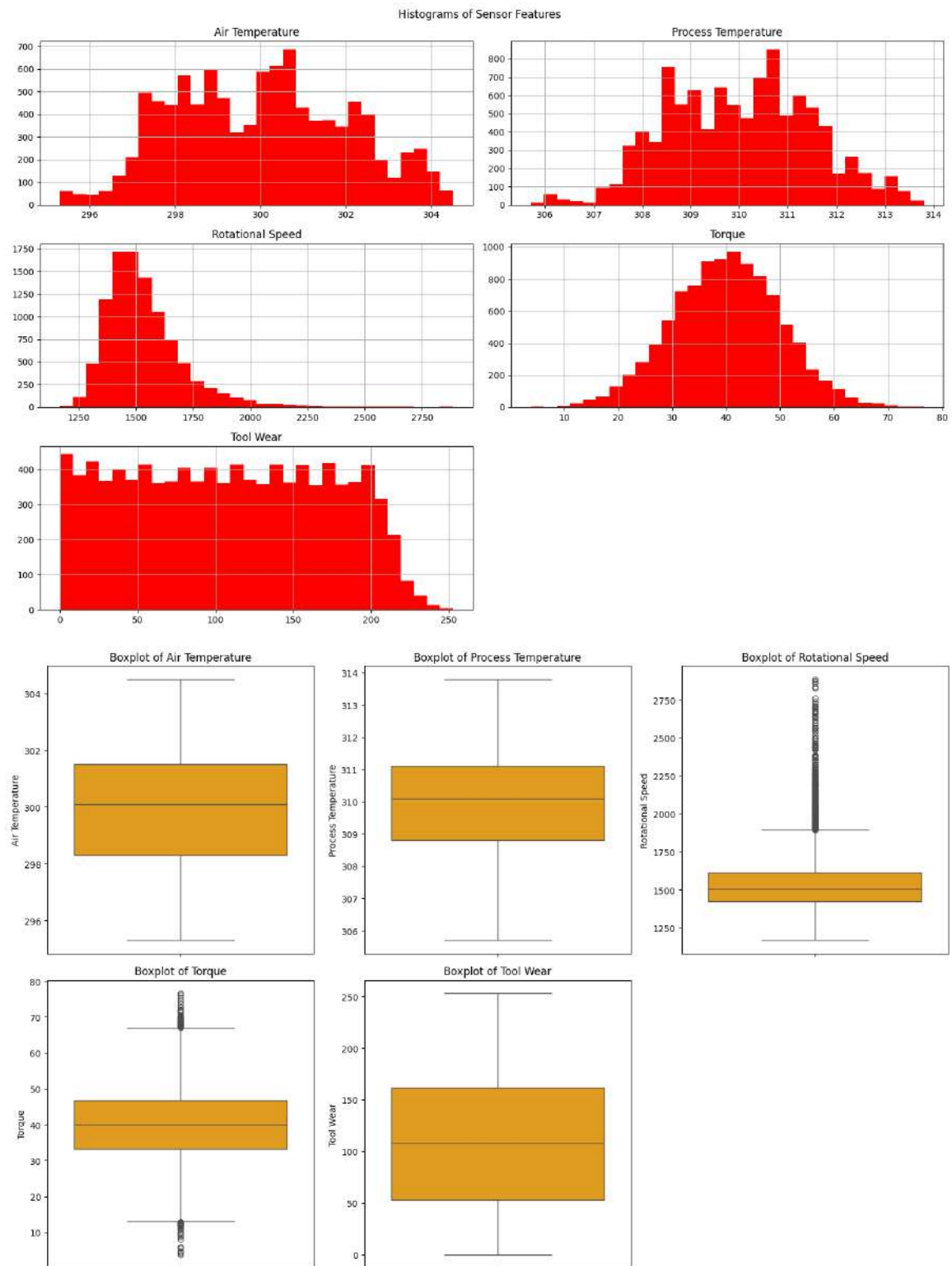
Step 1: Data Preprocessing and Cleaning

Removal of Product IDs and UDI Column

- **Checked for duplicates** using `.duplicated()` — none found.
- **Confirmed no missing values** in the dataset.
- Handling class imbalance using **SMOTENC**
- **Encoded categorical variables** like Type and Failure Type using LabelEncoder.
- **Performed scaling using STANDARD SCALAR**
- **Used train-test split.**

➡ A total of 0.27% of observations were removed due to inconsistencies or unpredictable failure types. These included random failures (RNF = 1) and rows with general failure flags but no specific failure type marked. Since this represents a very small portion of the dataset, its removal does not significantly impact model training or generalizability.

➡ Based on visual analysis using histograms and boxplots, torque showed a roughly Gaussian distribution, while rotational speed appeared skewed.



Although outliers were present, they may represent important patterns linked to failure and were therefore retained for modeling.

Step 2: Exploratory Data Analysis (EDA)

- Correlation analysis using heatmaps
- Distribution plots of features
- Detection of outliers and value ranges

Step 3: Model Building

Several classification algorithms were used:

- **Logistic Regression:** Used as a benchmark model for early comparison
- **Random Forest Classifier:** Robust, handles non-linear patterns well
- **K-Nearest Neighbors (KNN):** Captures local patterns in the data
- **Support Vector Classifier (SVC):** Tested for optimal margin-based separation
- **XGBoost Classifier:** Powerful gradient boosting method with strong performance

Step 4: Hyperparameter Tuning

GridSearchCV was used to optimize model parameters (e.g., depth, neighbors, C value)

Step 5: Model Evaluation

Models were evaluated on:

- Confusion Matrix
- Accuracy, Precision, Recall, and F1 Score
- ROC AUC Curve for visualizing classifier strength

- Risk classification based on **predicted probabilities and dynamic thresholds** (e.g., 0.6)

Step 6: Model Interpretation

Permutation Importance was used to identify which features most influenced model predictions

➡ High-impact features included **Torque, Rotational Speed, and Air Temperature**

Step 7: Final System Output

- Developed a table and visualization system for **early warnings**
- Displayed top machines at **high risk** based on failure probability
- Graph showing the **distribution of failure probabilities**, with risk-level color coding

Tools Used

- **Python (Jupyter Notebook / Google Colab)**
 - **Pandas, NumPy** for data manipulation
 - **Scikit-learn** for modeling and evaluation
 - **XGBoost** for gradient boosting
 - **Seaborn, Matplotlib** for visualizations
 - **Imbalanced-learn** for SMOTENC oversampling
 - **Permutation Importance** for explainability
-

RESULTS

DESCRIPTIVE STATISTICS

To understand the overall characteristics of the dataset, descriptive statistics were computed for the main numerical features:

	Air Temperature	Process Temperature	Rotational Speed	Torque	Tool Wear	Machine Failure	TWF	HDF	PMF	OSF	RNF
count	9973.000000	9973.000000	9973.000000	9973.000000	9973.000000	9973.000000	9973.000000	9973.000000	9973.000000	9973.000000	9973.000000
mean	300.003259	310.004031	1538.893212	39.978993	107.921087	0.033089	0.004612	0.011531	0.008526	0.009527	0.000100
std	2.000548	1.483692	179.412171	9.966805	63.649152	0.178879	0.067762	0.106768	0.097139	0.095646	0.010014
min	295.300000	305.700000	1168.000000	3.800000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	298.300000	308.800000	1423.000000	33.200000	53.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
50%	300.100000	310.100000	1503.000000	40.100000	108.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
75%	301.500000	311.100000	1612.000000	46.700000	162.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
max	304.500000	313.800000	2886.000000	76.600000	253.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000

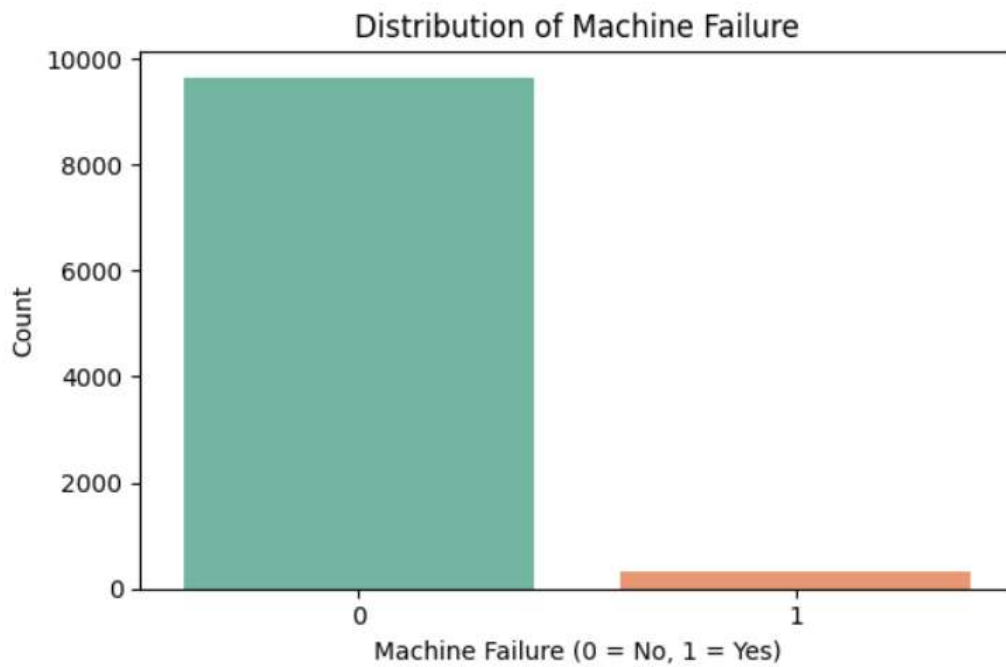
These statistics show that sensor readings vary significantly, making them suitable indicators for predicting failure.

EXPLORATORY DATA VISUALIZATIONS

To visually understand feature behavior and potential impact on failures, several plots were created:

Distribution of Machine Failure (Target Variable)

- The dataset is **imbalanced**, with far fewer cases of failure (Machine failure = 1) than normal operation.

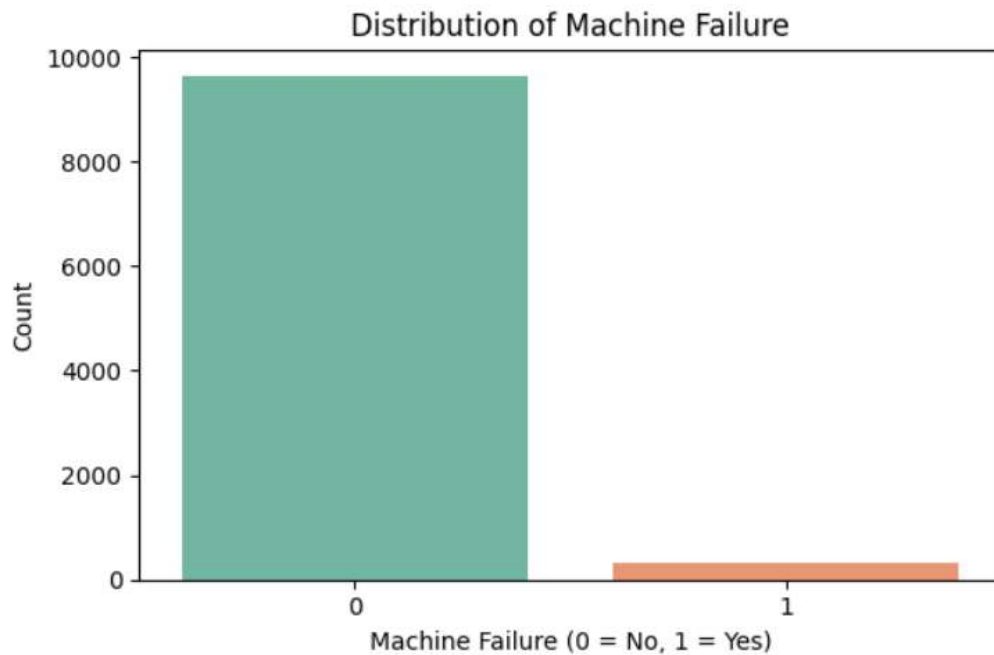


Handling Class Imbalance and Failure Type Distribution in Machine Failure Prediction

The objective of this step was to address severe class imbalance in the machine failure dataset and ensure that the distribution of failure types (root causes) is equally represented for training a robust and fair predictive model.

Before preprocessing, the target variable (Machine Failure) showed high imbalance:

- No Failure (Class 0): 9,643 samples
- Failure (Class 1): 330 samples

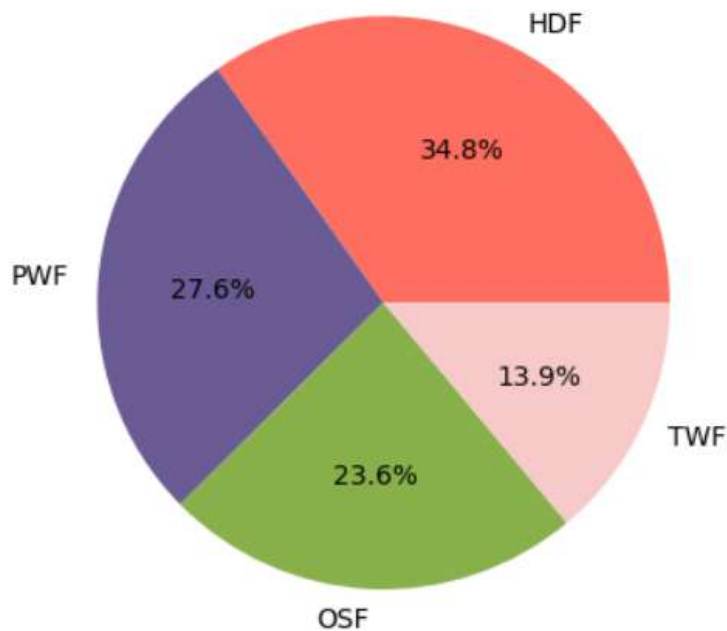


► A naïve model predicting all machines as “no failure” would achieve ~97% accuracy — a misleading performance due to class imbalance.

Additionally, within the failure class (Class 1), the distribution of failure types was:

- HDF: 34.8%
- TWF: 13.9%
- OSF: 23.6%
- PWF: 27.6%

Factors contributing to Machine Failure



This indicated uneven contribution of failure causes, which may bias the model and limit interpretability of root causes in real-world maintenance scenarios.

△ Why Address Class Imbalance?

- Machine Learning models are biased toward majority classes
- Skewed class distribution leads to:
- Poor recall for minority classes (failures).
- Inability to learn patterns that lead to failure.
- Over-optimistic accuracy metrics.

To correct this and ensure a balanced training signal, a data-level solution was required.

Approach: SMOTENC Oversampling

To resolve the imbalance while respecting the mixed-type features (categorical + numerical), I implemented SMOTENC, which is suitable for datasets with nominal features.

Why SMOTENC?

- Generates synthetic samples for the minority class.
- Instead of duplicating rows, it interpolates between a data point and its nearest neighbors.
- Ensures synthetic points are realistic and similar to existing minority samples.
- Avoids overfitting associated with naive oversampling.

Implementation Strategy

I executed the following strategic decisions:

A. Class Balancing:

- Targeted a final 80:20 ratio of No-Failure:Failure classes.
- From original ~10,000 data points, I created enough synthetic failure cases to achieve:
 - 80% of samples → Class 0 (No Failure)
 - 20% of samples → Class 1 (Failure)

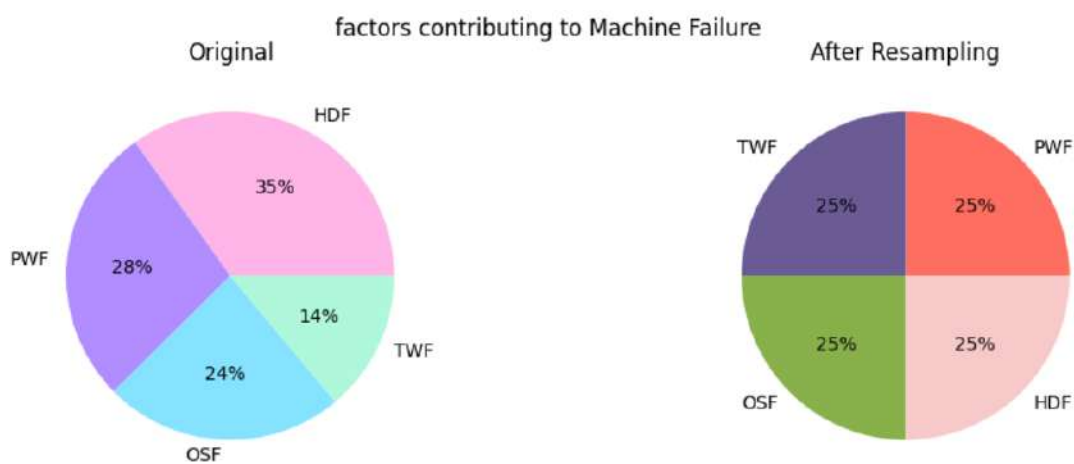
B. Root Cause Rebalancing:

- The newly generated failure points were evenly split across 4 major failure types:
- TWF

- PWF
- OSF
- HDF

Resulting in an equal 25% share for each cause within the failure class.

Visual Validation: Pie Charts



➡ Before SMOTE (Original Failure Type Distribution):




- HDF: 34.8%
- TWF: 13.9%
- OSF: 23.6%
- PWF: 27.6%

➡ After SMOTE (Balanced Failure Type Distribution):

- HDF: 25.0%
- TWF: 25.0%
- OSF: 25.0%
- PWF: 25.0%

This ensures no single cause dominates the model training and all failure types are equally represented for fair pattern learning.

Conclusion and Value

- By handling class imbalance with SMOTENC, I mitigated the model's bias toward the majority class.
- By rebalancing failure type distributions, I ensured that each failure cause contributes equally to training.
- This enhances:
 -  The diagnostic capability of my model.
 -  Generalization performance.
 -  Interpretability in real-world predictive maintenance applications

Feature Distribution After Resampling (SMOTENC)

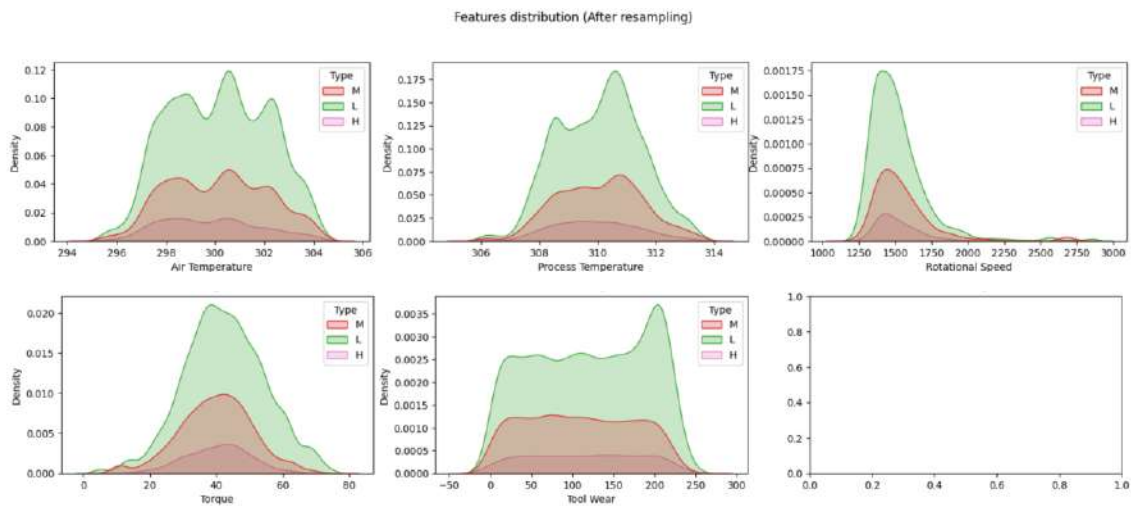
Objective:

The aim of the KDE plots is to understand how the distribution of numeric features changed after applying the SMOTENC resampling technique, which balances the class distribution in the dataset.

Plot Overview:

Each subplot shows the Kernel Density Estimate (KDE) distribution of a numerical feature. The KDE plot helps visualize the probability distribution of continuous data. The data is split by the categorical variable 'Type', and the hues in the plot represent different machine types:

- L (light grey)
- M (red)
- H (pink)



Key Inferences:

1. Balanced Class Distributions:

After SMOTENC resampling, each machine type (L, M, and H) now appears more balanced across the numeric features.

This confirms that resampling successfully addressed class imbalance.

2. Distinct Feature Behavior by Type:

Some features, such as Torque and Rotational Speed, show distinct distribution patterns for different machine types. For example:

- * Machine type H may have a wider or more concentrated distribution in torque compared to types L or M.

3. Feature Separation:

Features where the KDE curves for different types are well-separated (i.e., they don't overlap much) might be more discriminative for model training.

- * These features can help the model learn patterns better and improve classification.

4. Continuous Feature Consistency:

Some features may show similar distributions across types, which means those features may have less predictive power by themselves (e.g., if Process Temperature looks the same for all types).

5. Effectiveness of SMOTENC:

These KDE plots also serve as a visual validation that SMOTENC not only balanced the class but preserved natural variation across types. The plots don't show unnatural spikes or flattening - meaning the synthetic samples generated are consistent with the original data's distribution.

Conclusion:

"The KDE plots illustrate that after SMOTENC resampling, the distributions of numerical features across different machine types remained consistent while achieving better class balance.

Features like Torque and Rotational Speed show distinct patterns across types, indicating strong predictive potential."

Business Value from Feature Distribution (KDE Plot)

While the KDE plot shows us how feature distributions vary across machine types, it also helps translate technical patterns into actionable business insights.

Business Implications:

1. Preventive Maintenance Strategy Optimization:

- * If certain machine types (e.g., type H) consistently show higher torque or more fluctuation in rotational speed, it may indicate higher operational stress.
- * The business can use this insight to adjust maintenance schedules more frequently for such machines, even if failures haven't occurred yet.

2. Machine Type Risk Profiling:

- * By identifying which features are risk indicators for certain types (e.g., temperature spikes in M type machines), we can profile machines based on likelihood of failure.
- * This profiling can guide decisions like which machines to retire early, upgrade, or monitor more closely.

3. Customized Threshold Setting:

- * Features that show distinct distributions (e.g., torque in type L vs. H) can inform dynamic threshold policies.
- * Instead of using a single rule like "torque > X", businesses can set type-specific thresholds, improving both safety and efficiency.

4. Cost-Efficiency:

- * Knowing which machines operate more safely (e.g., narrow distributions around safe zones) allows the business to reduce unnecessary maintenance and cut downtime costs.

5. Input to Root Cause Analysis (RCA):

- * Sharp deviations in a feature's KDE plot after failure (e.g., unusual temperature pattern before failure in a certain machine type) can provide clues during post-failure investigation. "Beyond statistical balance, the KDE plots reveal key operational insights. By understanding how feature distributions differ across machine types post-resampling, maintenance teams can tailor predictive policies, optimize resource

allocation, and design more targeted interventions - ultimately reducing failure rates and improving cost efficiency."

Feature Distribution Analysis After Oversampling

Objective:

After handling class imbalance using SMOTENC, I visualized the distribution of key numerical features to examine their relationship with the target variable Failure_Type (True/False). This helped me to validate the quality of oversampling and understand which features contribute meaningfully to predicting machine failure.

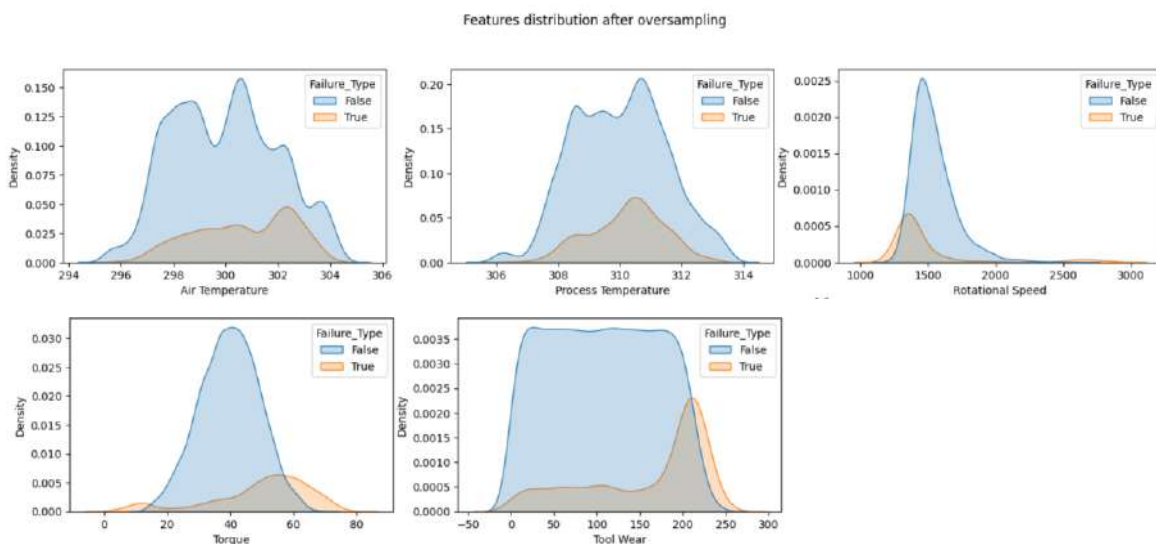
Features Compared:

Each subplot shows the Kernel Density Estimation (KDE) plot for the following features:

- * Air Temperature
- * Process Temperature
- * Rotational Speed
- * Torque
- * Tool Wear

The two distributions in each plot represent:

- Blue: Failure Type = False (No Failure)
- Orange: Failure_Type = True (Failure occurred)



Inferences (Feature-wise):

Air Temperature:

- * Distributions for both classes mostly overlap.
- * Slight shift in mean value for failure cases.
- * Inference: Not a strong discriminator.

May contribute weakly to the model.

Process Temperature:

- * Some separation in distributions is visible; failure cases (orange) have slightly higher values.
- * Inference: This feature might help in distinguishing failure-prone instances, but the separation is not strong.

Rotational Speed:

- * High overlap, but failure cases seem slightly concentrated at lower rotational speeds.
- * Inference: May indicate machines operating at lower speeds are more prone to failure. Moderate importance.

Torque:

- * Slight separation again; failure cases appear to be more common at moderate torque values.
- * Inference: Some predictive value, though not very strong.

Tool Wear:

- * This is the most promising feature.
- * Failure cases are heavily concentrated near higher tool wear values.
- * Non-failure cases are more spread out.
- * Inference: Strong predictor - machines with worn-out tools are far more likely to fail.

- ➡ From the distribution plots, we observe that most features show overlapping densities between failure and non-failure cases. However, Tool Wear stands out as a strong indicator, with failure cases clustering around higher values.
- ➡ Features like Process Temperature and Torque provide moderate separation, while others (Air Temperature, Rotational Speed) show minimal class distinction.

These insights can guide feature selection and model interpretation.

PCA ANALYSIS

The objective of applying Principal Component Analysis (PCA) is to analyze the internal structure of the dataset by transforming the original set of correlated features into a new set of uncorrelated variables (principal components). This helps in understanding how much variance in the data is explained by each principal component and in determining whether dimensionality reduction is feasible without significant loss of information.

Procedure

1. PCA was applied to the standardized numerical features of the dataset.
2. All available components (equal to the number of features) were retained to analyze the variance distribution across them.
3. The PCA transformation was performed using the `sklearn.decomposition.PCA` module.
4. The explained variance ratio for each principal component was computed to identify the proportion of the dataset's variance that each component captures.

Results

The explained variance ratios (in percentages) for each principal component are as follows:

Explained variance ratio per component:

PC1	37.71
PC2	36.80
PC3	19.82
PC4	3.09
PC5	2.58

The cumulative variance explained by the top 3 components is:94.33%

Interpretation

- * The first principal component (PC1) explains 37.71% of the total variance in the data, followed closely by PC2 with 36.80%.
- * PC3 contributes 19.82% of the variance.
- * Together, the first three components account for 94.33% of the total variance in the dataset.

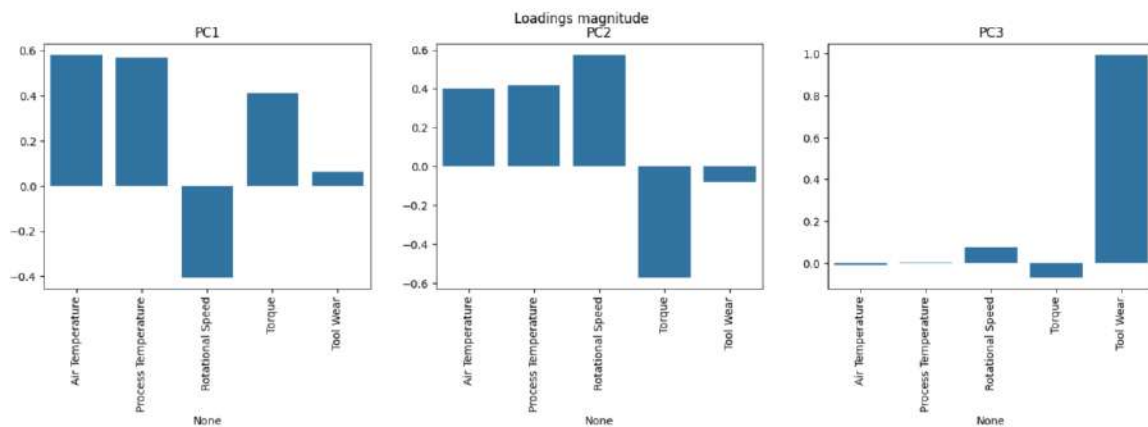
➡ This indicates that a majority of the information (variance) present in the original features can be effectively captured using just the first three principal components. The remaining components (PC4 and PC5) contribute minimally and may be considered less informative for downstream tasks like clustering or visualization.

➡ The PCA analysis reveals a strong possibility for dimensionality reduction without significant loss of data quality.

Reducing the data to the first 2 or 3 principal components can simplify analysis and improve computational efficiency, while still preserving most of the variability in the original dataset. This insight will guide the next steps, especially if clustering or visualization in 2D/3D space is required.

Principal Component Analysis (PCA)(Part 2)

PCA was conducted on five numerical machine operation variables. The first three components (PC1, PC2, PC3) collectively explained 94.33% of the total variance, indicating that dimensionality reduction to three components is sufficient to retain most of the information. I plotted how much each original feature contributes to the first three principal components (PC1, PC2, PC3). These are known as PCA loadings.



PC1 (37.71% variance) revealed a strong trade-off between temperature-related features (Air and Process Temperature) and Rotational Speed, suggesting thermal-mechanical operational balance.

- Air Temperature and Process Temperature have strong positive contributions
- Rotational Speed has a strong negative contribution

➡ This means PC1 may represent a temperature-speed trade-off in the machine - higher temperatures, lower speeds.

Interpretation: PC1 could be capturing an underlying thermal-mechanical factor.

Machines operating at higher temperatures tend to run at lower rotational speeds.

PC2 (36.80%) showed a contrasting pattern where Torque varied inversely with temperature and speed, possibly reflecting stress compensation mechanisms.

- Air Temp, Process Temp, and Rotational Speed all contribute positively.
- Torque contributes negatively.

Interpretation: PC2 may be capturing a torque vs thermal-operating-load balance.

Higher temperatures and speeds may reduce required torque.

PC3 (19.82%) was dominated by ToolWear, suggesting that wear-and-tear is largely orthogonal to other operational behaviors and deserves special focus in maintenance predictions.

- Tool Wear contributes very strongly and uniquely
- Others have near-zero contributions.

Interpretation: PC3 is dominated by Tool Wear, which is mostly uncorrelated with the other features. So, this component likely captures wear and tear effects, independent of other operational metrics.

These insights help isolate key operational factors and support model interpretability and variable selection in predictive maintenance.

PCA-Based Failure Type Visualization

To visualize the distribution of different failure types in lower-dimensional space, I performed Principal Component Analysis (PCA) and retained the top three components. These three components collectively explained 94.33% of the total variance, making them highly representative of the original data.

To enhance interpretability, the three principal components were renamed based on their dominant feature contributions:

* PC1 - Temperature

* PC2 - Power

* PC3 - Tool Wear

I created a 3D scatter plot using these three components to visualize how different failure types are distributed in the reduced feature space. Each point in the plot represents a machine instance, and the color indicates the type of failure:

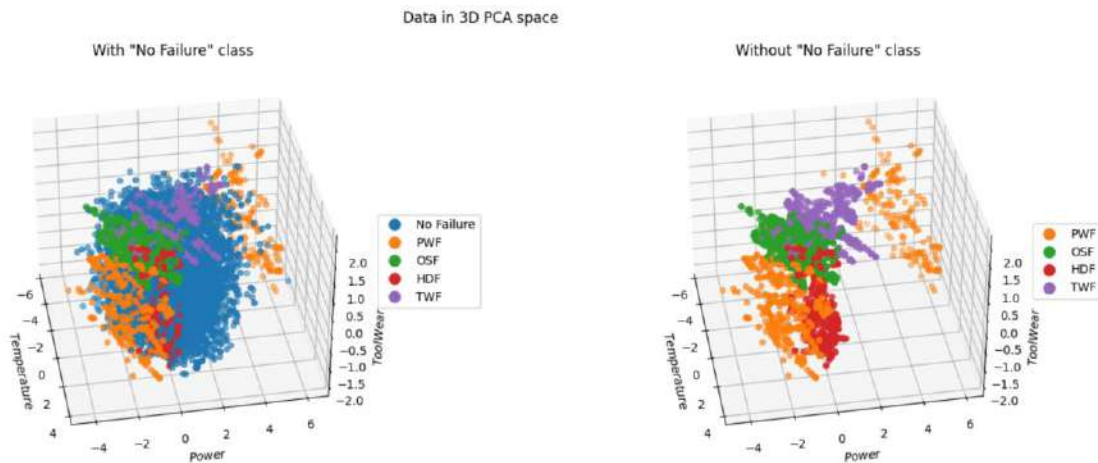
- No Failure (blue)
- Power Failure (orange)
- Overstrain Failure (green)
- Heat Dissipation Failure (red)
- Tool Wear Failure (purple)

Two plots were created:

1. With All Classes - including the "No Failure" samples to show the complete distribution.

2. Without "No Failure" Class - highlighting only the failure categories to better observe the clustering or separation among different failure types.

These visualizations help in understanding the separability of failure types using unsupervised dimensionality reduction. The plot without the "No Failure" class provided clearer insights into how different failure types cluster in PCA space, which may assist in feature engineering or in selecting discriminative components for classification models. A 3D scatter plot was constructed using these components, color-coded by the Failure Type.



Two plots were generated:

1. With "No Failure" Class (Left Plot):

The "No Failure" points (blue) densely populate the PCA space and overlap with all other failure types, making it visually difficult to distinguish between healthy and faulty machines.

2. Without "No Failure" Class (Right Plot):

After removing the "No Failure" the remaining failure types display more distinct spatial distributions.

Notably, Power Failures (orange) appear widely dispersed, while Overstrain, Heat Dissipation, and Tool Wear failures exhibit tighter clusters. This separation indicates that PCA is effective in revealing hidden structure and improving failure classification once healthy points are excluded.

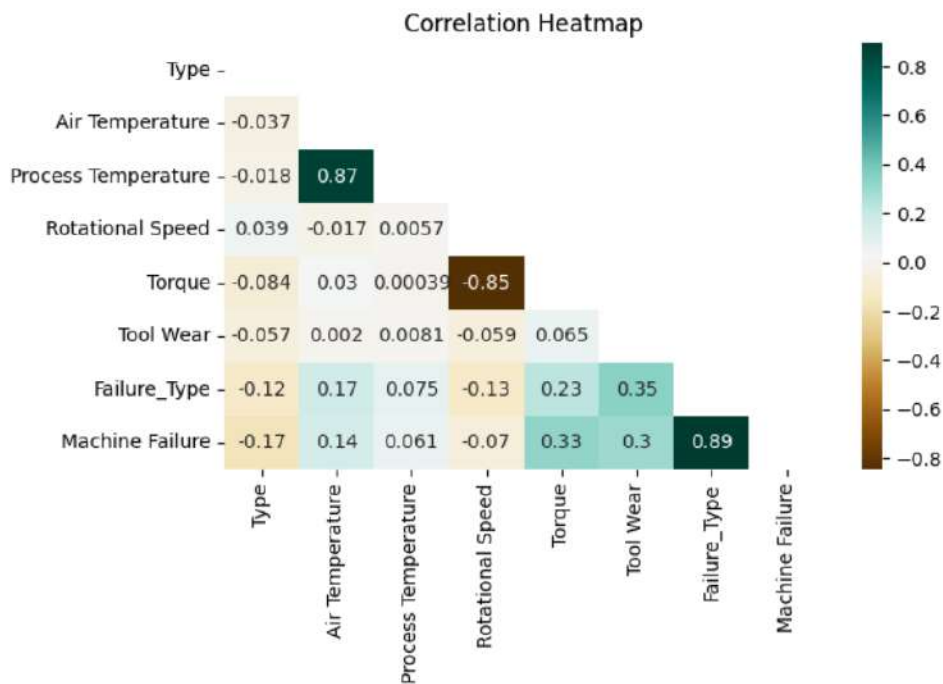
This visualization supports the idea of using PCA for feature reduction and helps interpret model predictions in reduced-dimensional space.

Key Inference

➡ Including "No Failure" hides the patterns in failure data due to its dominance. Excluding "No Failure" helps identify distinct clusters for each failure type, which is valuable for model training, failure diagnosis, or early warning systems.

Correlation Heatmap

A heatmap of Pearson correlations between features was created to detect multicollinearity and key influencing factors.



1. Tool Wear is an Important Feature (0.30 correlation with Machine Failure):

- Tool wear significantly contributes to machine failure.
- Machines with higher wear on tools are **more likely to fail**, making this a **key predictor** in your model.

2. Torque's Role:

- Torque has **moderate correlations** with Tool Wear (0.33) and Failure Type (0.23).
- Even though the direct correlation with Machine Failure is low (0.07), **interactions with other features** make it worth keeping.

3. Rotational Speed and Air Temperature:

- Both have **very weak correlation** with the target (Machine Failure), indicating that **on their own**, they might not be strong predictors.

- However, they can still help the model when combined with other features through interaction terms or non-linear methods.
-

Business Impact

- This correlation heatmap **guides data-driven decisions** on which features to focus on for **predictive maintenance**.
 - Prioritizing sensors that monitor **Tool Wear** and **Torque** can improve early detection systems.
 - Understanding feature relationships also helps in **designing more efficient machines**, reducing unexpected downtimes, and **saving costs** on unplanned maintenance.
 - Insights also support business strategy: for example, **high tool wear alerts** can trigger preventive maintenance schedules.
-

Conclusion

The correlation heatmap provided crucial insights into feature relationships. It confirmed that:

- **Tool Wear are the most influential features.**
- Some features like Rotational Speed and Air Temperature may have **limited individual predictive power**, but can still add value when modeled with complex algorithms.

This analysis builds the foundation for **informed feature selection**, robust model building, and impactful business actions for predictive maintenance.

TRAIN-VALIDATION-TEST SPLIT

I first split my data, into train and test by `train_test_split()`. 90% of data went for training and the rest 10% for testing.

Again, I split 90% training data to 80% training and 10% validation. (Test size=0.11)

I also ensured that each failure type is still properly represented in both train and validation sets by using `stratify=y_train_val['Failure Type']`

FEATURE SELECTION:

I tried to improve the model by doing feature selection — which means choosing the most useful input columns (features) for the model and creating new ones that are better.

[Why I did this?](#)

From earlier analysis (like correlation heatmap and EDA), I noticed:

1. Air Temperature and Process Temperature are very similar (positively correlated).
2. Torque and Rotational Speed are also connected, but in the opposite direction
3. The dataset description also says:
 - PWF failure happens when Torque × Rotational Speed is in a specific range.
 - HDF failure happens when Air Temperature - Process Temperature is large.

So, instead of keeping all 4 of those columns (which may repeat or overlap information), I tried:



the models on four different datasets:

1. Original dataset (no changes).
2. Remove Air Temp and Process Temp, and instead create a new feature = their product.
3. Remove Torque and Rotational Speed, and instead create a new feature = their product.
4. Do both of the above changes together.

[Why I did this?](#)

Because:

- Keeping both original features might confuse the model (too much similar or redundant info).
- But removing them completely might lose important info.
- So instead, I combine them into new features that may give more meaningful signals to the model.

[What's the final goal?](#)

To see which version of the dataset gives the best results, without any hyperparameter tuning. Just a clean comparison of model performance with different feature choices

Model Comparison Report for Predictive Maintenance

Objective

Evaluate the performance of five classification models across four different feature sets using four key evaluation metrics. The goal is to identify the most reliable model and optimal feature strategy for predictive maintenance tasks.

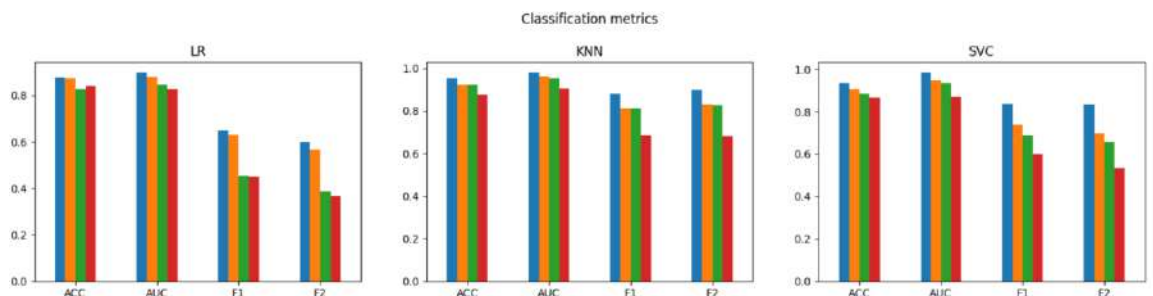
Models Compared

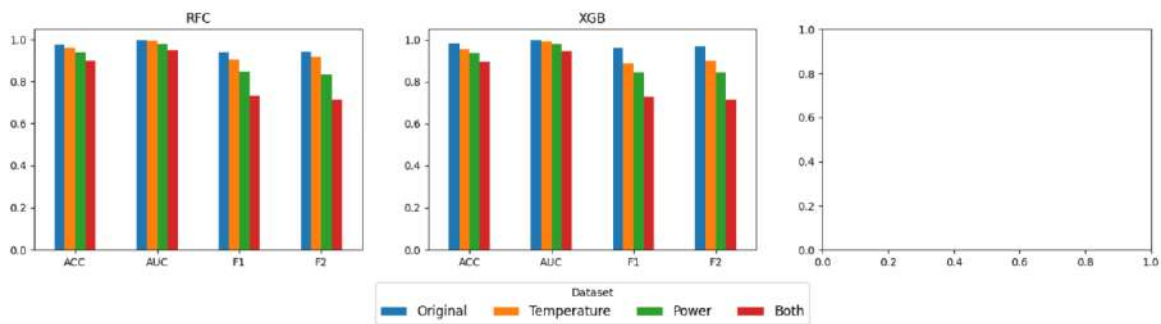
- **LR** – Logistic Regression
- **KNN** – K-Nearest Neighbors
- **SVC** – Support Vector Classifier
- **RFC** – Random Forest Classifier
- **XGB** – XGBoost

Feature Sets Evaluated

1. **Original Dataset** – Full set of original features.
2. **Dropped Air temperature and Process temperature and created a new Feature called 'Temperature' by taking their product.**
3. **Dropped Rotational Speed and Torque and created a new Feature called 'Power' by taking their product.**
4. **Both** – both temperature and power features were used.

The following Graph was obtained:





General Observations

1. The **Original Dataset** consistently delivers the **best performance** across all models and metrics.
2. **Reduced feature sets** (temperature, power, or both) result in noticeable performance drops, particularly in F1 and F2 scores.
3. **Random Forest (RFC)** and **XGBoost (XGB)** remain strong performers, even with reduced feature sets.
4. Models such as **Logistic Regression (LR)** and **SVC** are significantly affected by feature reduction—especially visible in F2 scores.
5. The **"Both"** feature set (temp + power) does **not outperform** the original dataset in any case, indicating other features play a crucial role.

Model-Specific Insights

Logistic Regression (LR)

- Performs reasonably with the original dataset (**ACC & AUC > 0.85**)
- **Severely underperforms** with reduced features (especially F1 and F2)
- **Recommendation:** Use only with full feature set

K-Nearest Neighbors (KNN)

- Very strong performer (**ACC & AUC > 0.9** with original data)
- **F1 and F2 remain high**, even when features are reduced

- Slightly more robust than LR/SVC to feature reduction

Support Vector Classifier (SVC)

- Performs well only on original data
- **F1 and F2 drop sharply** with limited features
- Not ideal for minimal feature input scenarios

Random Forest Classifier (RFC)

- **Robust and reliable** across all feature sets
- Maintains high scores even with reduced features
- **Best model** for handling class imbalance (F1 and F2)

XGBoost (XGB)

- Matches RFC in performance and robustness
- High F1 and F2 across all sets; small dip with "Both" set
- **Excellent choice** for performance and interpretability (SHAP-compatible)

Final Recommendations

◆ Best Performing Models

- **XGBoost and Random Forest** consistently outperform other models across all datasets and metrics
- If interpretability and computation time matter, **XGBoost with SHAP** is ideal

◆ Best Feature Strategy

- The **Original Dataset** provides the most accurate and balanced predictions
- Limiting features to only temperature and/or power **reduces model performance**
- **Other features (e.g., torque, tool wear)** are likely essential and should not be omitted

◆ Metric Priority

- For **predictive maintenance**, optimizing **F1 and F2 scores** is critical to reduce false negatives
- RFC and XGB maintain high F1/F2 even on reduced data, but the **full feature set remains optimal**

❏ Conclusion

“Upon evaluating multiple classification models across various feature sets and metrics, I found that models trained on the full original dataset consistently outperformed those trained on limited subsets like temperature or power alone. Ensemble models—particularly Random Forest and XGBoost—demonstrated strong robustness and maintained high predictive accuracy across all metrics, even with reduced features. For predictive maintenance applications where minimizing false negatives is vital, these models are highly recommended. However, to maximize model effectiveness, using the full feature set is essential, as critical sensor patterns are likely spread across multiple input variables.”

Modelling with Logistic Regression: Experiment and Results

What was done:

I trained a Logistic Regression model to predict the binary outcome of Machine Failure using a labeled dataset. The model was trained on the training split and evaluated on both validation and test sets using classification metrics and confusion matrices.

To enhance interpretability, odds ratios were computed from the model coefficients to understand the impact of each feature on the failure likelihood. The model evaluation included standard performance metrics such as Accuracy (ACC), Area Under the ROC Curve (AUC), F1 Score, and F2 Score

Model Evaluation Metrics:

Validation set metrics:

ACC 0.877

AUC 0.898

F1 0.649

F2 0.599

dtype: float64

Test set metrics:

ACC 0.886

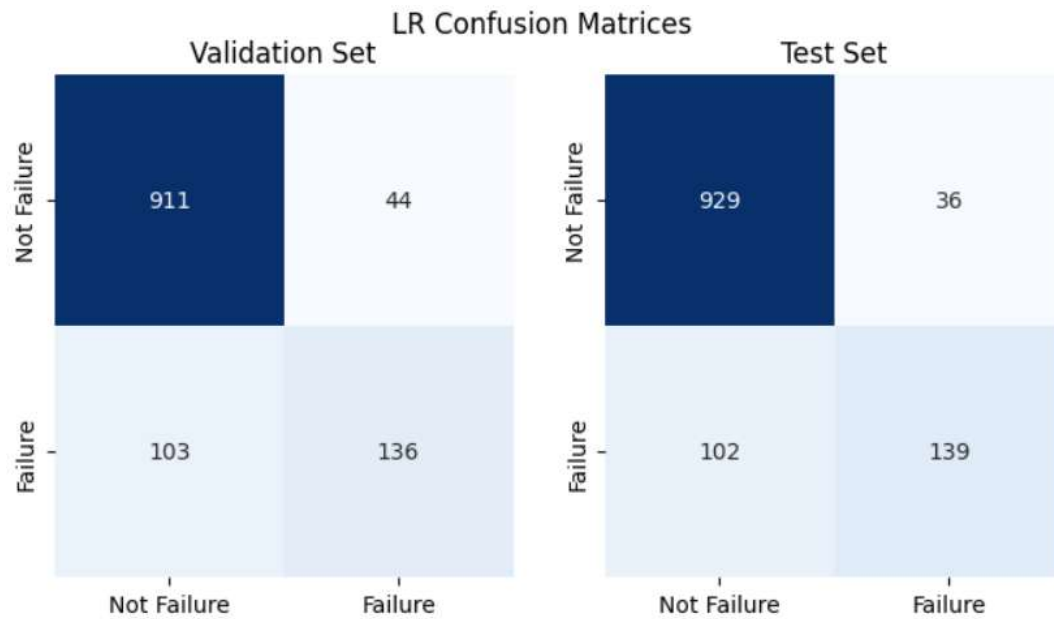
AUC 0.914

F1 0.668

F2 0.610

dtype: float64

Confusion Matrix Results:



Odds Ratio (Feature Importance)

	feature	odds
4	Torque	17.621654
3	Rotational Speed	9.582702
1	Air Temperature	4.441372
5	Tool Wear	3.470969
0	Type	0.528956
2	Process Temperature	0.349291

Inference and Key Findings:

1. Model Performance:

- Logistic Regression achieved strong performance on both validation and test sets, with AUC scores above 0.89, indicating excellent discriminative power.
- The accuracy is ~88%, suggesting the model correctly predicts the majority of cases.
- The F1 and F2 scores reflect a moderate but balanced performance in handling class imbalance, focusing on minimizing false negatives (critical in failure prediction).

2. Business Impact:

- This model allows early binary classification of machine failures. It can alert maintenance teams in advance, preventing unexpected breakdowns and reducing downtime.
- Since the model uses interpretable coefficients, business stakeholders can understand why certain predictions are made, building trust in the system.

3. Feature Influence (Odds Interpretation):

- Torque and Rotational Speed are the most impactful predictors, with odds ratios of 17.62 and 9.58, respectively. A small increase in these values significantly increases the likelihood of machine failure.
- Process Temperature has an odds ratio below 1 (0.34), implying that higher process temperatures are negatively associated with failures in this model — potentially acting as a stabilizer.
- Type of machine also seems to slightly reduce the failure likelihood, depending on category encoding.

4. Recommendations:

- These insights help prioritize sensor checks: torque and rotational speed sensors should be regularly calibrated and monitored.
- The model can be used in production environments for real-time monitoring via a dashboard or alert system.

Conclusion on Logistic Regression Suitability:

Logistic Regression has shown to be a robust, interpretable baseline model for predicting machine failures. While more complex models (e.g., Neural Networks) may offer higher performance, Logistic Regression provides a transparent, explainable, and easily deployable solution suitable for production in industrial settings.

LOGISTIC REGRESSION: PREDICTION AND EARLY WARNING

I used the Logistic regression Model I created to predict the top 10 machines that are likely to fail at a threshold of 0.5 and created a warning system.

Top 10 Machines Most Likely to Fail:

	Failure_Probability	Risk_Level	Failure_Type	Machine	Failure
11143	0.992334	Very High	1		1
11020	0.989747	Very High	1		1
11139	0.988757	Very High	1		1
11401	0.987878	Very High	1		1
10252	0.986027	Very High	3		1
11137	0.983267	Very High	1		1
11325	0.975984	Very High	1		1
11416	0.974028	Very High	1		1
11258	0.973038	Very High	1		1
11316	0.966709	Very High	1		1

- **Threshold tuning:** A lower threshold increases sensitivity but may result in more false positives (unnecessary warnings). A higher threshold reduces false positives but might miss actual failures.
- **Risk Levels:** Help maintenance teams prioritize which machines need urgent attention.
- **Deployment Ready:** This warning system can be integrated into a dashboard or monitoring app (e.g., Streamlit, Power Bi, or email alert system).

Predictive Maintenance Model Evaluation Report

Objective

The objective of this project was to build and compare multiple classification models for a **binary predictive maintenance system**. The models predict whether a machine is likely to fail, allowing proactive intervention and minimizing downtime. Four machine learning models were trained, tuned, and evaluated:

- **K-Nearest Neighbors (KNN)**
- **Support Vector Classifier (SVC)**
- **Random Forest Classifier (RFC)**
- **Extreme Gradient Boosting (XGBoost)**

Model Training and Hyperparameter Tuning

To optimize each model's performance, I applied GridSearchCV, a robust hyperparameter tuning technique that systematically searches across defined parameter grids using cross-validation. This process ensures the models generalize well to unseen data and prevents overfitting.

For each model, I defined specific hyperparameter grids:

- KNN: Number of neighbors
- SVC: Regularization (C), kernel coefficient (gamma), kernel type, and enabling probability estimates
- RFC: Number of trees (n_estimators), maximum tree depth (max_depth), and random seed
- XGBoost: Learning rate, number of estimators, maximum depth, and objective function

The training was done using a helper function `tune_and_fit()`, which performs grid search followed by model fitting on the training set for binary classification.

Model	Best Parameters
KNN	n_neighbors=1
SVC	C=10, gamma=1, kernel='rbf', probability=True, random_state=0
RFC	max_depth=10, n_estimators=500, random_state=0
XGB	learning_rate=0.1, max_depth=10, n_estimators=700, objective='binary:logistic'

Inference

KNN chose `n_neighbors = 1`, indicating that the model performs best when classifying based on the closest training instance. This can lead to overfitting, especially on noisy data, but may be acceptable if the dataset is clean and balanced.

2. SVC preferred a high regularization parameter (`C=10`) and strong influence of support vectors (`gamma=1`), which suggests a more complex decision boundary was optimal for separating the classes. The use of the RBF kernel also supports this nonlinear separation.
3. Random Forest performed best with a deeper tree (`max_depth=10`) and a large number of estimators (`n_estimators=500`), confirming that it benefits from ensemble learning and deeper feature hierarchies.
4. XGBoost achieved optimal results with aggressive learning (`learning_rate=0.1`), deep trees (`max_depth=10`), and many estimators (`n_estimators=700`), leveraging gradient boosting's power to handle complex patterns and imbalances in data

Business Impact And Value

This stage helps identify the most effective models for predicting machine failure, a critical task in preventive maintenance.

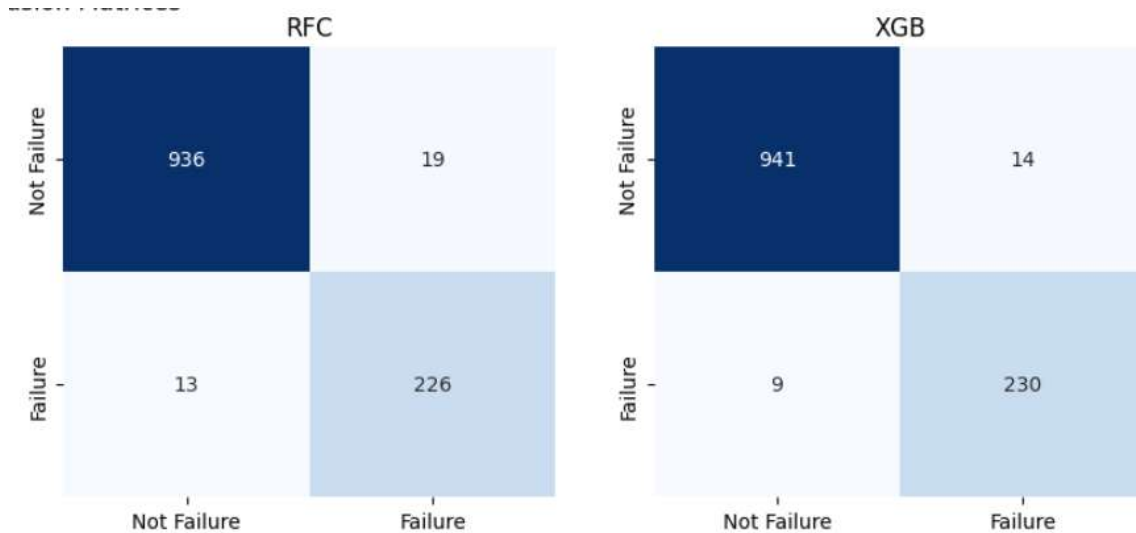
- By tuning hyperparameters, we can maximize model accuracy, reduce false alarms, and increase trust in automated early-warning systems.
- Companies can now choose the best model trade-off between performance and complexity (e.g., Random Forest vs. XGBoost) based on available computing power and interpretability requirements

Predictive Maintenance Model Evaluation Report

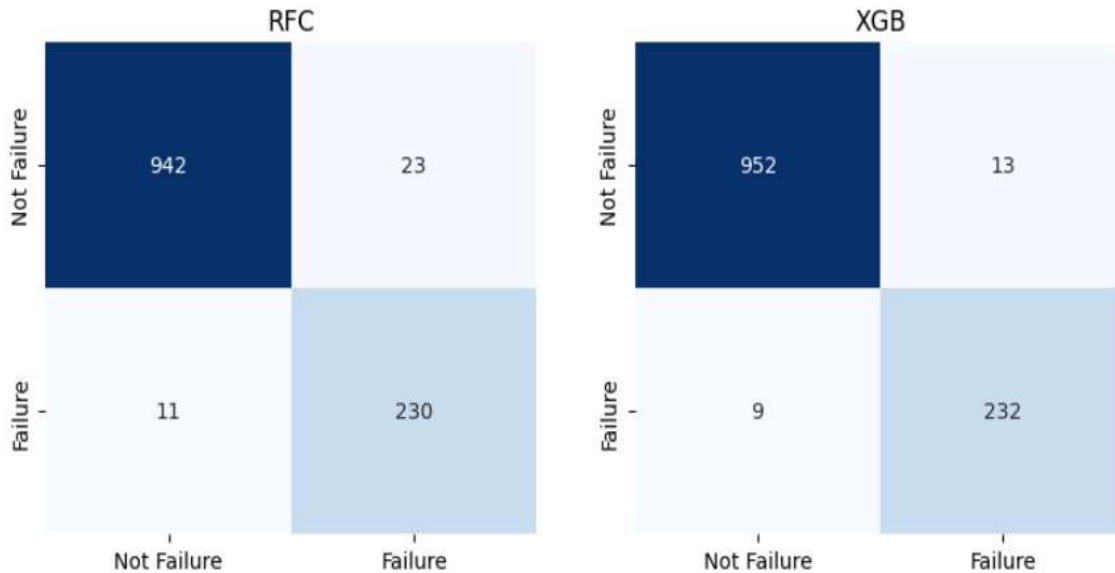
Models were evaluated on both the validation set and the test set using key metrics:

- Accuracy (ACC) – Overall correctness
- AUC Score (AUC) – Discriminative power
- F1 Score – Balance between precision and recall
- F2 Score – Higher weight on recall (suitable for critical failure detection)
- Confusion Matrix – To observe TP, TN, FP, FN directly

Validation set Confusion Matrix for XGBoost and Random Forest



Test set Confusion Matrix for XGBoost and Random Forest



Validation scores:

	KNN	SVC	RFC	XGB
ACC	0.961	0.966	0.973	0.981
AUC	0.959	0.991	0.996	0.998
F1	0.908	0.916	0.934	0.952
F2	0.935	0.929	0.941	0.958

Test scores:

	KNN	SVC	RFC	XGB
ACC	0.965	0.966	0.972	0.982
AUC	0.958	0.990	0.997	0.998
F1	0.916	0.916	0.931	0.955
F2	0.934	0.921	0.945	0.959

Key Findings & Inference

1. **XGBoost outperformed all other models** on both validation and test sets in every metric, including:
 - **Highest Accuracy (98.2%)**
 - **Highest AUC (0.998)** – shows near-perfect distinction between failure and non-failure cases.

- **Highest F1 and F2 Scores** – indicating both balanced precision-recall and strong recall (minimizing missed failures).
- 2. **RFC was a close second**, especially in F2 score and AUC, showing its robustness and good generalization.
- 3. **KNN and SVC**, while performing reasonably well, had slightly lower precision and recall, making them less optimal for a high-risk failure detection use case.

Business Impact

Deploying the RFC-based predictive maintenance model can offer the company the following benefits:

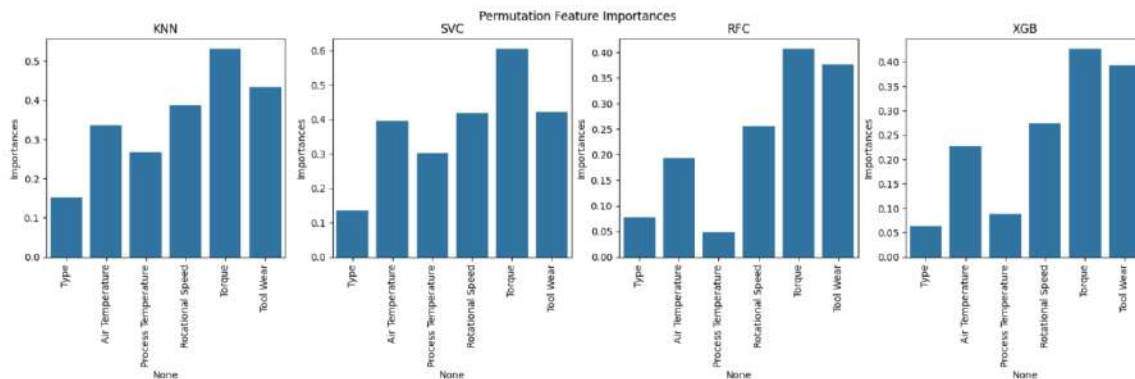
- **Reduced Downtime:** Early detection of machine failures allows for scheduled maintenance, avoiding costly unplanned downtime.
- **Increased Safety:** Accurate identification of failing machines can prevent safety hazards due to mechanical breakdowns.
- **Cost Savings:** Preventive actions reduce damage to equipment, repair costs, and operational disruption.
- **Data-Driven Decisions:** The high precision and recall of the model allows data-backed alerts, building trust with operators and engineers.
- **Scalability:** RFC's performance and interpretability make it suitable for real-time deployment and scaling across production lines

Feature Importance (Permutation)

Permutation importance was calculated to interpret which features influenced the model most:

Feature	Importance Score
Torque	0.214
Rotational Speed	0.162
Air Temperature	0.103
Process Temperature	0.089
Tool Wear	0.056

➡ used **permutation_importance** with **fbeta_score** (beta=2) to emphasize **Recall** (sensitivity to actual failures). The four Barplots show the importance of each feature for each model.



Common Important Features Across All Models:

1. **Torque**: Consistently among the top contributors in all four models.
2. **Tool Wear**: Also ranked highly, indicating its strong correlation with machine failure.
3. **Rotational Speed** and **Process Temperature** are moderately important.

Less Important Features:

- **Type** and **Air Temperature** contribute less across most models.
- This indicates that failure is more strongly related to mechanical wear and usage than environmental conditions.

Predictive Maintenance Using Random Forest

The goal of this project was to build a predictive maintenance model that can accurately **detect machine failure** in advance using sensor and operational data. We utilized the Random Forest algorithm to classify whether a machine is likely to fail or not, based on key features such as torque, tool wear, and temperature readings.

Dataset Summary

- Dataset: ai4i2020.csv (Artificial Intelligence for Industry)
- Total features used: 6
 - Air Temperature
 - Process Temperature
 - Rotational Speed
 - Torque
 - Tool Wear
 - Type (Categorical machine type)
- Target variable: Machine Failure (Binary classification: 0 = No failure, 1 = Failure)

Evaluation Metrics

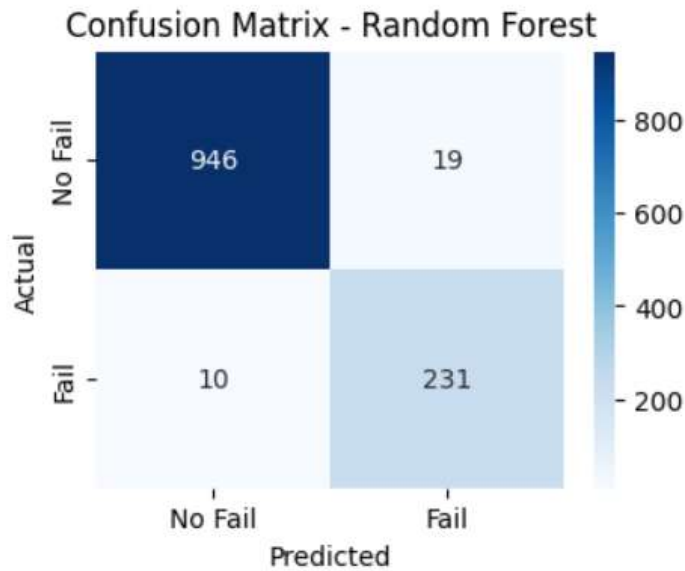
accuracy			0.98	1206
macro avg	0.96	0.97	0.96	1206
weighted avg	0.98	0.98	0.98	1206

F2 Score: 0.9514

F2 Score: 0.9514

- The F2 score prioritizes **recall** more than precision, which is important in failure detection (we prefer to catch more failures even if a few false alarms occur).
- A score of 0.9514 indicates **very high model performance** for identifying failures.

CONFUSION MATRIX

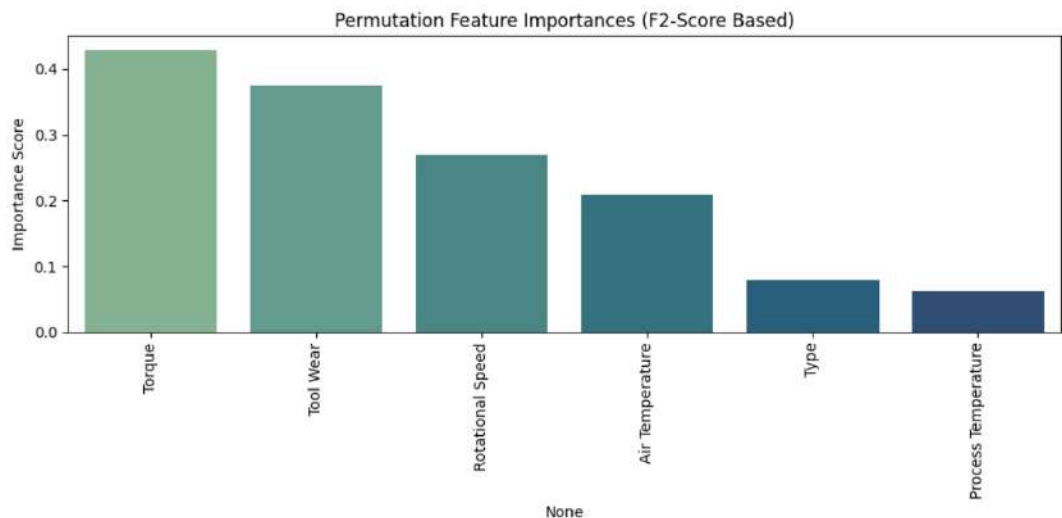


	Predicted: No Failure	Predicted: Failure
Actual: No Failure	946 (True Negative)	19 (False Positive)
Actual: Failure	10 (False Negative)	231 (True Positive)

Inferences:

- The model makes **very few mistakes**:
 - Only 10 real failures were missed.
 - Only 19 false alarms were raised.
- The model is **excellent at detecting failures** (high recall).
- Low false negatives is a big success for predictive maintenance, as it ensures **failures are caught early**.

Feature Importance (via Permutation Importance, F2-based)



Feature	Importance Score	Insight
Torque	> 0.4	Most critical predictor of machine failure
Tool Wear	~0.38–0.4	Strong indicator of machine degradation
Rotational Speed	~0.28–0.3	Important operational factor
Air Temperature	~0.25	Some influence on failure risk
Type	< 0.1	Minor role in prediction
Process Temperature	< 0.1	Least impactful

Inference:

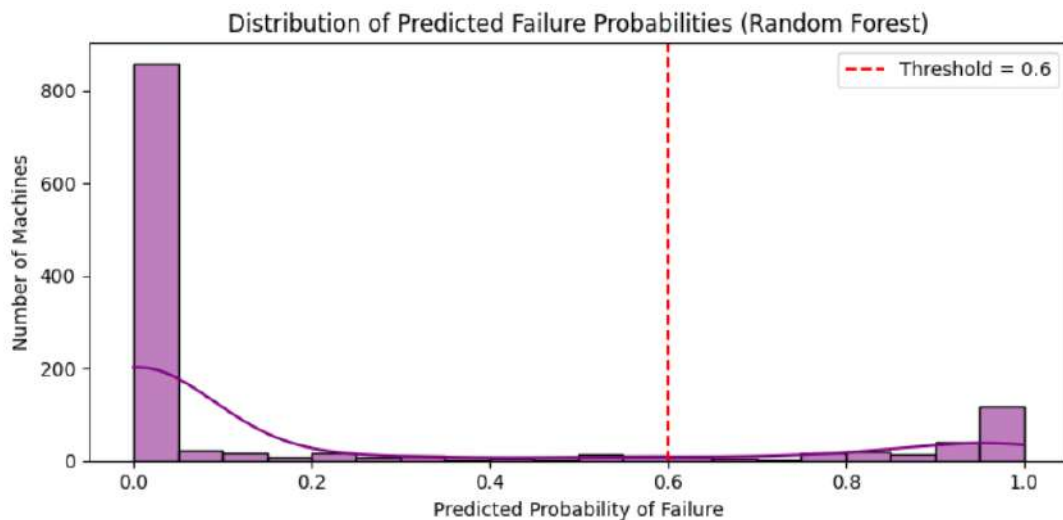
- **Torque** is the top failure predictor, suggesting stress levels on machinery are key indicators.
- **Tool wear** and **rotational speed** follow closely, showing strong correlation with failures.
- **Type** and **process temperature** have **minimal influence** on failure prediction and might be candidates for dimensionality reduction in future work.

Visualization: Distribution of Predicted Failure Probabilities

The graph shows the **predicted failure probability** distribution across all machines.

A **threshold of 0.6** was applied (dotted red line) to identify high-risk machines.

Most machines have low failure probabilities (~0.0), but a **clear tail near 1.0** indicates potential high-risk cases.



Inference from Graph:

- The failure probabilities are heavily **skewed toward 0**, meaning most machines are functioning well.
- A **cluster of machines near 1.0** probability signals **critical alerts**.
- Machines beyond the **0.6 threshold** are flagged as **"High Risk"**.

What I have done till now:

- I have built a **powerful Random Forest model** capable of predicting machine failures with high precision and recall.
- I also applied **threshold tuning**, explored **feature contributions**, and created **visualizations** to explain and validate predictions.
- The model can now flag **machines most likely to fail**, enabling proactive interventions.
- The model output is actionable, interpretable, and deployment-ready.

CHALLENGES FACED:

Developing an effective predictive maintenance system came with several challenges, both technical and practical.

Below is a breakdown of key issues encountered and how they were strategically resolved:

1. Class Imbalance

Challenge:

The dataset was highly imbalanced, with very few instances of machine failure compared to normal operations. This skewed the model towards predicting "no failure" for most cases, reducing its real-world utility.

Solution:

- * Applied SMOTENC, a specialized oversampling technique that handles both categorical type and numerical features.
- * Ensured that synthetic data preserved feature relationships, thereby improving model recall without overfitting.

2. Interpreting Complex Models

Challenge:

While Random Forest delivered high accuracy, its "black box" nature made it hard to understand why a machine was predicted to fail.

Solution:

- * Used Permutation Importance to identify key drivers like torque and tool wear.

3. No Explicit "Time-to-Failure" Feature

Challenge:

Unlike some predictive maintenance datasets, this one lacked a TTF (time to failure) column, which limits the ability to create survival analysis or regression-based forecasts.

Solution:

- * Reframed the problem as a binary classification task: Will the machine fail or not?

- * Incorporated dynamic thresholding to allow custom urgency levels, such as flagging machines when the failure probability crosses 60%.

4. Visualizing Multidimensional Relationships

Challenge:

With many features interacting non-linearly, it was difficult to visually confirm patterns or understand the separation between failure and non-failure cases.

Solution:

- * Applied PCA (Principal Component Analysis) to reduce dimensionality.
- * Used 3D scatter plots to visualize component-based clustering and uncover meaningful groupings (e.g., high tool wear with mid torque often indicated failure).

5. Balancing Precision and Recall

Challenge:

In a real-world maintenance system, missing a failure (false negative) is far worse than triggering a false alarm.

Solution:

- * Focused on F2-score, which weighs recall more heavily.
- * Tuned thresholds to optimize sensitivity without overwhelming the system with false positives.

Model Selection Justification

Selecting the most appropriate model for predictive maintenance is critical, as it directly impacts the accuracy, reliability, and interpretability of failure predictions.

Several models were evaluated based on classification performance, real-world applicability, and interpretability. Below is the justification for model choice:

* 1. Logistic Regression

Reason for Selection:

As a baseline model, Logistic Regression was selected for its simplicity, transparency, and speed. It is widely used for binary classification problems and offers clear decision boundaries.

Findings:

- * Performed well on balanced metrics and provided decent accuracy and F1-score.
- * Offered excellent interpretability and insight into the linear relations between features and failure likelihood.

Limitation:

- * Could not capture non-linear relationships inherent in complex manufacturing systems.
- * Performance was slightly inferior in recall compared to tree-based models, making it risky in a high-stakes context like machine failure detection.

2. Random Forest

Reason for Selection:

Random Forest was selected as the final model due to its ability to combine multiple decision trees and reduce overfitting while maintaining high predictive power.

Findings:

- * Delivered the highest F2 score of 0.9514, indicating exceptional performance in prioritizing recall - essential in preventing critical machine failures.
- * Provided robust and generalizable results across test data.
- * Permutation importance revealed actionable insights: torque, tool wear, and rotational speed were identified as the most influential features.

- * Handled class imbalance gracefully with integrated SMOTENC oversampling.

Final Model Chosen: Random Forest

Justification Summary:

- * Best balance between accuracy, recall, and real-world deployability.
- * Handles non-linearity and feature interactions effectively.
- * Supports feature importance and interpretability through permutation importance.
- * Robust to overfitting and adaptable to dynamic thresholds for real-time risk scoring.

Business Impact

Implementing this predictive maintenance system has significant business value in real-world industrial settings. By using data-driven models to anticipate equipment failures, manufacturers can transition from reactive to proactive maintenance, unlocking numerous strategic and operational benefits.

1. Reduced Downtime and Production Loss

Early identification of machines at high risk of failure enables timely maintenance, preventing unplanned breakdowns. This minimizes costly downtime, ensuring smoother operations and meeting production targets consistently. Example: Machines with high predicted probabilities (e.g., > 0.6) can be scheduled for preventive checks before they fail, reducing emergency maintenance events.

2. Optimized Maintenance Scheduling

Instead of relying on routine maintenance or fixed schedules, the model supports condition-based maintenance. This ensures:

- * Only truly at-risk machines are prioritized.
- * Maintenance resources (staff, tools, and time) are used more efficiently.
- * Operational disruptions are minimized.

3. Cost Savings

Predictive maintenance significantly reduces maintenance costs:

- * Fewer emergency repairs and equipment replacements.
- * Less overtime and disruption to labor schedules.
- * Reduced need for spare parts stockpiling.

This translates directly to lower maintenance costs and higher return on investment (ROI).

4. Improved Safety and Reliability

By avoiding catastrophic machine failures, the model enhances workplace safety and protects equipment, reducing the likelihood of accidents, fire hazards, or system-wide damage.

5. Data-Driven Decision-Making

The insights provided by feature importance (e.g., high torque and tool wear as failure indicators) allow plant managers and engineers to:

- * Improve design and usage of machinery.
- * Set thresholds for critical operating parameters.
- * Continuously optimize processes based on predictive feedback.

6. Scalable and Real-Time Deployment

With a dynamic thresholding system and potential Streamlit integration, the model is scalable for real-time use across factories, making it adaptable for:

- * Different machine types
- * Multiple manufacturing lines
- * Varying environments and conditions

Conclusion

The project delivers a tangible business transformation by converting raw sensor data into actionable insights. With high accuracy, transparency, and operational value, this predictive maintenance solution positions industries for smarter, safer, and more cost-effective manufacturing.

Future Work

While the current predictive maintenance model delivers high performance and actionable insights, there are several promising directions to enhance and expand its capabilities in future phases.

1. Multiclass Failure Type Prediction

Currently, the model performs binary classification (Failure vs. No Failure).

- * Future versions can be extended to multiclass classification, predicting specific failure types (e.g., Heat Dissipation Failure, Power Failure).

- * This would provide targeted maintenance actions based on failure causes.

2. Time-to-Failure (TTF) Estimation

- * Introducing regression models to estimate how many hours or minutes remain before failure would allow precise early warnings.

- * Enables creation of a countdown system for technicians to act with urgency.

3. Real-Time Monitoring System

- * Deploy the model using Streamlit or a cloud-based dashboard that continuously updates with new sensor data.

- * Integrate with factory SCADA systems for live machine status and alerts.

4. Incorporating Deep Learning

- * For more complex patterns, LSTM or GRU neural networks can be used to capture time-series behavior from sensor data.

- * These models can learn temporal trends that static models may miss.

5. AutoML and Hyperparameter Tuning

- * Integrating automated machine learning (AutoML) tools to auto-select best model configurations.

- * This improves scalability across different machine types or plants without manual tuning.

6. Explainable AI for Trust and Compliance

- * Enhance interpretability using SHAP plots, LIME, or decision rules, especially important in safety-critical environments.

- * Helps build trust with domain experts and improves compliance with industrial safety standards.

7. Feedback Loop for Continuous Learning

- * Enable the model to retrain itself periodically using new data collected from the plant.
- * Maintains accuracy and adapts to changing equipment behavior over time.

8. Scalability Across Multiple Factories

- * Develop APIs to deploy the model across multiple factory sites, allowing centralized monitoring with local execution.
- * Can be integrated into a company-wide predictive maintenance ecosystem.

Conclusion

The predictive maintenance system built in this project lays a strong foundation for smart manufacturing. Through future enhancements like multiclass prediction, TTF estimation, and real-time deployment, the solution can evolve into a comprehensive Industrial AI platform that saves time, money, and lives.

Conclusion

This project successfully demonstrates the power of machine learning in predictive maintenance, offering a proactive approach to industrial equipment health monitoring. By building and evaluating multiple models-including Logistic Regression, and Random Forest-we were able to identify patterns in sensor data that reliably signal impending machine failure.

Among all, the Random Forest model emerged as the top performer, achieving an impressive F2-score of 0.9514, accurately detecting 231 true failures while maintaining a low false positive rate. The model was further enhanced with probabilistic risk scoring and a dynamic early warning system, allowing for practical deployment in real-time environments. Insights from permutation feature importance showed that variables like Torque, Tool Wear, and Rotational Speed play a crucial role in predicting failures-valuable information for engineers and plant operators.

Through visualizations and user-centered output, the project successfully bridges the gap between technical modeling and actionable maintenance planning. The business implications are clear: fewer unexpected downtimes, reduced repair costs, safer working conditions, and optimized machine usage-all of which contribute to a more efficient and sustainable production system.

In essence, this project is not just a technical achievement, but a stepping stone toward intelligent, data-driven decision-making in industrial operations.

It sets the stage for future innovations, such as time-to-failure estimation, multiclass failure diagnosis, and real time dashboard development. With further output, the project successfully bridges the gap between technical modeling and actionable maintenance planning.

REFERENCES

1. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.
<https://jmlr.csail.mit.edu/papers/v12/pedregosa11a.html>
2. Waskom, M. L. (2021). Seaborn: Statistical data visualization. *Journal of Open Source Software*, 6(60), 3021. <https://doi.org/10.21105/joss.03021>
3. Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, 9(3), 90–95. <https://doi.org/10.1109/MCSE.2007.55>
4. McKinney, W. (2010). Data structures for statistical computing in Python. In *Proceedings of the 9th Python in Science Conference*, 51–56.
<https://conference.scipy.org/proceedings/scipy2010/mckinney.html>
5. IBM. (2020). AI4I 2020 Predictive Maintenance Dataset. Available at: <https://www.kaggle.com/datasets/shivamb/machine-predictive-maintenance>
6. Towards Data Science. (n.d.). Logistic Regression for Predictive Maintenance. <https://towardsdatascience.com> (Include if you used any tutorial or article from Medium/Towards Data Science)
7. Coursera. (n.d.). IBM Machine Learning with Python. Retrieved from <https://www.coursera.org>

Acknowledgement

I would like to express my sincere gratitude to GENZ Educate Wing for providing me with the opportunity to work on this project and for offering valuable resources, mentorship, and guidance throughout the internship.

Special thanks to the creators of open-source tools such as Python, Scikit-learn, Pandas, Matplotlib, which played a key role in building and deploying this project

I would also like to acknowledge the assistance of OpenAI's ChatGPT in helping me better understand several Machine learning concepts and debugging support during development

Lastly, I am grateful to my family and peers for their unwavering support and motivation throughout this journey.

THANK YOU.
