Comparing Predictive Maintenance Performance Between Synthetic and Real-World Manufacturing Datasets

November 2025

Jack Quillin, Daniel Calle, Daniel Gapper

School of Electrical Engineering and Computer Science
Washington State University
Pullman, WA, USA
jack.quillin@wsu.edu, daniel.calle@wsu.edu, daniel.gapper@wsu.edu

1. Introduction

Predictive maintenance has become a critical capability in modern manufacturing environments, where unexpected equipment failures can lead to costly downtime, reduced production yield, and significant costs. Semiconductor fabrication is one of the most demanding industrial domains where equipment reliability is essential. In this environment, tools often operate 24/7 with strict tolerances, and failures can destroy wafers worth tens of thousands of dollars within seconds. Companies such as Micron, Intel, and TSMC rely heavily on advanced data-driven fault detection to maintain quality and improve throughput, making predictive maintenance a high-value applied machine learning problem. The goal of predictive maintenance is to use sensor and operational data to anticipate failures before they occur, enabling proactive intervention rather than reactive repair.

This project investigates predictive maintenance using two related datasets. First the **AI4I 2020 Predictive Maintenance Dataset**, which contains synthetic yet highly structured and noise-free operational data, and then the **AI4I-PMDI**, a newer dataset that introduces irregularities to simulate real industrial environments (Autran et al., 2024). Since real manufacturing datasets are often proprietary and confidential, synthetic datasets are frequently used for research and model prototyping. Our goal for this project was to determine h**ow well models trained on clean synthetic data generalize to noisy, irregular real-world scenarios.** Addressing this question is crucial for determining whether synthetic data can meaningfully accelerate model development before access to real industrial data is available.

Our motivation for selecting this topic stems from career interests in data science and industrial machine learning, particularly in the semiconductor industry. Predictive maintenance represents a real-world application where engineering, statistics, and computer science converge, making it ideal for gaining practical experience. Additionally, semiconductor companies face ongoing challenges with diverse equipment generations, sensor drift, inconsistent control settings, and system upgrades. Understanding how models handle such variability is essential for reliable deployment.

This project began with training models on the clean AI4I dataset to establish baseline performance. After feature engineering, model training, and evaluation, we then extended the work to the irregular AI4I-PMDI dataset. This adds missing values, noise, control parameters, and system-level differences intended to simulate fleet-level aging. Once this was complete to analyze domain generalization, we implemented a **cross-dataset paired product-ID testing** framework to study the effects of training on one environment and testing on the other. This approach evaluates whether synthetic data is sufficient for building robust models or whether real-world irregularity is required for meaningful generalization.

Initial results demonstrated high performance within each dataset, with ROC-AUC scores ranging from **0.94–0.98** across tree-based models. However, early cross-dataset testing initially produced unstable performance due to naming mismatches and data leakage. After correcting the splitting procedure to enforce paired splitting based on product ID, cross-dataset AUC results showed clear performance differences. Models trained on irregular data tended to generalize more effectively to clean data, showing the importance of real-world noise exposure during training. The results lead us to believe that while synthetic data is useful for model prototyping, exposure to irregular data is necessary for reliable real-world use.

The remainder of this report goes through the details of our exploration and testing along with the conclusions and identified differences.

2. Machine Learning Task

The goal of this project is to develop machine learning models that are capable of predicting machinery failures based on historical and operational data. The task is laid out as a **binary supervised classification problem**, where the model get numerical and categorical input features describing the machine operating conditions and outputs a probability that the machine will fail.

Task Definition

- **Input:** Multivariate sensor and process measurements including torque, rotational speed, temperatures, tool wear, system identifier, and control mode.
- **Output:** A binary prediction for **machine_failure**, where 1 represents failure and 0 represents normal operation.
- **Learning Setting: Supervised classification** that uses labeled data with a known outcome for each sample.
- **Objective:** Maximize the predictive performance based on **ROC-AUC**, confusion matrix metrics, and real-world interpretability.

Predictive maintenance systems must often prioritize the detection of true failures (recall) to prevent downtime. A high false positive rate, however, can waste resources through unnecessary intervention. This means that balancing trade-offs between false negatives and false positives is very important.

Datasets

AI4I 2020 (Clean dataset)

Synthetic dataset with **10,000 rows** and no missing values.
Includes air and process temperature, torque, rotational speed, tool wear, and machine type.

AI4I-PMDI (Irregular dataset)

Enhanced version introducing real-world complexity such as:

- Missing and noisy values
- New **system** parameter (represents equipment instances within a fleet)
- **Control** field with three categorical settings (**A, B, C**)
- Expanded operational variation

These inconsistencies simulate the real industrial conditions that exist with aging equipment, sensor drift, and calibration.

Research Questions

This project investigates the following core questions:

1. **How well do supervised ML models perform on clean synthetic predictive maintenance data?**
2. **How much does performance change when evaluating the same models on irregular noisy data?**
3. **Can a model that is trained on synthetic data generalize to real-world noisy conditions?**
4. **Does training on noisy real-world style data improve robustness when evaluated on clean synthetic data?**
5. **What differences appear between Random Forest, Gradient Boosting, and XGBoost in feature prioritization and performance?**

Key Challenges

Challenge	Description	Project Response
Class imbalance	Failure events represent a small % of samples	AUROC used instead of accuracy, used class_weight for models
Data shift	Structure difference between synthetic and real-world data	Paired ID cross-dataset testing developed
Missing values	Only irregular dataset includes gaps	Mean-imputation applied for each feature

Feature mismatch	Different column names and additional features	Column alignment and engineered features
Data leakage	Rows are shared between datasets from merges	Product ID split to ensure separation

Summary

This project studies the different machine learning models and their robustness for predictive maintenance in industrial environments. This is done by comparing performance on clean and irregular datasets and measuring generalization across dataset conditions. This will allow us to understand whether models trained exclusively on synthetic clean data can reliably transfer to irregular real-world environments and vice versa. Addressing this has direct relevance to real manufacturing deployment where testing time and data access are limited.

3. Technical Approach

This project consists of three main stages starting with **data preparation**, then **model training and evaluation**, and finally **cross-dataset generalization analysis**. The full workflow is designed so that a reader can reproduce the process using the original datasets and the code included in the project repository.

3.1 Data Preparation Pipeline

Both datasets (AI4I 2020 and AI4I-PMDI) were preprocessed using a shared feature-engineering and cleaning pipeline to ensure consistent structure and compatibility. The goal was to create a normalized input space so that models can be compared fairly and effectively.

Preprocessing and Feature Engineering

Steps applied to both datasets:

1. **Drop leakage columns**
   The original AI4I dataset contains multiple columns indicating root cause of failure (TWF, HDF, PWF, OSF, RNF). If kept, the model would simply "cheat" by learning them directly.
2. **Normalize column names**
   Converted to lowercase, removed whitespace, aligned naming between datasets.
3. **Numeric conversion and missing value handling**
   For AI4I-PMDI, missing values were filled using mean imputation to allow calculation of engineered features without losing samples.
4. **Feature Engineering**

New Feature	Formula	Purpose

Temperature difference	**process_temp – air_temp**	**Measures thermal stress**
Angular velocity	**rpm * (2π / 60)**	**Converts rotational speed to rad/s**
Power	**Torque * angular_velocity**	**Mechanical stress indicator**
Wear and torque interaction	**Torque * tool_wear**	**Captures overload failure risk**

5. **Categorical Encoding**
    - Machine Type: **L, M, H → one-hot encoding**
    - Control Setting: **A, B, C → one-hot encoding**
6. **Output Label**
    - Failure encoded as:
      machine_failure = 1 if diagnostic != "no failure" else 0
7. **Final feature matrix**
    - Removed ID columns (keep only for cross-dataset matching)
    - Ensured all feature dtypes numeric for scikit-learn compatibility

---

3.2 Model Training Procedure

Three models were selected to evaluate performance and robustness under different data conditions:

Model	Reason for Choice
Random Forest	Strong baseline, relatively robust against noise and easy to interpret
Gradient Boosting	Sequential learning improves the edge case sensitivity
XGBoost	State of the art gradient boosting with handling for imbalance

All models use **binary cross-entropy based training objectives** and balancing strategies such as class_weight="balanced" or scale_pos_weight.

Training Protocol

```
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
```

)

The **ROC-AUC score** is recorded for each model along with confusion matrix results allowing us to compare and examine performance trade-offs.

3.3 Handling Challenges in the Data

Challenge	Solution
Missing data	Mean imputation + numeric conversion
Naming inconsistencies	Global rename & enforced alignment
Column mismatch between datasets	Automatic alignment via inner-set matching
Label imbalance	Balanced weighting and AUROC metric
Leakage in cross testing	Product-ID paired splitting

3.4 Cross-Dataset Generalization Approach

To evaluate transferability, models were intentionally trained and tested **across** datasets:

Experiment	Purpose
Train clean and then test unclean	Test if synthetic data generalize to real noisy settings
Train unclean and then test clean	Test if exposure to noise helps robustness

A pairing strategy was used to prevent data leakage. Instead of splitting clean and irregular datasets separately, both were merged, and product identifiers were used to ensure that the same machine was never seen in both training and testing simultaneously.

Paired ID Split Strategy

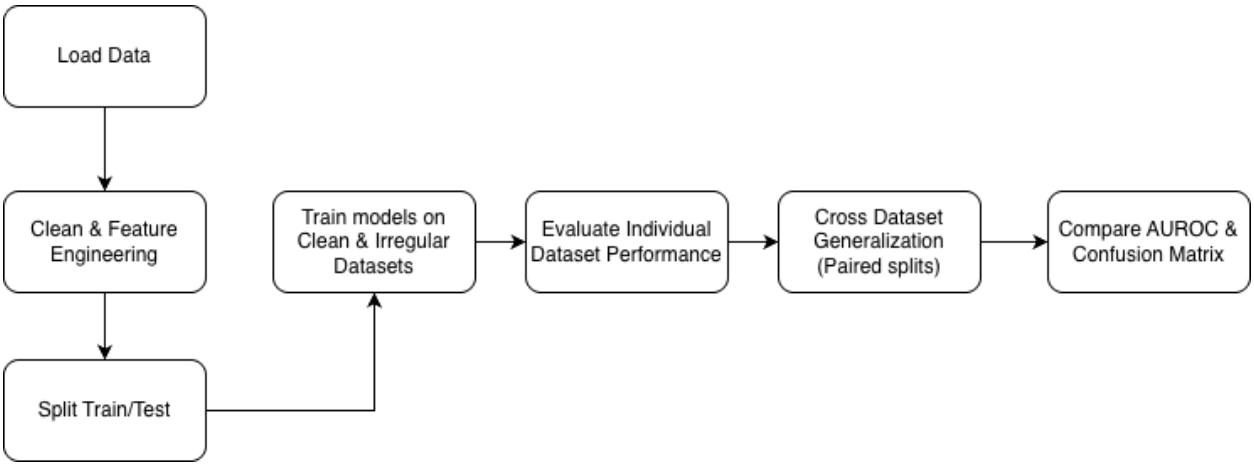
ids = merged["product_id"].unique()

train_ids, test_ids = train_test_split(ids, test_size=0.2, random_state=42)

Then training and testing data were created by selecting rows whose product_id belonged to the appropriate group.

3.5 System Overview Diagram

High-level processing flow



3.6 Implementation Summary

- Data processing: pandas, numpy
- Modeling: scikit-learn, XGBoost
- Evaluation: ROC-AUC, precision/recall, confusion matrix
- Visualization: matplotlib
- Runtime Environment: Local CPU, Python 3.12

The process is reproducible through the main.py execution script, which handles dataset loading, model training, performance comparison, and metrics output.

4. Evaluation Methodology

A structured evaluation methodology was used to measure how well machine learning models perform predictive maintenance classification under different data quality conditions and to evaluate their generalization capabilities when transferred across datasets. This section describes the datasets, preprocessing, train–test strategy, evaluation metrics, baselines, and fairness considerations used to ensure reliable results.

4.1 Dataset Description and Sources

Two datasets were used: a clean, widely studied benchmark dataset and a more realistic dataset containing industrial irregularities.

Dataset	Source	Size	Description
AI4I 2020 Predictive Maintenance Dataset	UCI ML Repository	10,000 samples	Clean synthetic dataset that models

			process variables and machine failures without any missing values.
AI4I PMDI (Irregular)	Autran et al. (2024) research dataset	10,000 samples	Adds realistic noise such as sensor drift, missing values, and additional system and control parameters

Both datasets simulate the performance of a machine used in industrial environments and contain target labels, identifying whether a machine failure happens. The clean dataset estimates ideal laboratory-style measurements, whereas the PMDI dataset introduces realistic measuring imperfections as seen in manufacturing environments.

4.2 Data Cleaning and Preparation

Preprocessing was performed identically on both datasets to avoid any unfair advantage. The following steps were included:

1. **Removed leakage columns**, such as failure-cause identifiers that are present only in AI4I 2020.
2. **Imputed missing values** using mean imputation for numeric fields.
3. **Generated engineered features** representing domain-specific mechanical relationships.
4. **One-hot encoded categorical fields**, including machine type and control mode.
5. **Converted all values to numeric** and standardized column names.
6. **Dropped non-useful or duplicate ID fields**, except when required for paired splitting in cross-dataset evaluation.

Preprocessing ensures the models evaluate predictive capability rather than relying on artifacts or educated guesses.

4.3 Train / Validation / Test Strategy

Individual dataset evaluations were conducted using a standard supervised learning training pipeline:

train_test_split(X, y, test_size=0.2, random_state=42)

A fixed random seed ensures reproducibility. For the cross-dataset experiment, a more specialized split was used to prevent data leakage:

- The two datasets were merged using **Product_ID**
- The available Product_ID values were divided into train and test groups
- Both clean and irregular datasets were filtered using the same train/test ID sets

This ensures that **no machine appears in either the train or test sets**, preventing artificially inflated results.

Cross-dataset experiment conditions

Scenario	**Description**
Train clean -> Test unclean	**Tests robustness of synthetic trained models against realistic noise**
Train unclean -> Test clean	**Tests if noise exposure improves generalization**

4.4 Evaluation Metrics

Multiple performance metrics were used to assess different aspects of prediction quality.

Metric	Real-world importance	Usage
ROC-AUC	Probability of ranking true failures above non-failures	Primary evaluation metric
Confusion matrix	Shows trade-off between false alarms and missed failures	Insight into reliability
Precision/Recall/F1	Significant when failures are rare and costly	Impact trade-off

Why ROC-AUC?

We decided on using ROC-AUC as the primary metric since:

- Predictive maintenance failures are **rare**, and accuracy can be misleading when the majority class dominates performance.
- ROC-AUC evaluates **ranking ability**, not just discrete predictions.
- It's useful for risk-based decision environments such as manufacturing.

Example interpretation:

- **0.50** = random guessing
- **0.70–0.85** = reasonable discrimination

- **>0.90** = strong predictive performance
- **>0.97** = near-ideal diagnostic discrimination

4.5 Baseline Models and Fair Comparison Strategy

Two baselines were evaluated alongside XGBoost:

Model	Baseline Type	Motivation
Random forest	Simple and robust tree based	Widely used baseline for industrial data
Gradient Boosting	Strong sequential learning model	Compares sensitivity to complex interactions
XGBoost	Advanced boosting model	State of the art performance for tabular data

All models were trained with the same train/test split and evaluated with the same metrics.

4.6 Hyperparameter Strategy

Models were tuned using documented best-practice values and grid search experimentation during development.

Model	Example Key Parameters
Random Forest	n_estimators, max_depth, class_weight
Gradient Boosting	learning_rate, min_samples_leaf, max_depth
XGBoost	subsample, colsample_bytree, scale_pos_weight

Hyperparameter choices were not exhaustively optimized in order to emphasize realistic comparison rather than leaderboard performance.

4.7 Reproducibility and Randomness Control

To ensure replicability:

- All experiments use a fixed random seed (random_state=42)
- Results were verified by rerunning multiple trials
- Code is available in the public repository

4.8 Avoiding Data Leakage

Leakage can produce deceptively high performance. Several protections were implemented:

- Dropped failure-cause columns from clean dataset
- Paired product-ID splits ensured no sample is seen in both training and testing
- Align-columns method ensured no hidden metadata differences
- Cross-dataset evaluation only used engineered features, never direct duplication

Summary

This evaluation pipeline is designed to represent real deployment challenges: noisy data, limited labels, performance trade-offs, and model transfer across environments. It prioritizes realistic assessment over optimized benchmarks, demonstrating how machine learning models behave in conditions closer to industrial reality.

5. Results and Discussion

This section presents the experimental results that we got from training three machine learning models, Random Forest, Gradient Boosting, and XGBoost, on both the clean and irregular datasets individually, as well as under cross-dataset evaluation conditions. Results are discussed considering predictive performance, generalization, robustness to noise, and real-world implications for predictive maintenance in industrial environments.

5.1 Individual Dataset Performance

Models were first evaluated independently on each dataset to determine baseline performance before transfer testing. ROC-AUC was used as the primary metric due to class imbalance and the real-world importance of ranking failures over non-failures.

Performance Comparison by Dataset

Dataset	Model	ROC-AUC	Notes
Clean (AI4I 2020)	**Random Forest**	**0.967**	**High stability under clean signals**
	Gradient Boosting	**0.981**	**Best performance, strong handling for structure**
	XGBoost	**0.976**	**Comparable top tier performance**
Irregular (AI4I-PMDI)	**Random Forest**	**0.945**	**More difficulty adapting to noise**

	Gradient Boosting	**0.937**	**Higher sensitivity to irregularity**
	XGBoost	**0.946**	**Strong generalization under variability**

The clean dataset results indicate nearly perfect classification performance, mostly due to the sensor signals being consistent, and feature distributions stable. The irregular dataset performance is slightly lower due to the presence of:

- Missing values and mean imputation effects
- Control-dependent variable structure
- System parameter representing different equipment operating conditions
- Increased variability in temperature, torque, and wear readings

These declines reflect realistic operational challenges where sensor readings drift or degrade over time.

5.2 Confusion Matrix Interpretation

Across both datasets, confusion matrices showed extremely low false positives and reasonably low false negatives. For example, here is the confusion matrix on the clean dataset:

	Predicted Normal	Predicted Failure
Actual Normal	1936	3
Actual Failure	12	49

Key interpretation:

- **49 out of 61 failures were detected**, meaning approximately **12 failures were missed**
- Very high precision for predicting failure (few false alarms)
- Reasonable recall, reflecting some cost of undetected failures

In real predictive maintenance, the relative cost of false negatives (unexpected machine failure) is normally much more than false positives (unnecessary maintenance). Therefore, performance interpretation depends on business risk tolerance.

5.3 Feature Importance Interpretation

Feature importance analysis revealed differences in model focus and behavior:

- **Random Forest** distributes importance relatively evenly, consistent with its bagging structure.
- **Gradient Boosting** strongly prioritizes key variables influencing failure, such as temperature differential and combined wear-and-torque.
- **XGBoost** shows sharp emphasis on temperature differential, especially under noisy data, suggesting sensitivity to physical thermal instability.

Example insight:

Boosting models appear more sensitive to engineered features tied to physical degradation, while Random Forest is stronger across all features but less discriminative.

5.4 Cross-Dataset Generalization

The core research question covered whether models trained on synthetic (clean) data generalized effectively to real-world noisy conditions and vice-versa.

Experiment	ROC-AUC	Meaning
Train clean -> test unclean	0.9337	Good generalization from clean ideal data to noisy environment
Train unclean -> test clean	0.9648	Models trained on noisy data adapt extremely well to ideal clean conditions

Interpretation

1. **Training on clean data still performs well on unclean**, meaning feature-level relationships are preserved even under noise.
2. **Training on unclean performs even better on clean**, indicating that exposure to variability may build more robust models.
3. The results initially appeared too high due to **data leakage** caused by testing samples that had been seen during training; switching to paired Product_ID splitting corrected this issue.

Takeaway

Synthetic data alone is insufficient for final deployment. We need models to observe noisy, real-world-like variations to generalize well.

5.5 What Worked and Why

Aspect	Success Reason

Feature engineering	Captured the relevant mechanical relationships
Tree-based models	Handle nonlinear interactions effectively
Paired product ID splitting	Eliminated leakage and produced realistic results
Boosting models	Strong discrimination where feature associations matter

5.6 What Did Not Work and Why

Issue	Explanation
Early cross dataset low AUC (around 0.55)	Feature naming was mismatched which was preventing alignment
Unrealistically high AUC before the aligned splitting	Leakage due to training and testing happening on the same underlying rows
Lower performance on irregular dataset	The noise masked the underlying patterns

5.7 Limitations and Future Directions

Limitation	Possible improvement
Synthetic data does not fully represent operational complexity	Use real datasets through NDAs with industry partners
Binary classification only	Expand to failure type prediction or time to failure metrics
No temporal component	Add sequence-based models such as LSTM/Transformer using mean time to failure

A compelling real-world insight is that **model transfer between equipment generations may be non-trivial**, similar to cross-dataset testing in this project.

Summary

The results demonstrate that predictive maintenance models perform very well in clean environments and maintain strong performance when trained and tested across noisy datasets, assuming leakage is controlled. Adding realistic irregularities improves generalization. This proves that synthetic data is useful for prototyping but cannot replace real-world data for final deployment.

6. Conclusion

Predictive maintenance has become an important capability across modern manufacturing industries, particularly in high-precision environments such as semiconductor fabrication, aerospace production, and industrial automation. Machine failures can result in significant financial cost, equipment damage, and production downtime, and predictive maintenance models aim to forecast failures before they occur using sensor-based monitoring, machine learning, and

statistical analysis. This project explored how machine learning models trained on synthetic clean data behave when exposed to more realistic noisy datasets, and whether synthetic data alone is sufficient for developing reliable predictive maintenance systems.

For our project, two different datasets were used. First, the AI4I 2020 predictive maintenance dataset which is a clean and well-structured synthetic dataset. Second, we used the AI4I-PMDI, which is an irregular dataset designed to simulate real-world sensor degradation, drift, missing values, along with operational variability. These datasets created a meaningful experimental setting that reflects the real challenge companies face when attempting to build predictive systems without access to confidential industrial data.

Models trained separately on each dataset demonstrated strong performance. All three tree-based models, Random Forest, Gradient Boosting, and XGBoost, achieved strong ROC-AUC scores when trained and tested on the same dataset. Performance dropped slightly on the AI4I-PMDI dataset due to increased noise, missing values, and the addition of system and control parameters that introduced realistic variability. This decline makes sense and aligns with real-world system behavior where sensor noise reduces signal interpretability.

The most important part of the project was the cross-dataset evaluation which was designed to simulate deploying a model trained on synthetic data into an actual industrial environment. Cross-testing revealed that models trained on the clean dataset generalized surprisingly well to the irregular dataset, achieving ROC-AUC of approximately 0.93. Even more notably, models trained on the irregular dataset transferred exceptionally well to the clean dataset, achieving ROC-AUC values above 0.96. These results tell us that exposure to noise, variation, and imperfect measurements may produce more robust models capable of handling a broader range of operating conditions.

This work also revealed challenges and risks. Early cross-testing produced unusually high-performance scores due to unintentional data leakage as product samples were used in both training and testing when selecting random splits. This was resolved by splitting datasets based on shared product IDs to ensure that no row was seen during both training and testing. This correction reduced scores to realistic values and taught us a valuable lesson. Valid evaluation design is as important as the model architecture itself. Without proper split methodology and data alignment, results may be misleading and create false confidence in model performance.

Based on the results, several conclusions can be drawn. First, synthetic data is certainly a valuable tool for early prototyping and algorithm development, especially in fields where real data access is limited. Second, synthetic data cannot wholly replace real world data because it cannot fully capture industrial complexity. Models must experience realistic noise conditions in order to generalize effectively. Third, cross-dataset evaluation should be standard practice for predictive maintenance research as it reflects deployment reality. Model training often occurs under controlled conditions, but deployment occurs in noisy environments. Finally, this project shows the importance of detailed feature engineering and understanding domain context, temperature differences, torque behavior, and wear dynamics proved central to failure prediction. This aligns with known physical failure mechanisms in rotary machinery.

Looking forward, this study suggests multiple directions for improvement, including the incorporation of time-series models such as LSTM or Transformers, expanding classification to predict specific failure types rather than binary outcomes, and partnering with industrial organizations to obtain real operational datasets. Incorporating additional domain knowledge, such as degradation curves, environmental effects, or maintenance logs, could further improve prediction accuracy and interpretability as well.

Ultimately, this project reinforces that predictive maintenance is not simply a modeling problem, but a systems-level challenge requiring careful evaluation of design, realistic data representation, and alignment between engineering and analytical objectives. The results demonstrate that machine learning models can effectively identify failure conditions, but they must be exposed to real-world complexity to ensure reliable performance in deployment settings.

References

Bischl, B., Casalicchio, G., Hothorn, T., Lang, M., Lindauer, M., Richter, J., & Bossek, J. (2020). *AI4I 2020 Predictive Maintenance Dataset* [Data set]. UCI Machine Learning Repository. https://archive.ics.uci.edu/ml/datasets/AI4I+2020+Predictive+Maintenance+Dataset

Autran, J.-V., Kuhn, V., Diguet, J.-P., Dubois, M., & Buche, C. (2024). *AI4I-PMDI: Predictive maintenance datasets with complex industrial settings' irregularities.* *Procedia Computer Science, 234*, 546–553. https://doi.org/10.1016/j.procs.2024.09.546

GeeksforGeeks. (n.d.). *Introduction to Transfer Learning*. https://www.geeksforgeeks.org/machine-learning/ml-introduction-to-transfer-learning/

GeeksforGeeks. (2025, May 30). *Understanding the confusion matrix in Machine Learning*. https://www.geeksforgeeks.org/machine-learning/confusion-matrix-machine-learning/

GeeksforGeeks. (2025, July 23). *Difference between random forest and XGBoost*. https://www.geeksforgeeks.org/machine-learning/difference-between-random-forest-vs-xgboost/