# Enhancing Aircraft Engine RUL Prediction: Interpretable Models and Bayesian Optimization

Juan Echeagaray *, Jonathan Montalvo-Urquizo † and María Guadalupe Villarreal-Marroquín ‡
School of Engineering and Sciences
Tecnologico de Monterrey
Monterrey, Nuevo León, México
Email: *pabloechg@outlook.com, †jmontalvo@tec.mx, ‡maria.villarreal@tec.mx

***Abstract*—This project aims to develop an advanced predictive maintenance (PdM) framework for turbofan engines, a crucial component of aircraft operation. Predicting Remaining Useful Life (RUL) is essential for maintaining safety and efficiency. The approach involves converting run-to-failure data into more efficient formats, segmenting flights, and extracting features using statistical descriptors. A variety of efficient machine learning models are employed, with the selected model predicting RUL, while Bayesian Optimization is applied to optimize this primary model and the two secondary models that provide the prediction intervals. Interpretability is ensured using Shapley values.**

**Efficiency is demonstrated through a proof of concept using the NCMAPSS dataset, showing that accurate RUL estimates can be achieved on a personal computer with a standard GPU. Performance is expected to improve when implemented on dedicated hardware. This framework is anticipated to significantly reduce operational costs by enabling informed maintenance scheduling, thereby enhancing safety and reliability.**

***Index Terms*—Predictive Maintenance, Time Series Segmentation, XGBoost, SHAP, Prediction Intervals, Bayesian Optimization.**

## I. Introduction

Modern industries heavily rely on condition-based monitoring systems to collect and analyze vast amounts of data from machinery, encompassing sensor readings, vibrations, temperatures, acoustic data, and more [1], [2]. These data serve as the backbone for assessing the Remaining Useful Life (RUL) of critical equipment. Accurate RUL predictions hold immense significance, as they offer the potential to revolutionize operational efficiency by enabling preventive maintenance precisely when required. This approach not only reduces maintenance costs [3] but also enhances the sustainability of organizations [4] and provides a competitive edge over industry counterparts. Moreover, in safety-critical scenarios, such as aviation (where turbofan engines play a pivotal role), precise RUL predictions are crucial to avoid catastrophic mid-flight engine failures and production line stoppages.

From a computational perspective, three primary modeling approaches have emerged: physics-driven models, data-driven models, and hybrid models that combine aspects of both. Among these, data-driven models have gained prominence due to their relatively straightforward implementation and remarkable accuracy. This category, particularly, has been dominated by the application of deep learning techniques, which are adept at capturing intricate relationships within the data collected from Condition-Based Monitoring (CBM) systems [5], [6], [7], [8], [9]. However, the computational demands associated with training and deploying deep learning models often pose challenges, making them unsuitable for certain scenarios. In such cases, lower-footprint models like gradient boosting algorithms and linear models may offer a more practical alternative, albeit with slightly reduced performance. These models also lend themselves well to hyperparameter tuning, which can be difficult to achieve with most deep learning approaches.

While a significant portion of research endeavors is focused on enhancing model performance, an equally crucial but often neglected aspect is model interpretability. In the current landscape of machine learning and AI, model interpretability and accountability hold paramount importance. These qualities are not just regulatory requirements; they are markers of responsible AI development. Laws such as the General Data Protection Regulation (GDPR) [10] and the Algorithmic Accountability Act [11] have reinforced the necessity for algorithms deployed to the general public to be transparent and accountable.

Beyond compliance, these laws underscore the fundamental principle that a trustworthy AI model should not only make accurate predictions but also provide insights into why it arrives at those conclusions. This ensures that AI systems are transparent, understandable, and ultimately serve as valuable tools in an ever-evolving technological ecosystem. As a new trend gains momentum, placing model interpretability on the same level as model accuracy, it reinforces the idea that model transparency and accountability are essential for ethical and reliable AI systems. The main question is not so much if we can get an explainable AI (XAI) solution, but rather if it we can get an XAI with an accuracy comparable to that of regular AI/ML? [12]

In the broader field of machine learning, Shapley values have been extensively used to improve model understanding. For example, [13] demonstrated the utility of Shapley values in understanding the contributions of auxiliary inputs such as sex and age for accurate detection of acute myocardial infarction. Furthermore, the work by [14] emphasized how Shapley values can help accurate ML models become transparent in critical scenarios such as Peer-to-Peer lending. In the context of predictive maintenance (PdM), it becomes imperative to establish means for comprehending and explaining model predictions, making the insights from these studies highly relevant to this

domain.

In line with model interpretation, the issue of prediction uncertainty is crucial. Studies across various domains have highlighted the importance of incorporating prediction intervals (PI) in machine learning models [15]. For instance, in stock forecasting, PI application builds trust in risk analysis models [16]. Similarly, in wind power production, PI aids in understanding future energy yield for reliability and cost-effectiveness [17]. Moreover, in predicting Remaining Useful Life (RUL) of engines, PI helps schedule maintenance accurately [9].

Integrating prediction intervals provides stakeholders valuable insights into prediction ranges or confidence levels. This additional information facilitates informed decision-making, balancing maintenance needs and operational efficiency.

The paper is organized as follows: Section II formally introduces the problem to be addressed, while Section III outlines the research objectives and the corresponding evaluation metrics used to assess performance. In Section IV, the quantitative techniques to be used are introduced. Finally, Sections V and VI present the main findings and conclusions, as well as directions for future research.

## II. Problem Statement

The foundation for this work is the New Commercial Modular Aero-Propulsion System Simulation (NCMAPSS) dataset proposed in [18], encompassing sensor readings, environmental descriptors, auxiliary variables, virtual sensor readings, and RUL values; the NCMAPSS dataset represents an upgrade from the previous CMAPSS dataset by the inclusion of real world flight conditions into the simulation regime. Figure 1 aids in conceptualizing the core components of a turbofan engine, as represented in the NCMAPSS dataset [18].

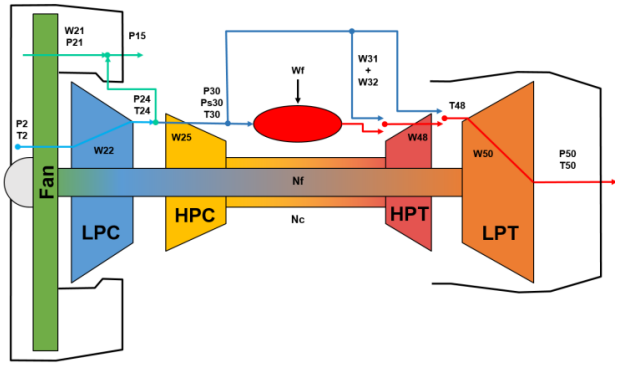


Fig. 1: NCMAPSS Turbofan engine diagram, taken from [18].

The NCMAPSS dataset, distinguished by its high quality, served as the chosen data source for the PHMAP 2021 Data Challenge [19]. It was prioritized over the CMAPSS dataset due to its incorporation of a broader range of flying conditions and multiple failure modes, providing a more realistic simulation of actual flights with heightened fidelity. While sharing the fundamental objective with PHMAP of estimating the remaining useful life (RUL) for a fleet of turbofan engines under challenging conditions, the current research extends this aim by applying more stringent constraints:

- **Efficient Modeling:** Given computational limitations, the chosen model for training must prioritize efficiency. This constraint rules out most deep learning approaches, which may be computationally intensive.
- **Model Interpretability:** In light of the criticality of the situation, model interpretability is paramount. It is imperative to understand why a model produces a specific RUL estimate, ensuring transparency and trustworthiness of the predictions.
- **Confidence and Prediction Intervals:** Additionally, there is a need to develop confidence in the model's predictions. This entails providing prediction intervals for each RUL estimate, allowing for a better assessment of the prediction's reliability.

## III. Objectives

The primary objectives of this study are to develop:

1) PdM framework to predict RUL for a designated fleet of machinery.
2) Models which ensure reproducibility, stability, robustness and confidence.
3) Tools to interpret and visualize the model's predictions.

The previous objectives are to be accomplished subject to the following constraints and assumptions:

- RUL prediction of an uniform fleet of machines.
- Availability of a labeled dataset with run to failure sequences of each machine.

### A. Evaluation Metrics

This paper follows the metric definitions for (1) in [18] to develop comparable results with prior research. Given two vectors $y, \hat{y} \in \mathbb{R}^m$ representing real RUL labels and RUL estimates (unscaled), the PHMAP loss function for the predictive model is defined as:

$$\begin{aligned} \text{RMSE}(y,\hat{y}) &= \sqrt{\frac{1}{m}\sum_{i=1}^{m}(y_i-\hat{y}_i)^2} \\ \textit{NASA}(y,\hat{y}) &= \frac{1}{m}\sum_{i=1}^{m}\left[\exp(\alpha\cdot(y_i-\hat{y}_i))-1\right] \\ \alpha &= \begin{cases} \frac{-1}{10} & \text{if } y_i-\hat{y}_i \le 0 \\ \frac{1}{13} & \text{if } y_i-\hat{y}_i > 0 \end{cases} \\ \mathcal{L}(y,\hat{y}) &= \frac{1}{2}\left(\text{RMSE}(y,\hat{y}) + \textit{NASA}(y,\hat{y})\right). \end{aligned} \tag{1}$$

NASA's scoring function is popularly used in aeronautics since it is an asymmetric loss function with higher penalties for overestimates [20].

In addition to RUL point estimates, this research also aims to develop a model for estimating prediction intervals for any

RUL prediction; the loss function to optimize being the Mean Pinball Loss subject to a specified quantile $\tau$, defined as

$$\begin{aligned} \beta &= \begin{cases} \tau & \text{if } y_i - \hat{y}_i \geq 0 \\ \tau - 1 & \text{if } y_i - \hat{y}_i < 0 \end{cases} \\ \mathcal{L}_\tau(y, \hat{y}) &= \frac{1}{m} \sum_{i=1}^{m} \beta(y_i - \hat{y}_i). \end{aligned} \tag{2}$$

## IV. Quantitative Techniques

In this section, the quantitative techniques employed in the study are outlined. For a visual overview of the process, the diagram in Figure 2 provides a comprehensive overview of the research methodology, depicting the step-by-step process flow employed in this study.

### A. Data Resampling and Downcasting

In dealing with large datasets that exceed the capacity of a personal computer's memory, practical solutions are essential for efficient iterative research. One such approach involves resampling the data, typically recorded over several hours, to a coarser granularity, such as every $n$ seconds (see Figure 3). This step significantly reduces the computational load while preserving the core information needed for analysis.

Additionally, downcasting data types from 64-bit floating points to 32-bit floating points proves valuable in optimizing computational resources. Although 64-bit precision is often more than necessary for many analysis tasks, it comes at a cost in terms of computational complexity and memory usage. Downcasting to 32-bit floating points minimizes these issues without substantially compromising the integrity of the analysis. In most cases, the loss of precision is negligible.

Furthermore, Table I provides an overview of the dataset sizes used in this work, highlighting the impact of these strategies on data manageability.

TABLE I: Size of each dataset in the PHMAP 2021 Competition (in GB).

| Dataset | Size (GB) |
|---|---|
| DS01-005.h5 | 2.9 |
| DS02-006.h5 | 2.5 |
| DS03-012.h | 3.7 |
| DS04.h5 | 3.8 |
| DS05.h5 | 2.6 |
| DS06.h5 | 2.5 |
| DS07.h5 | 2.7 |
| DS08a-009.h5 | 3.2 |
| DS08c-008.h5 | 2.4 |
| DS08d-010.h5 | 2.9 |
| DS_Validation_f.h5 | 2.9 |
| Total | 32.1 |

Implementing these strategies (data resampling and datatype downcasting) not only eases data handling in resource-constrained settings but also ensures that critical information remains intact, facilitating streamlined and efficient iterative studies.

### B. Time Series Segmentation

Machine learning models often require fixed-shaped individual samples, making segmentation critical. It is hypothesized that valuable time series information resides within its segments. By emphasizing segmentation and feature extraction at this level, we aim to create a richer dataset for training machine learning models.

*1) Feature Scaling:* Given the distinct scales of the variables present in the dataset, a Min-Max scaling routine (3) is applied on each flight individually. For any given flight encoded in a matrix $X \in \mathcal{M}_d^T(\mathbb{R})$, of $T$ rows (i.e., flight length) and $d$ columns (i.e., number of sensors):

$$\begin{aligned} X_{min} &= \mathbb{1}_{(d\times 1)} \min_i \{m_{ij} : 1 \leq i \leq T, \forall j \in \{1, \dots d\}\} \\ X_{max} &= \mathbb{1}_{(d\times 1)} \max_i \{m_{ij} : 1 \leq i \leq T, \forall j \in \{1, \dots d\}\} \\ X_{std} &= \frac{(X - X_{min})}{X_{max} - X_{min}}. \end{aligned} \tag{3}$$

*2) Binary Segmentation:* In dealing with extensive flight data processing, the use of approximate methods for time series segmentation proves to be essential. This section explores these efficient techniques, chosen to manage the sheer volume of flight records. These methods have shown promising results, making them a practical choice for large-scale time series analysis.

For a given signal $Y = \{y_t\}_{t=1}^{t=T}$ of $T$ samples, where $y_t \in \mathbb{R}^d$, we assume there exists a set $\mathcal{T} = \{t_1^*, t_2^*, \dots, t_n^*\}$ coding the $n-1$ stages of $Y$. As an example, see the time series depicted in Figure 4, the real set $\mathcal{T}$ for this series would be $\{158, 332, 500\}$.

Change point detection consists in searching for the optimal segmentation $\mathcal{T}$ of a time series $y$ [21]. An easy approach to establish a loss function for the algorithm is to simply define such loss as (4) where $C$ is a measure of the goodness of fit for the selected signal:

$$V(\mathcal{T}, y) = \sum_{k=0}^{K} C(y_{t_k, t_{k+1}}). \tag{4}$$

A popular choice for $C$ is the L2 loss, given a subset $y_{\mathcal{I}} = \{y_t\}_{t\in\mathcal{I}}$ and $\bar{y}$ being the component wise mean of $y_{\mathcal{I}}$, the L2 loss is defined as:

$$C(y_{\mathcal{I}}) = \sum_d \sum_{t\in\mathcal{I}} ||y_t - \bar{y}||_2^2. \tag{5}$$

Binary Segmentation is an approximate method to estimate the real set $\mathcal{T}$, it's a sequential greedy algorithm [21], [22] that estimates the first change point as described in:

$$\hat{t}_1 := \underset{1\leq t<T-1}{\arg\min}\ C(y_{0\dots t}) + C(y_{t\dots T}). \tag{6}$$

Binary Segmentation may also be constrained to a subset of possible change points by fixing some external hyperparameters, such as the minimal size for a segment and the indices at which segment splits are tested. The present study empirically chose a minimum size of $\lfloor 0.12 \cdot T \rfloor$ and a restriction on the indices to be of the form $\lfloor 0.05 \cdot T \rfloor$. These parameters may be subject to the hyperparameter optimization regime proposed in Section IV-F, but its left out due to time and resource constraints.

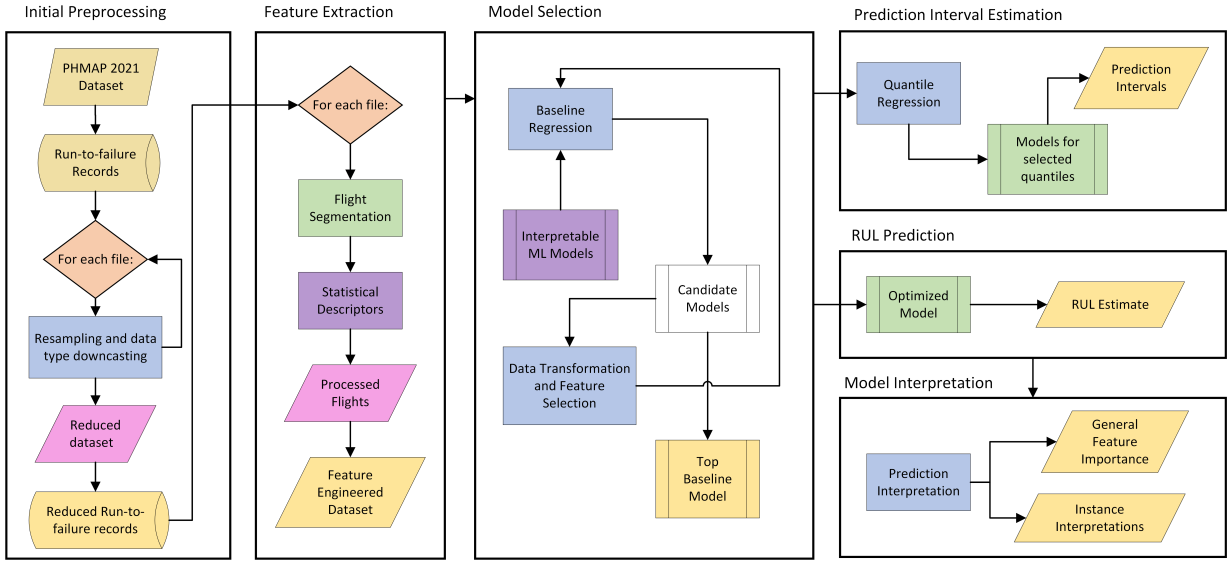


Fig. 2: Proposed methodology overview.

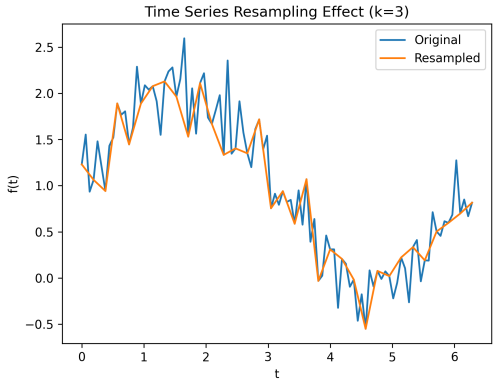


Fig. 3: Example of time series resampling (keep every $k$th record).

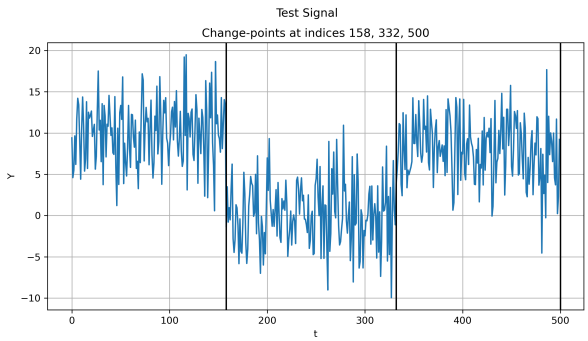


Fig. 4: Example of change points in univariate time series.

*3) Feature Engineering:* The uniform segments obtained from the previous segmentation step serve as the foundation for feature engineering. In this section, a set of statistical descriptors is applied to each of these segments, generating a feature vector that encapsulates all relevant information from a flight.

Given a segment $S_i$ from a sample $F$, which encompasses $T_i$ timestamps, $S_i \in \mathcal{M}_d^{T_i}(\mathbb{R})$ with $T_i \geq \alpha$ where $\alpha$ is a constraint on the minimum size of $S_i$. A set of descriptors $\mathcal{F}_s = \{f \mid f : \mathcal{M}_d^{T_i}(\mathbb{R}) \to \mathbb{R}^d\}$ is applied on each segment $S_i$ individually, this research employs the descriptors in Table II:

TABLE II: Proposed set of statistical descriptors.

| Statistical Descriptors | |
|---|---|
| Minimum | Std. Deviation |
| 25th Percentile | Variance |
| Median | Kurtosis |
| 75th percentile | Skew |
| Maximum | Coefficient of Variation |
| Mean | - |

Applying each statistical descriptor on all the segments $S_i$, results in a feature vector $\hat{F} \in \mathbb{R}^{d \cdot |\mathcal{F}_s| \cdot |S|}$. Given the frequent calls to these statistical descriptor functions, prioritizing efficient implementations is essential to streamline computational resources for timely analysis.

In cases where the resulting vector exhibits high dimensionality, it may be beneficial to apply a feature selection algorithm; a simple approach would be to select features based upon a variance threshold under the assumption that nearly constant features offer little value to a machine learning model [23]. The choice of threshold is arbitrary and should be part of any hyperparameter optimization regime.

### C. Model Selection

In the model selection phase, we start with a processed dataset and create an 80-20 train-test split for robust evaluation. We then choose efficient machine learning models (see Table III for the proposed list) with characteristics such as linear regression coefficients or decision tree splits. These models are fitted and evaluated on both training and test

subsets [1], calculating PHMAP loss and RMSE for each and recording total training times.

TABLE III: Proposed set of ML models.

| Interpretable Models | |
|---|---|
| Linear Regression [24] | Ridge Regression [24] |
| Lasso [24] | LassoLars [24] |
| ElasticNet [24] | Random Forest [24] |
| XGBoost [25] | LightGBM [26] |
| Catboost [27] | SVM [24] |
| Decision Tree [24] | Dummy Regression (Mean) [24] |

If necessary, apply any data transformation techniques like standardization, scaling, or PCA to improve model performance. This iterative process of model fitting, evaluation, and possible transformation continues until a satisfactory collective performance is achieved.

After completing the iterations, models are strategically selected based on training scores while taking into account their respective training times. This approach ensures that the chosen algorithms are not only effective but also computationally feasible for subsequent hyperparameter optimization.

### D. Model Interpretability

Shapley values offer a model agnostic approach to post-hoc model interpretation [28], where one explains a single prediction given the original input features. Adapting the original Shapley value formula, which traditionally assesses each player's contribution to game outcomes, allows us to quantify the significance of individual feature values in predicting model outcomes. In this context, the initial coalition $S$ of players turns into the coalition of feature values for any instance in which a model generates predictions. Considering a predictive model $\hat{f}$ trained on a set of $p$ features, the Shapley value for any feature $j$ is determined as the weighted average of all marginal contributions that the feature can make, encapsulating its unique impact on the overall predictive outcome:

$$\phi_j = \sum_{S \subseteq \{1,\dots p\}-\{j\}} \frac{|S|!(p-|S|-1)!}{p!} \times (val(S \cup \{j\}) - val(S)). \quad (7)$$

To estimate the contribution of any given coalition of feature values we marginalize the model predictions over all the features not present in the coalition $S$:

$$val(S) = \int \hat{f}(x_1, \dots, x_p) d\mathbb{P}_{x \notin S}. \quad (8)$$

Shapley values also possess vital properties for effective model interpretability:

- **Efficiency:** feature contributions express how a single prediction deviates from the mean prediction of $\hat{f}$

$$\sum_{j=1}^{p} \phi_j = \hat{f}(x) - \mathbb{E}[\hat{f}(X)]. \quad (9)$$

[1]Mean Squared Error is chosen as a common loss function to optimize

- **Symmetry:** two features have the same Shapley value if and only if their contributions are the same

$$val(S \cup \{j\}) = val(S \cup \{k\}) \quad \forall S \subseteq \{1, \dots, p\} - \{j, k\} \Rightarrow \phi_j = \phi_k.$$

- **Null Feature:** a feature that has no effect on a prediction has a zero Shapley value

$$val(S \cup \{j\}) = val(S) \quad \forall S \subseteq \{1, \dots, p\} \Rightarrow \phi_j = 0.$$

- **Linearity:** the contribution of an ensemble of models is the same as the sum of the contribution of each model in the ensemble

$$\phi_j^{f+g} = \phi_j^f + \phi_j^g.$$

Efficient approximations exist, catering to specific model types like linear, tree-based, and deep learning models. These approximations balance accuracy with computational feasibility [28], [29], [30], making Shapley values practical for understanding various machine learning models [13], [14], [31].

### E. Prediction Intervals

In predictive modeling, all predictions inherently carry uncertainty due to various factors, such as data noise and model limitations.

Prediction intervals offer a practical way to quantify this uncertainty. Instead of providing a single point estimate, they define a range within which future values are likely to fall based on observed variables. When paired with a point estimate, prediction intervals indicate the level of uncertainty associated with that prediction, often expressed as a confidence level (e.g., 90%, 95%, 99%) [32].

Quantile regression is a powerful technique used to create prediction intervals [16], [17], [9]. For the base case, it involves developing two models, one for the lower and one for the upper bound of the interval. By specifying the desired confidence level, quantile regression captures the spread of potential outcomes effectively [33], [15].

Quantile regression relies on the Pinball Loss (2) to model the conditional quantiles of the target variable given the covariates. It can be shown that minimizing the Pinball Loss for a given quantile is equivalent to formulating a Conditional Quantile function for a target variable $y$ given some covariates $X$ [34]:

$$\mathcal{Q}_\tau(Y|X) = \arg\min_{q(X)} \mathbb{E}[\mathcal{L}_\tau(Y, q(X))]. \quad (10)$$

### F. Hyperparameter Optimization

Hyperparameter optimization methods offer a systematic approach to find improved hyperparameter configurations, avoiding the assumption that default settings are ideal or relying on human intuition.

Efficient computation is a critical concern when optimizing hyperparameters due to the substantial cost associated with evaluating machine learning models. It is imperative to use algorithms that leverage information from previous evaluations

to iteratively suggest better configurations. In contrast to Grid Search and Random Search, which explore the hyperparameter space without learning, Bayesian Optimization algorithms consider each past trial to propose more promising configurations.

Tree-structured Parzen Estimators (TPE) is a practical implementation of Bayesian Optimization [35]. Instead of modeling the distribution of the objective function given hyperparameters $p(y|x)$, it models the distribution of hyperparameters given the objective function. TPE defines:

$$p(x|y) = \begin{cases} g(x) & y < y^* \\ b(x) & y \geq y^* \end{cases} \tag{11}$$

The probability density function $g(x)$ is constructed as a mixture of Gaussian distributions fitted to hyperparameter configurations $x^{(i)}$ tied so far, where the associated loss function $f(x)$ is below a predefined threshold $y^*$. Any points not meeting this criterion are covered by the distribution $b(x)$. TPE dynamically selects a new threshold, which is a predetermined quantile $\gamma$ of the observed losses at each iteration within the optimization process.

TPE suggests the next hyperparameter configuration by sampling a point from $g(x)$ that maximizes the ratio in (12). This ratio serves as an approximation for the Expected Improvement function.

$$\begin{aligned} S_g &= \{x : x \sim g(x)\} \\ x_s &= \arg\max_{x \in S_g} \frac{g(x)}{b(x)} \end{aligned} \tag{12}$$

## V. Results

### A. Computational Resources

The present research was conducted in a WSL2 virtual machine with the specifications described in Table IV:

TABLE IV: Computational Resources used.

| Component | Detail |
|---|---|
| OS | `Ubuntu 20.04.6 LTS on Windows 10 x86_64` |
| CPU | `12th Gen Intel i5-12450H (12) @ 2.496GHz` |
| GPU | `NVIDIA GeForce RTX 3060` |
| RAM | `9949MiB` |

The code necessary to replicate and use the developed tools can be accessed via this link.

### B. Experimental Results

By resampling every 5 observations and converting 64-bit floats to 32-bit floats as described in Section IV-A, the PHMAP dataset experienced a substantial 96% reduction in its total size (see Table V). The resulting processed data files are subsequently saved as parquet files, preserving the specified data types.

Following the application of Binary Segmentation as proposed in Section IV-B, with a minimum size requirement of 12% of the total flight length and the identification of 3 change points, an analysis of the dataset was conducted. The statistical descriptors, as outlined in Table II, were computed. Figure 5 illustrates the Coefficient of Variation for a randomly selected

TABLE V: Size of each dataset before and after resampling and downcasting.

| Dataset | Original (GB) | Processed Size (MB) |
|---|---|---|
| DS01-005 | 2.9 | 113 |
| DS02-006 | 2.5 | 97 |
| DS03-012 | 3.7 | 142 |
| DS04 | 3.8 | 146 |
| DS05 | 2.6 | 104 |
| DS06 | 2.5 | 102 |
| DS07 | 2.7 | 108 |
| DS08a-009 | 3.2 | 126 |
| DS08c-008 | 2.5 | 96 |
| Total | 26.4 GB | 1034 MB |

aircraft within the dataset over its initial operational life. The diagram shows two main horizontal axis encoding the flight number the plane took and the sequential number of segment being analyzed; given how early the health state of the sampled plane declined (at its 26th cycle) one can hypothesize that the initial spikes in the variation of fuel flow may be an indicator of an action taken by the pilots or a weather condition that damaged the plane's engine.

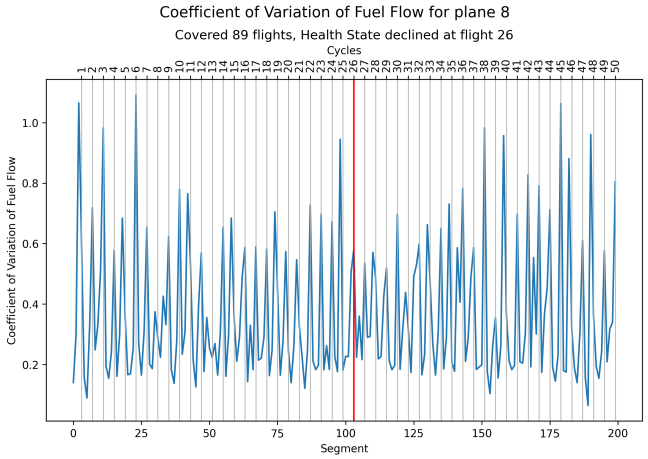


Fig. 5: Coefficient of Variation for a random plane throughout its operational life.

With the new feature-engineered dataset, the ML models presented in this study underwent training and testing. The dataset was partitioned into an 80-20 split for this purpose, with default hyperparameters used for each model. The models were ranked based on their PHMAP loss on the test set, as indicated in Table VI. It is noteworthy that models employing the gradient boosting method outperformed other models.

The recommendation is to use XGBoost, despite not having the lowest loss or the shortest training time among gradient boosting algorithms. This choice is motivated by the maturity of the XGBoost library, its GPU acceleration capabilities (not used for this test), and its widespread use in both research and industry [36], [37], [38], [39]. All these qualities streamline the training and optimization process.

The TPE algorithm was executed for a total of 500 iterations, serving three distinct models. These models encompass the estimation of the Remaining Useful Life (RUL) of an

TABLE VI: Train and Test metrics for the selected suite of ML models.

| Model | Train Loss | Test Loss | Train RMSE (cycles) | Test RMSE (cycles) | Training Time (s) |
|---|---|---|---|---|---|
| Catboost | 2.043 | 5.321 | 3.769 | 9.399 | 25.692 |
| LightGBM | 2.636 | 5.366 | 4.823 | 9.478 | 2.475 |
| **XGBoost** | **0.697** | **5.726** | **1.304** | **10.048** | **6.989** |
| Random Forest | 2.086 | 5.898 | 3.837 | 10.338 | 223.095 |
| Ridge | 6.340 | 6.775 | 11.033 | 11.746 | 0.043 |
| Elastic Net | 6.888 | 7.023 | 11.872 | 12.126 | 0.029 |
| Lasso | 6.933 | 7.053 | 11.937 | 12.170 | 0.0285 |
| Lasso Lars | 6.933 | 7.053 | 11.937 | 12.170 | 0.043 |
| SVM | 6.878 | 7.168 | 11.893 | 12.372 | 12.919 |
| Decision Tree | 0.0 | 9.472139 | 0.0 | 14.871 | 3.393 |
| Dummy (mean) | 16.779 | 17.324 | 23.688 | 24.351 | 0.0008 |
| Linear Regression | 5.142 | 87743.231 | 9.112 | 14.664 | 0.246 |

engine, as well as two additional models responsible for establishing the lower and upper bounds for the prediction interval associated with the RUL estimation.

Each trial involved a 5-fold cross-validation split, and the result was determined as the average loss over these validation splits. The proposed search space, as well as the best configurations discovered for each model, are detailed in Table VII.

TABLE VII: Hyperparameter Search Space for TPE and results for RUL estimators and, Lower and Upper Bounds.

| Parameter | Range | RUL | Lower | Upper |
|---|---|---|---|---|
| Variance Threshold | $U(0.1, 1)$ | 0.102 | 0.1 | 0.944 |
| Learning Rate | $\exp(U(-4.5, 0))$ | 0.021 | 0.116 | 0.042 |
| Boosting Rounds | $U_{\mathbb{Z}}(100, 10000)$ | 1963 | 5859 | 3830 |
| Max Depth | $U_{\mathbb{Z}}(2, 30)$ | 30 | 15 | 20 |
| Min Child Weight | $U_{\mathbb{Z}}(2, 50)$ | 29 | 41 | 38 |
| Subsample | $U(0.01, 1)$ | 0.726 | 0.854 | 0.795 |
| Gamma | $U(0, 100)$ | 62.667 | 0.124 | 0.109 |
| Alpha | $U(0, 100)$ | 8.721 | 18.859 | 11.387 |
| Lambda | $U(0, 100)$ | 34.832 | 20.921 | 82.706 |
| Loss | - | 5.687 | 1.853 | 2.399 |

In Figure 6, the predictions generated by the previous models over the entire operational lifespan of an aircraft are depicted.

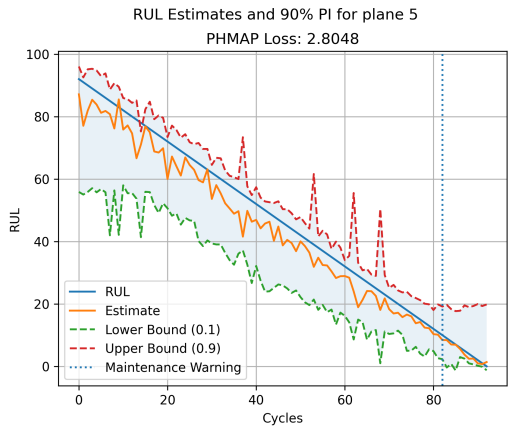


Fig. 6: RUL estimates and PI for the operational life of a plane.

Notably, the model tends to adopt a conservative approach in its predictions, aligning with the intended effect of the PHMAP loss function, where a preference for underestimates is established for safety considerations. Additionally, the provided Prediction Interval (PI) offers stakeholders a relative measure of prediction uncertainty at each point. It is noteworthy that the PI tends to flatten towards the end of the plane's operational life, indicating that the conforming models have learned the general behavior of planes during this final stage. These observations collectively emphasize the reliability and safety-conscious nature of the RUL estimation model.

To further enhance safety measures, the research suggests issuing a maintenance warning whenever the model predicts a Remaining Useful Life (RUL) value that falls below the Root Mean Square Error (RMSE) computed for the entire dataset.

To offer a deeper insight into the predictions generated by the Remaining Useful Life (RUL) estimator, Shapley values were computed for each prediction of the dataset. Figure 7 presents the distribution of Shapley values for the top 12 features utilized by the model, ranked by the mean absolute Shapley value across the predictions.

The color scale, transitioning from blue to red, signifies the relative values of features in terms of high or low, considering their distribution. On the x-axis, Shapley values are associated with each feature numerical value. For instance, the Shapley values affirm that as the flight number increases, the model predicts a lower Remaining Useful Life (RUL), as indicated by the high negative values for high flight numbers.

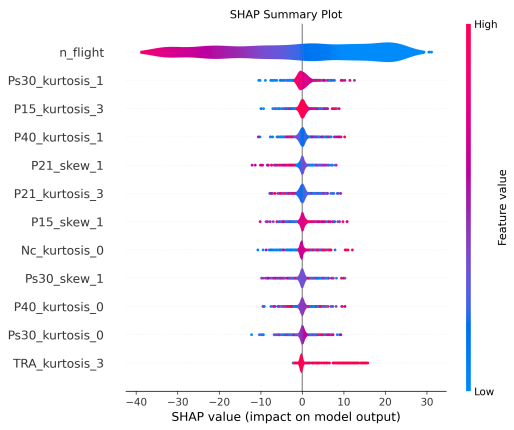


Fig. 7: Summary of Shapley values for the dataset.

For more detailed analysis, individual predictions can be

examined, as shown in Figure 8. This figure depicts how each feature value, accompanying each feature label on the left vertical axis, deviates the model predictions from their expected value (shown in the lower right corner) to the point where they add up to the actual prediction, labeled as $f(x)$, for this example in specific, the most important feature was the flight number, its Shapley value shows how being at its 57th flight reduces its operational cycles by almost 16 from the average prediction for all the planes in the dataset.

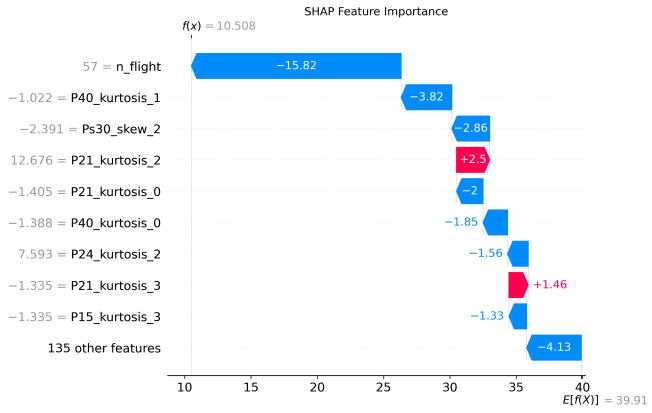


Fig. 8: Example of Shapley values for a single prediction.

## VI. Conclusion and Future Work

The research has yielded a comprehensive toolbox of techniques and utilities developed to optimize models systematically, resulting in improved performance beyond their basic counterparts. Although the models in this study provided predictions on a par with those achieved through Deep Learning methods, there was a slight reduction in overall performance [40]. The trade-off is best evaluated against the notable advantages offered by the models, including faster and more efficient training and inference processes and enhanced interpretability through the provided tools.

Looking forward, potential future work encompasses the incorporation of the segmentation algorithm into the hyperparameter optimization process. This entails exploring various segmentation methods and determining the optimal number of segments, which may lead to further performance enhancements. This will increase the complexity of the process and will need a more robust error handling system.

Another avenue for exploration involves modifying the feature selection technique by replacing the variance-based approach with the coefficient of variation. This adjustment could lead to more robust feature selection, particularly for features within a narrower range of values. Standardizing features by their mean can improve the model's capabilities.

Additionally, further experimentation with the hyperparameter optimization process is advisable. Simple adjustments, such as increasing the number of trials conducted by the Tree-structured Parzen Estimators (TPE) and broadening the search space intervals, may reveal superior configurations that were previously overlooked.

In pursuit of continuous improvement, ongoing research endeavors aim to enhance the effectiveness and efficiency of predictive maintenance frameworks, making a significant contribution to the evolving landscape of predictive maintenance methodologies.

## References


[1] M. S. Azari, F. Flammini, S. Santini, and M. Caporuscio, "A systematic literature review on transfer learning for predictive maintenance in industry 4.0," *IEEE Access*, vol. 11, pp. 12 887–12 910, 2023.

[2] M. Achouch, M. Dimitrova, K. Ziane, S. S. Karganroudi, R. Dhouib, H. Ibrahim, and M. Adda, "On predictive maintenance in industry 4.0: Overview, models, and challenges," *Applied Sciences*, vol. 12, p. 8081, 8 2022.

[3] H. Brink, A. Krych, O. R. Cardenas, and S. Tiwari, "Establishing the right analytics-based maintenance strategy," 7 2021. [Online]. Available: https://www.mckinsey.com/capabilities/operations/our-insights/establishing-the-right-analytics-based-maintenance-strategy

[4] K. Karuppiah, B. Sankaranarayanan, and S. M. Ali, "On sustainable predictive maintenance: Exploration of key barriers using an integrated approach," *Sustainable Production and Consumption*, vol. 27, pp. 1537–1553, 2021.

[5] W. Zhang, D. Yang, and H. Wang, "Data-driven methods for predictive maintenance of industrial equipment: A survey," *IEEE Systems Journal*, vol. 13, no. 3, pp. 2213–2227, 2019.

[6] A. Lövberg, "Remaining useful life prediction of aircraft engines with variable length input sequences," *Annual Conference of the PHM Society*, vol. 13, 12 2021.

[7] N. DeVol, C. Saldana, and K. Fu, "Inception based deep convolutional neural network for remaining useful life estimation of turbofan engines," *Annual Conference of the PHM Society*, vol. 13, 12 2021.

[8] D. Solís-Martín, J. Galán-Páez, and J. Borrego-Díaz, "A stacked deep convolutional neural network to predict the remaining useful life of a turbofan engine," *Annual Conference of the PHM Society*, vol. 13, 11 2021.

[9] M. Zhang, D. Wang, N. Amaitik, and Y. Xu, "A distributional perspective on remaining useful life prediction with deep learning and quantile regression," *IEEE Open Journal of Instrumentation and Measurement*, vol. 1, pp. 1–13, 2022.

[10] General data protection regulation (gdpr) - official legal text. [Online]. Available: https://gdpr-info.eu/

[11] GovInfo. S. 3572 (is) - algorithmic accountability act of 2022. [Online]. Available: https://www.govinfo.gov/app/details/BILLS-117s3572is

[12] P. P. Angelov, E. A. Soares, R. Jiang, N. I. Arnold, and P. M. Atkinson, "Explainable artificial intelligence: an analytical review," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 11, no. 5, p. e1424, 2021.

[13] L. Ibrahim, M. Mesinovic, K.-W. Yang, and M. A. Eid, "Explainable prediction of acute myocardial infarction using machine learning and shapley values," *IEEE Access*, vol. 8, pp. 210 410–210 417, 2020.

[14] M. J. Ariza-Garzón, J. Arroyo, A. Caparrini, and M.-J. Segovia-Vargas, "Explainability of a machine learning granting scoring model in peer-to-peer lending," *IEEE Access*, vol. 8, pp. 64 873–64 890, 2020.

[15] A. Brando, C. Torres-Latorre, J. A. Rodríguez-Serrano, and J. Vitrià, "Building uncertainty models on top of black-box predictive apis," *IEEE Access*, vol. 8, pp. 121 344–121 356, 2020.

[16] G. Alfonso, A. D. Carnerero, D. R. Ramirez, and T. Alamo, "Stock forecasting using local data," *IEEE Access*, vol. 9, pp. 9334–9344, 2021.

[17] Y. Zhou, Y. Sun, S. Wang, R. J. Mahfoud, H. H. Alhelou, N. Hatziargyriou, and P. Siano, "Performance improvement of very short-term prediction intervals for regional wind power based on composite conditional nonlinear quantile regression," *Journal of Modern Power Systems and Clean Energy*, vol. 10, no. 1, pp. 60–70, 2022.

[18] M. A. Chao, C. Kulkarni, K. Goebel, and O. Fink, "Aircraft engine run-to-failure dataset under real flight conditions for prognostics and diagnostics," *Data*, vol. 6, p. 5, 1 2021.

[19] ——, "Phm society data challenge 2021," 2021.

[20] A. Saxena, K. Goebel, D. Simon, and N. Eklund, "Damage propagation modeling for aircraft engine run-to-failure simulation," in *2008 international conference on prognostics and health management*. IEEE, 2008, pp. 1–9.

[21] C. Truong, L. Oudre, and N. Vayatis, "Selective review of offline change point detection methods," *Signal Processing*, vol. 167, p. 107299, 2020.

[22] F. Jiang, Z. Zhao, and X. Shao, "Time series analysis of covid-19 infection curve: A change-point perspective," *Journal of econometrics*, vol. 232, no. 1, pp. 1–17, 2023.

[23] J. Li, K. Cheng, S. Wang, F. Morstatter, R. P. Trevino, J. Tang, and H. Liu, "Feature selection: A data perspective," *ACM computing surveys (CSUR)*, vol. 50, no. 6, pp. 1–45, 2017.

[24] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[25] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16. New York, NY, USA: ACM, 2016, pp. 785–794. [Online]. Available: http://doi.acm.org/10.1145/2939672.2939785

[26] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "Lightgbm: A highly efficient gradient boosting decision tree," *Advances in neural information processing systems*, vol. 30, pp. 3146–3154, 2017.

[27] A. V. Dorogush, V. Ershov, and A. Gulin, "Catboost: gradient boosting with categorical features support," *CoRR*, vol. abs/1810.11363, 2018. [Online]. Available: http://arxiv.org/abs/1810.11363

[28] H. Chen, I. C. Covert, S. M. Lundberg, and S.-I. Lee, "Algorithms to estimate shapley value feature attributions," *Nature Machine Intelligence*, pp. 1–12, 2023.

[29] C. Molnar, *Interpretable machine learning*. Lulu. com, 2020.

[30] D. Fryer, I. Strümke, and H. Nguyen, "Shapley values for feature selection: The good, the bad, and the axioms," *IEEE Access*, vol. 9, pp. 144 352–144 360, 2021.

[31] M. Wang, K. Zheng, Y. Yang, and X. Wang, "An explainable machine learning framework for intrusion detection systems," *IEEE Access*, vol. 8, pp. 73 127–73 141, 2020.

[32] E. N. Barron and J. G. Del Greco, *Probability and Statistics for STEM: A Course in One Semester*. Springer Nature, 2022.

[33] Y. Romano, E. Patterson, and E. Candes, "Conformalized quantile regression," *Advances in neural information processing systems*, vol. 32, 2019.

[34] R. Koenker and K. F. Hallock, "Quantile regression," *Journal of economic perspectives*, vol. 15, no. 4, pp. 143–156, 2001.

[35] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Algorithms for hyper-parameter optimization," *Advances in neural information processing systems*, vol. 24, 2011.

[36] R. Cai, S. Xie, B. Wang, R. Yang, D. Xu, and Y. He, "Wind speed forecasting based on extreme gradient boosting," *IEEE Access*, vol. 8, pp. 175 063–175 069, 2020.

[37] L. Magadán, J. Roldán-Gómez, J. C. Granda, and F. J. Suárez, "Early fault classification in rotating machinery with limited data using tabpfn," *IEEE Sensors Journal*, pp. 1–1, 2023.

[38] R. Shwartz-Ziv and A. Armon, "Tabular data: Deep learning is not all you need," *Information Fusion*, vol. 81, pp. 84–90, 2022.

[39] L. Grinsztajn, E. Oyallon, and G. Varoquaux, "Why do tree-based models still outperform deep learning on typical tabular data?" *Advances in Neural Information Processing Systems*, vol. 35, pp. 507–520, 2022.

[40] "2021 PHM Conference Data Challenge Results - PHM Society Data Repository," 10 2021. [Online]. Available: https://data.phmsociety.org/2021-phm-conference-data-challenge-winners/