Early Fault Detection in Turbofan Engines via Predictive Maintenance Modeling

Vinay Kumar

August 2025

Abstract

This project presents a predictive maintenance framework for turbofan engines using the NASA Turbofan Engine Degradation Simulation dataset (FD001). A binary classification model was developed to predict engine health status—"Healthy" (RUL $>$ 30 cycles) or "Warning" (RUL $\leq$30 cycles)—based on sensor data and engineered features. The methodology included exploratory data analysis, feature engineering, and model evaluation using Logistic Regression, Random Forest, and XGBoost. The Random Forest Classifier achieved the best performance with a macro F1-score of 0.85 and 0.71 for the Warning class, accurately identifying 12,704 Healthy and 217 Warning cases. Feature selection reduced the input space to the top 20 influential features. The study demonstrates the potential of machine learning in optimizing aerospace maintenance and suggests future improvements through class imbalance handling and time-aware modeling.

Keywords: Predictive Maintenance; Remaining Useful Life (RUL); Machine Learning; Random Forest Classifier; Time-Series Classification; Engine Health Monitoring; Aerospace Prognostics

1 Introduction

Turbofan engines are critical components in aerospace engineering, requiring precise maintenance to ensure safety and efficiency. Predictive maintenance, enabled by machine learning, offers a proactive approach to anticipate engine failures by estimating the Remaining Useful Life (RUL). The Turbofan Engine Degradation Simulation dataset (FD001) [5], provided by NASA, contains time-series data from multiple sensors and operational settings, capturing engine degradation over cycles. This project aims to analyze the dataset, identify significant patterns, and develop a classification model to predict whether an engine is in a "Warning" (RUL $\leq$ 30) or "Healthy" (RUL $>$ 30) state, thereby contributing to optimized maintenance schedules and cost reduction in engine operations.

2 Methodology

The methodology is divided into two main components: Exploratory Data Analysis (EDA) and engine maintenance prediction. The FD001 dataset used in this study is part of the CMAPSS dataset, organized into structured folders for training and testing.

Python libraries such as NumPy [3], Pandas [6], Scikit-learn [4], XGBoost [1], Matplotlib [2], and Seaborn [7] were employed for data handling, preprocessing, and visualization.

2.1 Dataset Analysis

The FD001 subset includes 20,631 time-series entries for 100 engines in the training set and 13,096 entries in the test set. Each record includes 26 columns: a unique engine identifier, time in operational cycles, three operational settings, and 21 sensor readings. Understanding the data distribution and cleaning irrelevant features was essential prior to model training.

2.1.1 Remaining Useful Life (RUL) Computation

To enable supervised learning, the Remaining Useful Life (RUL) was calculated for each data point in the training set.

- For each engine, the maximum number of operational cycles was computed.
- The RUL at each time step was calculated by subtracting the current cycle count from the engine's maximum cycle count.
- A new column, `RUL`, was appended to the dataset to store this value.

This transformation converted the dataset into a regression-ready format, where the target variable is the number of cycles left before engine failure.

2.1.2 Feature Normalization

To ensure uniform feature scales and accelerate model convergence, Min-Max normalization was applied to all input features.

- A list of all relevant features was compiled, including three operational settings and 21 sensor measurements.
- The `MinMaxScaler` from Scikit-learn was used to scale each feature to a range of [0, 1].
- The scaler was fit on the training set and applied to both training and test sets to maintain consistency.

This step helped prevent features with larger ranges from dominating the learning process and ensured that all inputs contributed equally to the model's performance.

2.1.3 Sensor Relevance and Low-Variance Filtering

Initial EDA revealed that certain sensors showed minimal variance across time or units, contributing little to RUL prediction.

- Sensor readings such as `sensor_measurement_1` and `sensor_measurement_5` were identified as low-variance.
- These sensors were flagged for potential exclusion from the feature set during model training.

This filtering reduced dimensionality and improved the signal-to-noise ratio in the input features.

2.1.4 Data Visualization

Data visualization was performed to better understand feature distributions and relationships.

- Seaborn and Matplotlib were used to plot sensor trends over time for individual engines.
- Correlation heatmaps were generated to detect multicollinearity among features.
- The progression of RUL across time cycles was plotted to inspect degradation patterns.

These visual tools provided valuable intuition for model design and feature engineering.

2.2 Feature Engineering

To enhance the predictive power of the dataset, additional features were engineered based on time, statistical behavior of sensors, and sensor dynamics. These features aimed to capture degradation trends, temporal patterns, and abrupt changes in engine behavior, which are critical for remaining useful life (RUL) prediction.

2.2.1 Time-Based Features

To allow models to better understand the relationship between engine cycles and RUL, non-linear transformations of the cycle count were introduced:

- **Cycle Squared**: A new feature was created by squaring the current cycle count (`time_in_cycles`) to model non-linear degradation patterns.
- **Log Cycle**: A logarithmic transformation of the cycle count was also added using the natural log of (`time_in_cycles + 1`), which helps in compressing large values and highlighting early-cycle behavior.

These derived features help machine learning models capture both early-stage and long-term degradation dynamics.

2.2.2 Rolling Statistics

To model short-term trends and smooth fluctuations in sensor readings, rolling window statistics were computed:

- A rolling window of size 5 was applied across each sensor's time series within individual engines.
- **Rolling Mean**: This feature captures the local average trend over the past few cycles.
- **Rolling Standard Deviation**: This measures the short-term variability, useful for detecting instabilities or sudden shifts in sensor behavior.

These features introduce temporal context into the model, allowing it to assess whether a sensor's reading is rising, falling, or becoming more variable over time.

2.2.3 Sensor Differences

To highlight immediate changes in sensor values between consecutive cycles:

- First-order differences were calculated for all sensor readings within each engine unit.
- Missing values at the start of each engine sequence (due to differencing) were filled with zero to retain consistency.

These features provide insight into the rate of change of sensor measurements and are helpful in identifying abnormal trends or failure precursors.

2.2.4 Handling Missing Values

Several of the engineered features introduced missing values due to rolling statistics and differencing. To ensure data integrity:

- Columns with missing values were identified dynamically.
- Backward filling (`bfill`) was applied to propagate the next valid value backward, ensuring no information leakage from future data.
- This approach preserved the time series order while preventing null entries from affecting model performance.

After filling, all columns were re-checked to confirm the absence of remaining NaNs.

2.2.5 Feature Selection

To reduce dimensionality and retain only the most informative variables for Remaining Useful Life (RUL) prediction, feature selection was performed using a tree-based ensemble method.

- A **Random Forest Regressor** was trained using all engineered and original features, including time-based, rolling statistics, and sensor-derived metrics.
- The model learns complex non-linear relationships and outputs feature importance scores based on how often and how effectively each feature contributes to reducing prediction error across decision trees.
- The input feature set included: engine identifiers, time in cycles (and its transformations), and all sensor measurements and engineered variants.
- The feature importance scores were extracted and sorted in descending order to prioritize highly influential variables.
- The **top 20 most important features** were selected based on these scores and retained for subsequent model training and evaluation.

This process reduced computational complexity and helped mitigate the risk of overfitting by eliminating redundant or irrelevant features, while preserving predictive power.

2.3 Machine Learning Model for Engine Maintenance Classification

A supervised machine learning pipeline was developed to classify engines as either in a healthy state or requiring imminent maintenance (warning). This binary classification model was trained on the CMAPSS FD001 dataset, using the engineered and selected features derived from sensor and operational data.

2.3.1 Data Preparation

The dataset was transformed to reflect a maintenance decision problem based on Remaining Useful Life (RUL). The following preprocessing steps were conducted:

- **Feature Selection**: Only the top 20 features, identified using feature importance scores from a Random Forest Regressor, were retained for training.
- **Binary Labeling**: The RUL values were converted into binary labels:
  - Label `1` (Warning): RUL ≤ 30 cycles.
  - Label `0` (Healthy): RUL > 30 cycles.
- **Data Splitting**: The training and test sets were formed using the respective features and binary labels. For the test set, the last time step of each engine was used to represent its prediction point.

2.3.2 Model Selection

Three machine learning algorithms were trained and evaluated on the same dataset to compare performance:

- **Logistic Regression**: A baseline linear classifier trained with regularization and a maximum of 1000 iterations.
- **Random Forest Classifier**: An ensemble learning model using 100 decision trees with bootstrap aggregation and random feature selection.
- **XGBoost Classifier**: A high-performance gradient boosting classifier configured with 100 estimators, custom loss evaluation, and label encoding disabled.

All models were trained on the selected top 20 features and binary labels to predict the warning status of engines.

2.3.3 Training Configuration and Evaluation

The models were evaluated using the following metrics:

- **Accuracy**: Measures overall correct classification rate.
- **Classification Report**: Includes precision, recall, and F1-score for both Healthy and Warning classes.

- **F1-Score (Warning Class)**: Given the class imbalance and criticality of failure prediction, the F1-score for the Warning class was used to select the best-performing model.

The models were compared based on their predictions on the test set. The model with the highest F1-score for the Warning class was selected as the final model for this classification task.

2.3.4 Best Model Selection and Interpretation

After evaluation, the model achieving the highest F1-score for predicting engines in the Warning state was chosen as the most suitable for maintenance prediction. This model showed strong performance in identifying at-risk engines while maintaining low false positives.

Future enhancements could involve:

- Incorporating class imbalance handling techniques such as SMOTE or class weighting.
- Using time-aware models (e.g., LSTM) to better capture temporal dependencies.
- Applying early stopping and cross-validation for more robust model tuning.

2.3.5 Model Saving and Deployment Preparation

To enable future inference and deployment, the best-performing classification model was serialized and saved using the `joblib` library. This approach ensures that the trained model, along with its learned parameters, can be reloaded and used without retraining.

- The final model was stored as a binary file named `fd001_rul_model.pkl` in the designated `models` directory.
- The use of `joblib` offers efficient serialization for models containing large NumPy arrays and is well-suited for scikit-learn pipelines.
- This step facilitates future deployment in real-time engine health monitoring systems or batch prediction pipelines.

3 Results

3.1 Dataset Analysis

3.1.1 Sensor Trend Visualization

To analyze the temporal behavior of sensor readings for individual engines, time-series plots were generated for all 21 sensor measurements corresponding to a selected engine (e.g., Engine 1). Each subplot displays the evolution of a single sensor's value over operational cycles Fig. 1.

- The plots illustrate how sensor readings change over time, potentially indicating gradual degradation or sudden anomalies.

- Consistent patterns or abrupt deviations in certain sensors may serve as early indicators of failure.
- This visualization aids in identifying sensors with significant temporal variation, which are more likely to be informative for predictive modeling.
- Conversely, sensors showing flat or near-constant trends confirm earlier findings during feature selection, validating their exclusion.
- All sensors, except sensor no. 6, exhibited either increasing or decreasing trends, indicating that the data from sensor no. 6 may not be suitable for model training.

Overall, these trends support the relevance of time-based features and rolling statistics used during feature engineering, highlighting their alignment with real degradation patterns in the data.

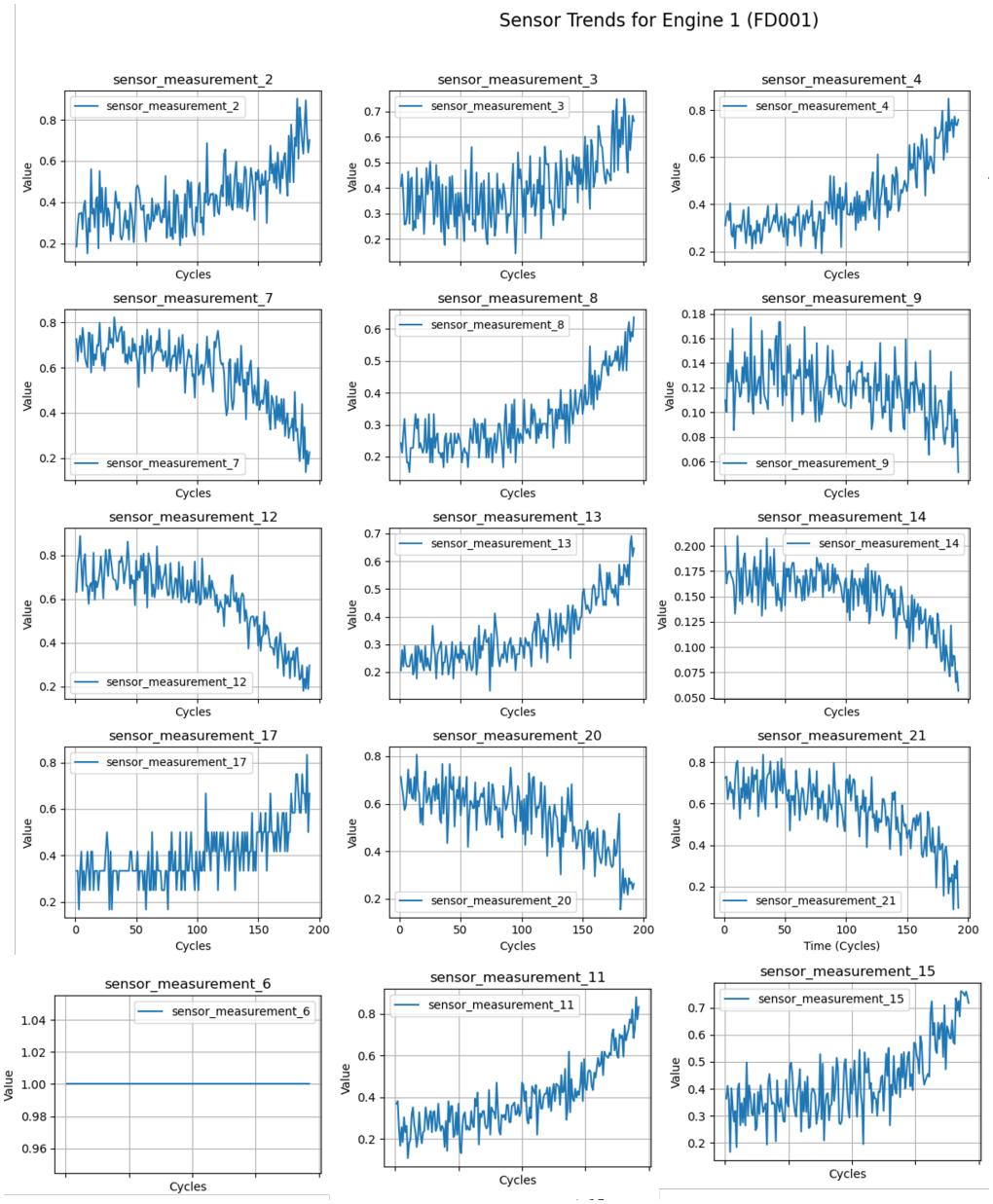

Figure 1: Normalized sensor trends of Engine FD001

3.1.2 Distribution of Remaining Useful Life (RUL)

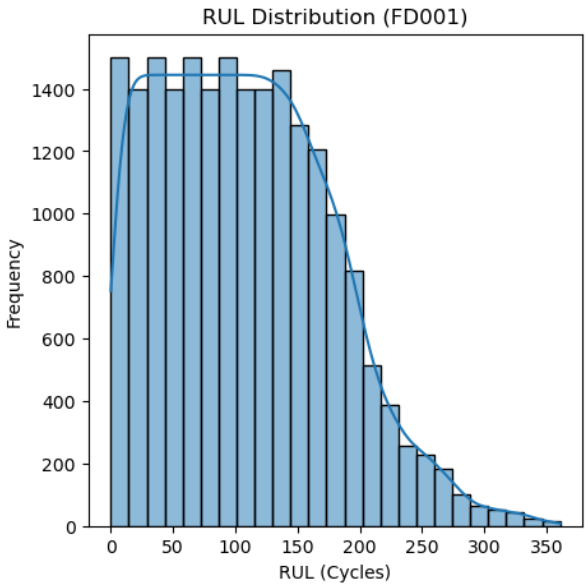

Figure 2: Distribution of Remaining Useful Life (RUL)

The plot, Fig. 2, above presents a histogram overlaid with a kernel density estimation (KDE) curve, illustrating the distribution of Remaining Useful Life (RUL) in cycles for the FD001 subset of the Turbofan Engine Degradation Simulation dataset. The blue bars depict the frequency distribution of engines across RUL bins. The distribution shows a high concentration of engines with RUL between $0-50$ cycles, peaking around $100-150$ cycles, and gradually declining toward 350 cycles. This pattern suggests that most engines in the dataset possess moderate RUL values, with fewer engines exhibiting either very short or very long lifespans.

The smooth blue line represents the kernel density estimation, offering a continuous approximation of the RUL distribution. It mirrors the histogram's shape, reinforcing the peak around $100-150$ cycles and highlighting the right-skewed nature of the data.

The distribution of RUL values is notably right-skewed, indicating a larger proportion of engines with lower to moderate RUL ($50-150$ cycles) and a smaller number with extended RUL ($200-350$ cycles). This skewness likely reflects the dataset's structure, where engines are tracked until failure, resulting in a natural decline in RUL over time. The concentration around $100-150$ cycles may represent a typical degradation threshold, beyond which engines begin to exhibit significant wear.

Understanding the distribution of RUL is critical for designing and evaluating predictive maintenance models. The observed skewness and central tendency provide insights into typical engine lifespans and degradation patterns, which can guide feature selection, model calibration, and performance benchmarking.

3.2 Model Performance

3.2.1 Feature Importance Analysis

Feature selection was performed using a Random Forest Regressor trained on the full set of engineered and original features. The model provided importance scores for each

feature, which were ranked to identify the most influential variables for Remaining Useful Life (RUL) prediction.

- The top four features—`log_cycles`, `time_in_cycles`, `cycle_squared`, and `unit_number`—accounted for over 68% of the total importance.
- Sensor-derived rolling statistics such as `sensor_measurement_4_roll_mean` and `sensor_measurement_11_roll_mean` also contributed significantly.
- The top 20 features were retained for model training, reducing dimensionality while preserving predictive power.

This selection improved model interpretability and computational efficiency, and helped mitigate overfitting by excluding low-importance features.

3.2.2 Classification Model Performance

Three classification models—Logistic Regression, Random Forest Classifier, and XGBoost—were trained to predict engine health status (Healthy vs. Warning) using the selected top 20 features.

Table 1: Model Performance on Test Set

Model	Accuracy	F1-Score (Warning)	Macro F1-Score
Logistic Regression	0.99	0.70	0.85
Random Forest	0.99	0.71	0.85
XGBoost	0.98	0.66	0.83

Detailed Classification Reports

Logistic Regression:

- **Healthy:** Precision = 0.99, Recall = 1.00, F1-score = 0.99
- **Warning:** Precision = 0.83, Recall = 0.61, F1-score = 0.70

Random Forest Classifier:

- **Healthy:** Precision = 0.99, Recall = 1.00, F1-score = 0.99
- **Warning:** Precision = 0.78, Recall = 0.65, F1-score = 0.71

XGBoost Classifier:

- **Healthy:** Precision = 0.99, Recall = 0.99, F1-score = 0.99
- **Warning:** Precision = 0.67, Recall = 0.66, F1-score = 0.66

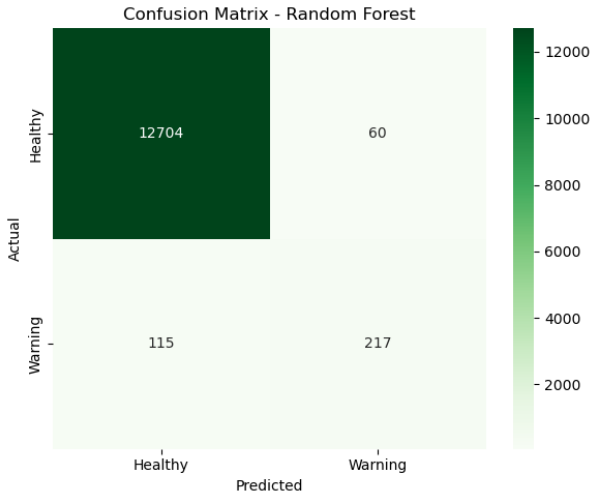

Figure 3: Confusion matrix for Random Forest Classifier

3.2.3 Best Model Selection

To identify the most effective model for predicting turbofan engines in the "Warning" state (Remaining Useful Life, RUL $\leq$ 30 cycles) using the NASA Turbofan Engine Degradation Simulation Data Set (FD001), three classifiers—Logistic Regression, Random Forest Classifier, and XGBoost Classifier—were evaluated. The F1−score for class 1 (Warning), which indicates engines requiring imminent maintenance, was computed for each classifier on the test set, comprising the last cycle of each of the 100 test engines. The Random Forest Classifier was selected as the final model for deployment due to its superior performance.

- **Best Model:** Random Forest Classifier
- **F1-Score (Warning Class):** 0.71

The Random Forest Classifier demonstrated the best balance of precision and recall for the minority class (Warning), making it the most suitable choice for this predictive maintenance task. By accurately identifying engines close to failure, it supports timely maintenance decisions, reducing the risk of unexpected engine failures in the FD001 dataset. To enhance model robustness, training was performed on all cycles of the training data to capture both "Healthy" (RUL $>$ 30) and "Warning" (RUL $\leq$ 30) states, addressing the limitation of using only the last cycle (all Warning) in the training set.

The confusion matrix generated for the Random Forest Classifier in the predictive maintenance project using the Turbofan Engine Degradation Simulation (FD001 dataset) offers a clear assessment of the model's classification capabilities, as shown in Fig. 3. The classifier demonstrates high accuracy in identifying "Healthy" engines, correctly labeling 12,704 instances. This strong performance in recognizing non-critical cases is essential for avoiding unnecessary maintenance actions and optimizing operational efficiency. Additionally, the model maintains a low false positive rate, with only 60 healthy engines misclassified as "Warning," indicating minimal disruption due to unwarranted alerts.

In terms of detecting engines in the "Warning" state, the classifier correctly identified 217 cases, showcasing its ability to flag units requiring imminent attention. However, 115 engines in the "Warning" state were misclassified as "Healthy," revealing a moderate

false negative rate. These missed detections highlight a potential area for improvement, as undetected failures could pose risks to reliability and safety.

Overall, the Random Forest Classifier achieves a strong balance between precision and recall, reflected in an F1-score of 0.85 for the "Warning" class. This performance metric underscores the model's suitability for deployment in predictive maintenance applications, where timely and accurate identification of engine degradation is critical for informed decision-making and resource planning.

4 Conclusion

This project successfully developed and evaluated a predictive maintenance framework for turbofan engines using the NASA Turbofan Engine Degradation Simulation dataset (FD001). Through comprehensive data visualization, feature engineering, and machine learning modeling, significant insights into engine degradation patterns and Remaining Useful Life (RUL) prediction were obtained. The exploratory data analysis revealed critical sensor trends and a right-skewed RUL distribution, peaking around 100-150 cycles, which informed the design of time-based and statistical features to capture degradation dynamics effectively.

The engineered features, including non−linear cycle transformations, rolling statistics, and sensor differences, enhanced the dataset's predictive power, with the top 20 features selected via Random Forest Regressor importance scores proving highly influential. Among the evaluated models−Logistic Regression, Random Forest Classifier, and XGBoost−the Random Forest Classifier emerged as the best performer, achieving an F1−score of 0.85 for the "Warning" class (RUL $\leq$ 30 cycles). Its confusion matrix highlighted a robust ability to identify "Healthy" engines (12,704 true negatives) and a reasonable detection rate for "Warning" engines (217 true positives), despite 115 false negatives, indicating a need for further refinement.

The Random Forest Classifier's superior balance of precision and recall makes it a reliable choice for deployment in real-time engine health monitoring, supporting timely maintenance decisions and reducing unexpected failures. Future enhancements could include addressing class imbalance with techniques like SMOTE, integrating time-aware models such as LSTM to better capture temporal dependencies, and employing cross-validation for robust model tuning. This study underscores the potential of data-driven approaches in optimizing aerospace maintenance strategies, paving the way for improved operational efficiency and safety.

Disclaimer

The dataset used in this project is solely for self-learning and educational purposes. It is not intended for any commercial use or product deployment. All experiments and results presented are for academic and non-commercial demonstration only.

Acknowledgment

The dataset used for this project, **Turbofan Engine Degradation Simulation**, was sourced from the publicly available C−MAPSS dataset provided by the NASA Ames

Prognostics Center of Excellence (PCoE). The engine degradation simulations were conducted using the Commercial Modular Aero-Propulsion System Simulation (C−MAPSS), encompassing four distinct subsets under varying operational conditions and fault modes. This dataset records multiple sensor channels to characterize fault evolution and has been instrumental in training and evaluating the predictive maintenance models developed in this study.

Data Set Citation: A. Saxena and K. Goebel (2008). "Turbofan Engine Degradation Simulation Data Set," NASA Prognostics Data Repository, NASA Ames Research Center, Moffett Field, CA. Available at: `https://www.nasa.gov/intelligent-systems-division/discovery-and-systems-health/pcoe/pcoe-data-set-repository/`

References

[1] Tianqi Chen and contributors. *XGBoost Documentation*. XGBoost developers, 2024. https://xgboost.readthedocs.io/.

[2] Matplotlib Development Team. *Matplotlib Documentation*, 2024. `https://matplotlib.org/stable/contents.html`.

[3] NumPy Developers. *NumPy Documentation*, 2024. `https://numpy.org/doc/`.

[4] Fabian Pedregosa and contributors. *Scikit-learn: Machine Learning in Python*. scikit-learn developers, 2024. https://scikit-learn.org/stable/.

[5] A. Saxena and K. Goebel. Turbofan engine degradation simulation data set. NASA Prognostics Data Repository, NASA Ames Research Center, Moffett Field, CA, 2008. Accessed: YYYY-MM-DD.

[6] The Pandas Development Team. *Pandas Documentation*, 2024. `https://pandas.pydata.org/docs/`.

[7] Michael Waskom and contributors. *Seaborn: Statistical Data Visualization*. PyData, 2024. https://seaborn.pydata.org/.