Optimizing Aircraft Engine RUL Prediction with Bayesian-Enhanced Interpretable ML and SHAP

Saturday 28th October, 2023

Juan Pablo Echeagaray González
School of Engineering and Sciences
Instituto Tecnológico y de Estudios Superiores de Monterrey
Monterrey, Nuevo León, México
pabloechg@outlook.com

***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 but also enhances the sustainability of organizations 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 indispensable to avoid catastrophic mid-flight engine failures and production line stoppages.

In the realm of RUL prediction, 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 [3], [4], [5], [6], [7]. 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 broader field of machine learning, Shapley Values have been extensively used to improve model understanding. For example, [8] demonstrated the utility of Shapley Values in understanding the contributions of auxiliary inputs such as sex and age for accurately detection of acute myocardial infarction. Furthermore, the work by [9] emphasized how Shapley Values can help accurate ML models become transparent in critical scenarios such as Peer-to-Peer lending. In the context of machinery subject to predictive maintenance, 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 [10]. For instance, in stock forecasting, PI application builds trust in risk analysis models [11]. Similarly, in wind power production, PI aids in understanding future energy yield for reliability and cost-effectiveness [12]. Moreover, in predicting Remaining Useful Life (RUL) of engines, PI helps schedule maintenance accurately [7].

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 proposed methodology is introduced. Finally, Sections V and VI present the main findings and conclusions, as well as directions for future research. Additionally, Section VII provides details on the resources utilized in the research and directs readers to a repository containing the code necessary for replicating the presented results.

II. Problem Statement

In alignment with the PHMAP 2021 Data Challenge, the main goal is to estimate the remaining useful life (RUL) of a fleet of turbofan engines under challenging conditions, characterized by high variability and multiple failure modes. It's important to note that this task adheres to specific 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.

The foundation for this challenge is the New Commercial Modular Aero-Propulsion System Simulation (NCMAPSS) dataset proposed in [13], encompassing sensor readings, environmental descriptors, auxiliary variables, virtual sensor readings, and RUL values. Figure 1 aids in conceptualizing the core components of a turbofan engine, as represented in the NCMAPSS dataset [13].

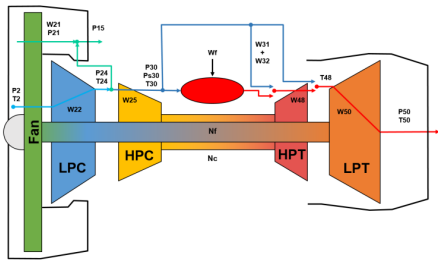

Fig. 1: NCMAPSS Turbofan engine diagram, taken from [13]

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 PHMAP 2021 Data Challenge metrics to develop comparable results [13]. Given two vectors $y, \hat{y} \in \mathbb{R}^m$ representing real RUL labels and RUL estimates, the loss function for the predictive model is defined as (1):

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

NASA's scoring function is popularly used in aeronautics since it's an asymmetric loss function with higher penalties for overestimates [14].

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 Pinball Loss (2), subject to a specified quantile τ.

$$\mathcal{L}_\tau(y, \hat{y}) = \frac{1}{m}\left[(y - \hat{y})\tau \mathbb{1}\left\{y \geq \hat{y}\right\} + (\hat{y} - y)(1 - \tau)\mathbb{1}\left\{\hat{y} > y\right\}\right] \tag{2}$$

IV. Methodology

In this section, the methodology employed in the study is outlined. For a visual overview of the process, readers are directed to the accompanying diagram in Figure 2.

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

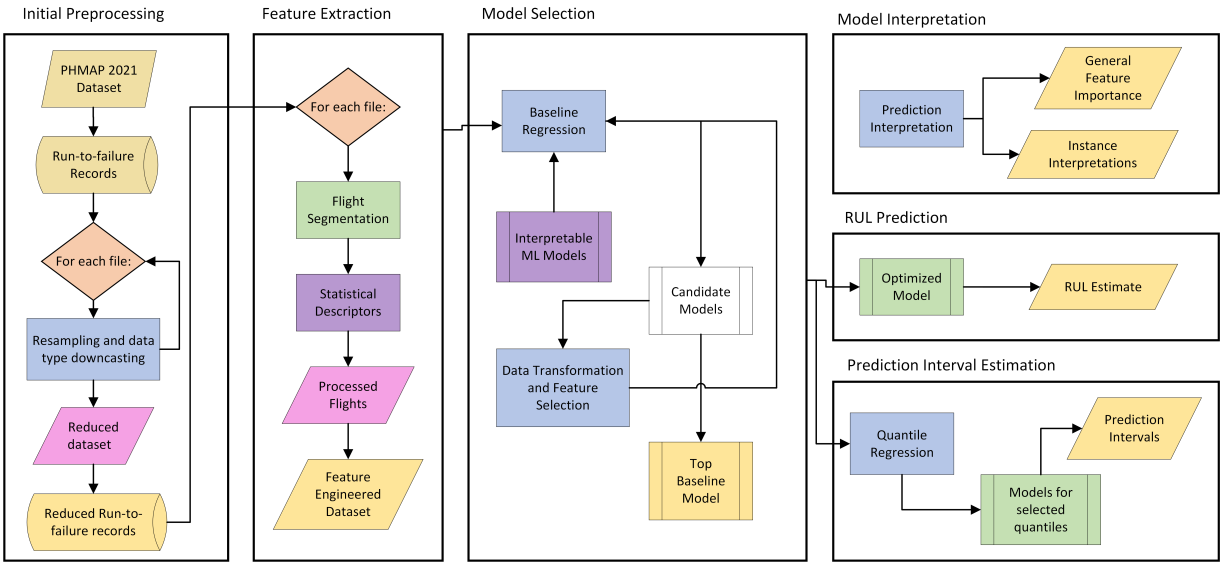

Fig. 2: Proposed Methodology Overview

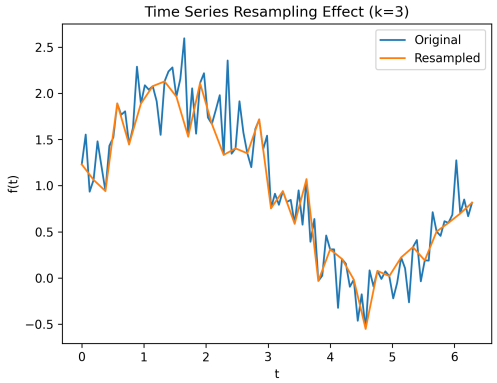

Fig. 3: Example of time series resampling (keep every kth record)

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

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 research project, highlighting the impact of these strategies on data manageability.

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's hypothesized that valuable time series information resides within its segments. By emphasizing segmentation and feature extraction at this level, researchers 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 length T having d variables:

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

2) Binary Segmentation: In dealing with extensive flight data processing, the use of approximate methods for time series segmentation proves 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 $\{158, 332, 500\}$.

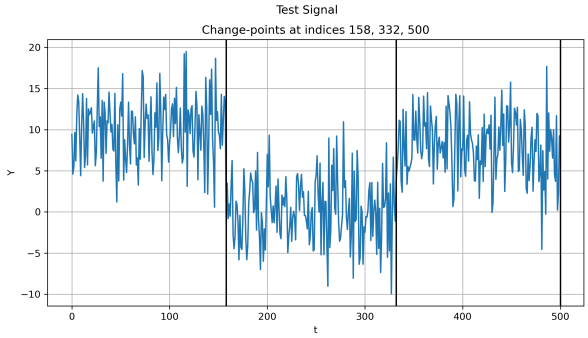

Fig. 4: Example of change points in univariate time series

Change point detection consists in searching for the optimal segmentation $\mathcal{T}$ of a time series y [15]. 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 (5):

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

Binary Segmentation is an approximate method to estimate the real set $\mathcal{T}$, it's a sequential greedy algorithm [15], [16] that estimates the first change point as described in (6):

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

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.

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 α is a constraint on the minimum size of S_i. A set of descriptors $\mathcal{F}_s = \{f \mid f : \mathcal{M}_d^{T_i}(\mathbb{R}) \to \mathbb{R}^d\}$ [1] 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 $\hat{F} \in \mathbb{R}^{d \cdot |\mathcal{F}_s| \cdot |S|}$.

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 [17]. 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 a 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 [2], calculating PHMAP loss and RMSE for each and recording total training times.

TABLE III: Proposed set of ML models

Interpretable Models	
Linear Regression [18]	Ridge Regression [18]
Lasso [18]	LassoLars [18]
ElasticNet [18]	Random Forest [18]
XGBoost [19]	LightGBM [20]
Catboost [21]	SVM [18]
Decision Tree [18]	Dummy Regression (Mean) [18]

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

In the current landscape of machine learning and AI, model interpretability and accountability hold paramount importance. These qualities are not just regulatory requirements; they

[1] Given the frequent calls to these statistical descriptor functions, prioritizing efficient implementations is essential to streamline computational resources for timely analysis.

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

are markers of responsible AI development. Laws such as the General Data Protection Regulation (GDPR) [22] and the Algorithmic Accountability Act [23] 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? [24]

Shapley values offer a model agnostic approach to post-hoc model interpretation [25], where one explains a single prediction given the original input features. They possess vital properties for effective model interpretability:

- **Efficiency:** feature contributions express how a single prediction deviates from the mean prediction of $\hat{f}$
- **Symmetry:** two features have the same Shapley Value if and only if their contributions are the same
- **Null Feature:** a feature that has no effect on a prediction has a zero Shapley value
- **Linearity:** the contribution of an ensemble of models is the same as the sum of the contribution of each model in the ensemble

Efficient approximations exist, catering to specific model types like linear, tree-based, and deep learning models. These approximations balance accuracy with computational feasibility [25] [26] [27], making Shapley Values practical for understanding various machine learning models [8] [9] [28].

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%) [29].

Quantile regression is a powerful technique used to create prediction intervals [11] [12] [7]. 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 [30] [10].

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 [31]:

$$\mathcal{Q}_\tau(Y|X) = \arg\min_{q(X)} \mathbb{E}[\mathcal{L}_\tau(Y, q(X))] \tag{7}$$

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 [32]. 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:

$$p(x|y) = \begin{cases} g(x) & y < y^* \\ b(x) & y \geq y^* \end{cases} \tag{8}$$

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 γ 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 (9). This ratio serves as an approximation for the Expected Improvement function.

$$\begin{aligned} S_g &= \{x : x \sim g(x)\} \\ x_s &= \arg\max_{x \in S_g} \frac{g(x)}{b(x)} \end{aligned} \tag{9}$$

V. Experimental Results

By resampling every 5 observations and converting 64-bit floats to 32-bit floats, the PHMAP dataset experienced a substantial 26-fold reduction in its total size (see Table IV). The resulting processed data files are subsequently saved as parquet files, preserving the specified data types.

Following the application of Binary Segmentation 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 aircraft within the dataset over a sample of its operational life.

TABLE IV: 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

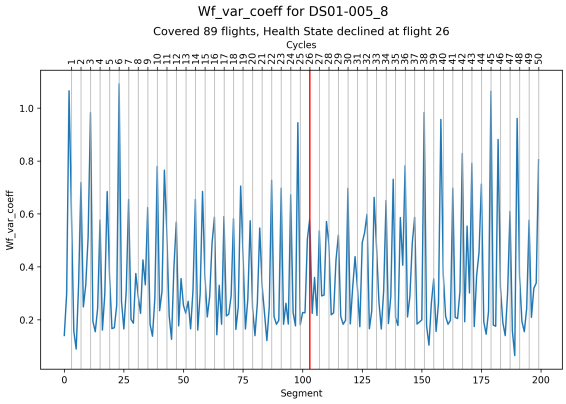

Fig. 5: Coefficient of Variation for a random plane throughout its operational life

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 V. 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 and its GPU acceleration capabilities (not used for this test), which streamline the training 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 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 VI.

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

Parameter	Range	RUL	Lower	Upper
Variance Threshold	$U(0.1, 1)$	0.10208	0.1	0.9447
Learning Rate	$\exp(U(-4.5, 0))$	0.02114	0.1163	0.042279
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	$U(0.01, 1)$	0.72610	0.85454	0.79559
Gamma	$U(0, 100)$	62.667	0.12458	0.10927
Alpha	$U(0, 100)$	8.7213	18.8598	11.3874
Lambda	$U(0, 100)$	34.832	20.921385	82.70694
Loss	-	5.6873	1.853	2.3995

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

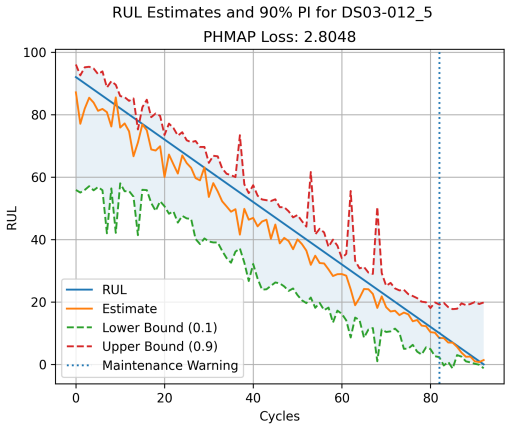

Fig. 6: RUL estimates and PI for the operational life of a plane

To offer a more profound insight into the predictions generated by the Remaining Useful Life (RUL) estimator, Shapley Values were computed for each prediction. Figure 7 presents the distribution of Shapley values for every feature utilized by the model. For more detailed analysis, individual predictions can be examined, as demonstrated in Figure 8.

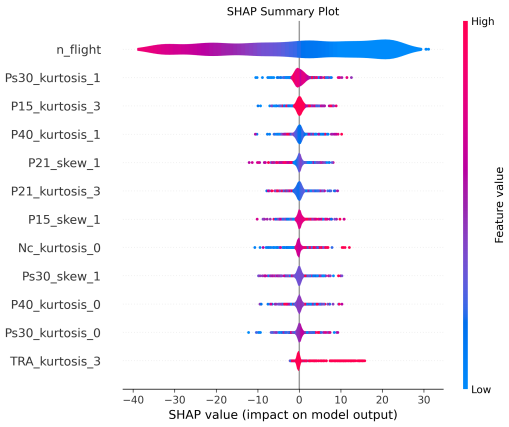

Fig. 7: Summary of Shapley Values for the dataset

TABLE V: 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	2.043735	5.321611	3.769342	9.399465	25.692841
LightGBM	2.636837	5.366961	4.823249	9.478604	2.475869
XGBoost	**0.69788**	**5.726742**	**1.304671**	**10.0481**	**6.989698**
Random Forest	2.086446	5.898664	3.837787	10.338893	223.095046
Ridge	6.340865	6.775478	11.033299	11.746066	0.043501
Elastic Net	6.888502	7.023012	11.872991	12.126897	0.029605
Lasso	6.933741	7.053696	11.937772	12.170086	0.028591
Lasso Lars	6.933741	7.053696	11.937772	12.170086	0.043695
SVM	6.878442	7.168354	11.893748	12.372685	12.91969
Decision Tree	0.0	9.472139	0.0	14.871512	3.39316
Dummy (mean)	16.77968	17.324815	23.688969	24.351877	0.000823
Linear Regression	5.142092	87743.231556	9.112421	14.664064	0.246407

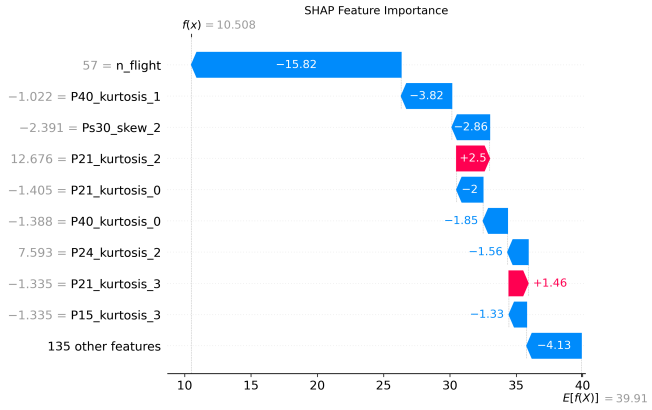

Fig. 8: Example of Shapley Values for a single prediction

VI. Conclusion

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 par with those achieved through Deep Learning methods, there was a slight reduction in overall performance. 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. However, researchers should be prepared for the increased complexity of the process and the need for robust error handling.

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.

VII. Resources

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

TABLE VII: 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.

VIII. Acknowledgments

The author extends heartfelt gratitude to Dr. Jonathan Montalvo-Urquizo and Dr. María Guadalupe Villarreal Marroquín for their invaluable guidance and unwavering support throughout the research project. Their profound expertise and mentorship significantly influenced the development of this work, and their contributions are deeply appreciated.

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] 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.

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

[5] 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.

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

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

[12] 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.

[13] 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.

[14] 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.

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

[16] 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.

[17] 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.

[18] 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.

[19] 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

[20] 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.

[21] 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

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

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

[24] 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.

[25] 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.

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

[27] 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.

[28] 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.

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

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

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

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