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

Juan Echeagaray

School of Engineering and Sciences
Tecnologico de Monterrey

November 28, 2023

Agenda

Introduction

Why predictive maintenance?

Maintenance scheduled by data analytics on historical data:

- Enhance operational efficiency.
- Sustainability and cost reduction by up to 40% [1].
- Competitive advantage.
- **Safety at the forefront.**

Figure: Turbine failure mid-flight [2]

Predictive maintenance faces 2 main challenges:

Implementation

Need a model to estimate the RUL of a set of machinery, in an efficient and reliable manner.

Interpretability

Need to *understand* why a model produces a given RUL estimate, becomes more important in critical environments.

Objectives

What are we going to do?

Objectives

This research project aims to develop:

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

Figure: Sample of a uniform fleet of cars

Objectives

Scope

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.

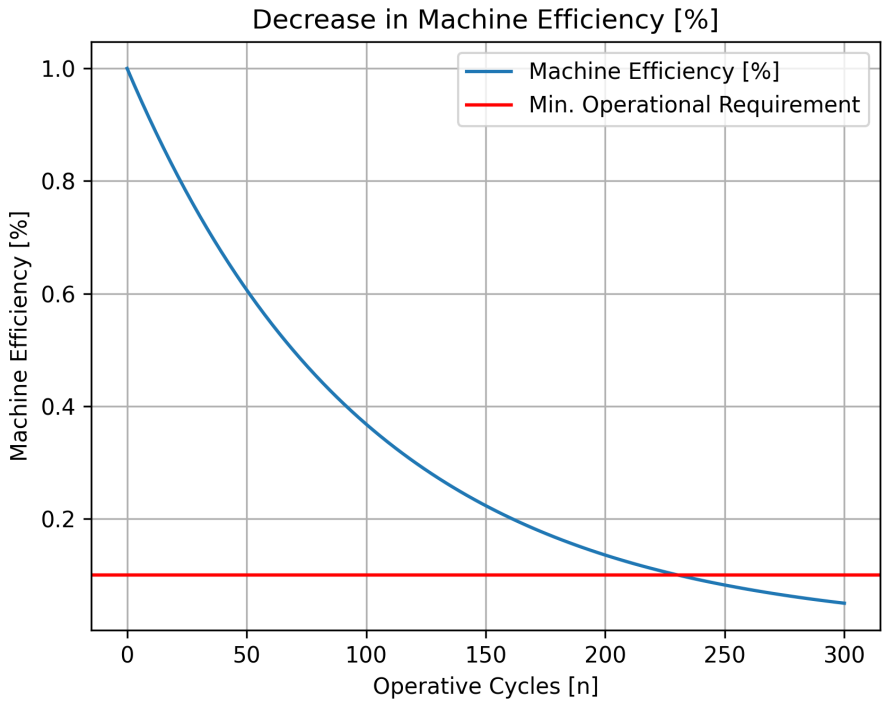


Figure: Efficiency loss as a machine operates

The 2021 PHMAP challenge proposed the loss function (1); the average of the RMSE and NASA's scoring function [3].

$$\mathcal{L}(y,\hat{y}) = \frac{1}{2}\left(\sqrt{\frac{1}{m}\sum_{i=1}^{m}(y_i-\hat{y}_i)^2} + \frac{1}{m}\sum_{i=1}^{m}\exp(\alpha\cdot(y_i-\hat{y}_i)) - 1\right) \tag{1}$$

$$\alpha = \begin{cases} \frac{-1}{10} & \text{if } y_i - \hat{y}_i \leq 0 \\ \frac{1}{13} & \text{if } y_i - \hat{y}_i > 0 \end{cases}$$

As a remark, (1) is an asymmetric loss function with a higher penalty for overestimates.

Exploratory Data Analysis

What kind of data do we have?

Exploratory Data Analysis

NCMAPSS Dataset

Flight conditions and readings from a fleet of turbofan engines, derived from NASA's CMAPSS model, including real flight conditions and relates the degradation process to the operating history of the machine. [4]

- Used in the PHMAP 2021 Data Challenge [5]
- State of the art prognosis dataset (akin to MNIST and CIFAR for CV)

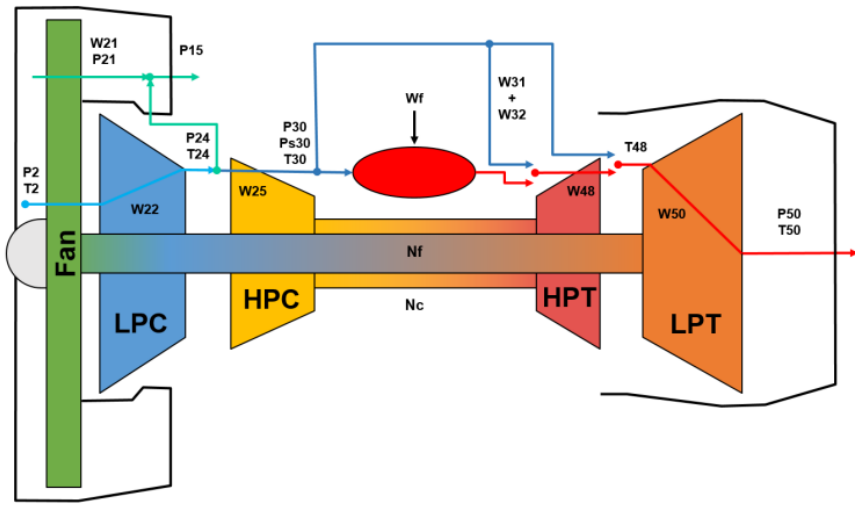


Figure: CMAPSS turbofan engine schematic

Exploratory Data Analysis

Data Overview

- Split into 10 h5 files
- Sampling frequency of 1 second
- Contains sensor readings, environmental descriptors, auxiliary variables, virtual sensor readings and RUL values

Symbol	Description	Units
alt	Altitude	ft
Mach	Flight Mach number	-
TRA	Throttle-resolver angle	%
T2	Total temperature at fan inlet	°R
Wf	Fuel flow	pps
Nf	Physical fan speed	rpm
Nc	Physical core speed	rpm
T24	Total temperature at LPC outlet	°R
T30	Total temperature at HPC outlet	°R
T48	Total temperature at HPT outlet	°R
T50	Total temperature at LPT outlet	°R
P15	Total pressure in bypass-duct	psia
P2	Total pressure at fan inlet	psia
P21	Total pressure at fan outlet	psia
P24	Total pressure at LPC outlet	psia
Ps30	Static pressure at HPC outlet	psia
P40	Total pressure at burner outlet	psia
P50	Total pressure at LPT outlet	psia
RUL	Remaining Useful Life	cycles
unit	Unit number	-
cycle	Flight cycle number	-
Fc	Flight class	-
hs	Health state	-

Table: General description of dataset variables [5]

Exploratory Data Analysis

Sample Operating Conditions

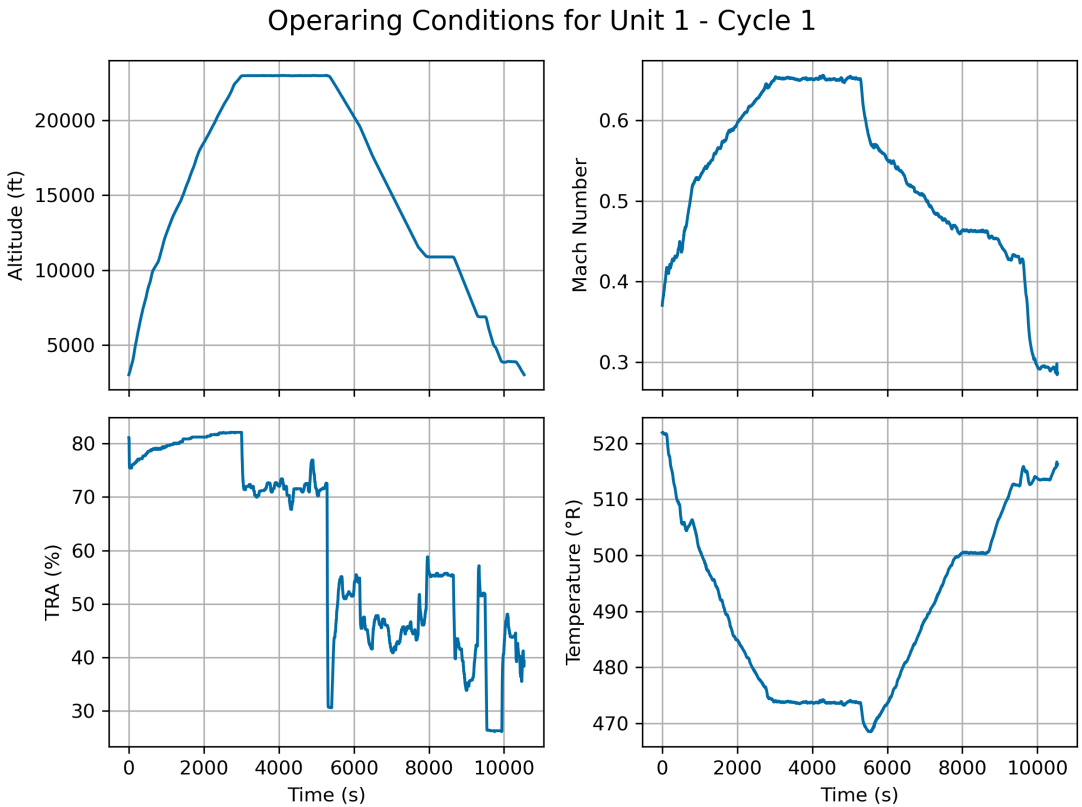


Figure: Operating conditions for the first flight of Unit 1

Exploratory Data Analysis

Operating Conditions per Flight Class

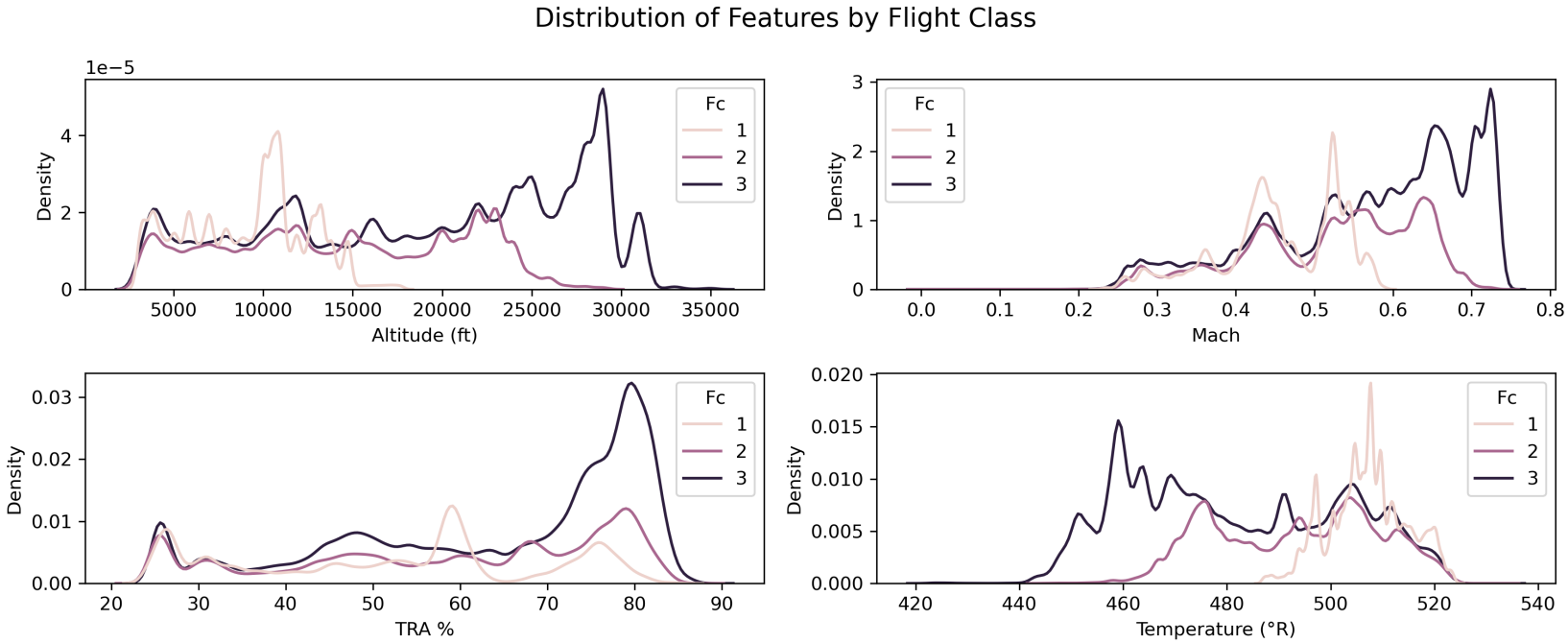


Figure: Distribution of operating conditions per Flight Class

Distinctive operating conditions per FC where FC2 can be seen as a mixture of FC1 and FC3

Exploratory Data Analysis

Final Sensor Readings

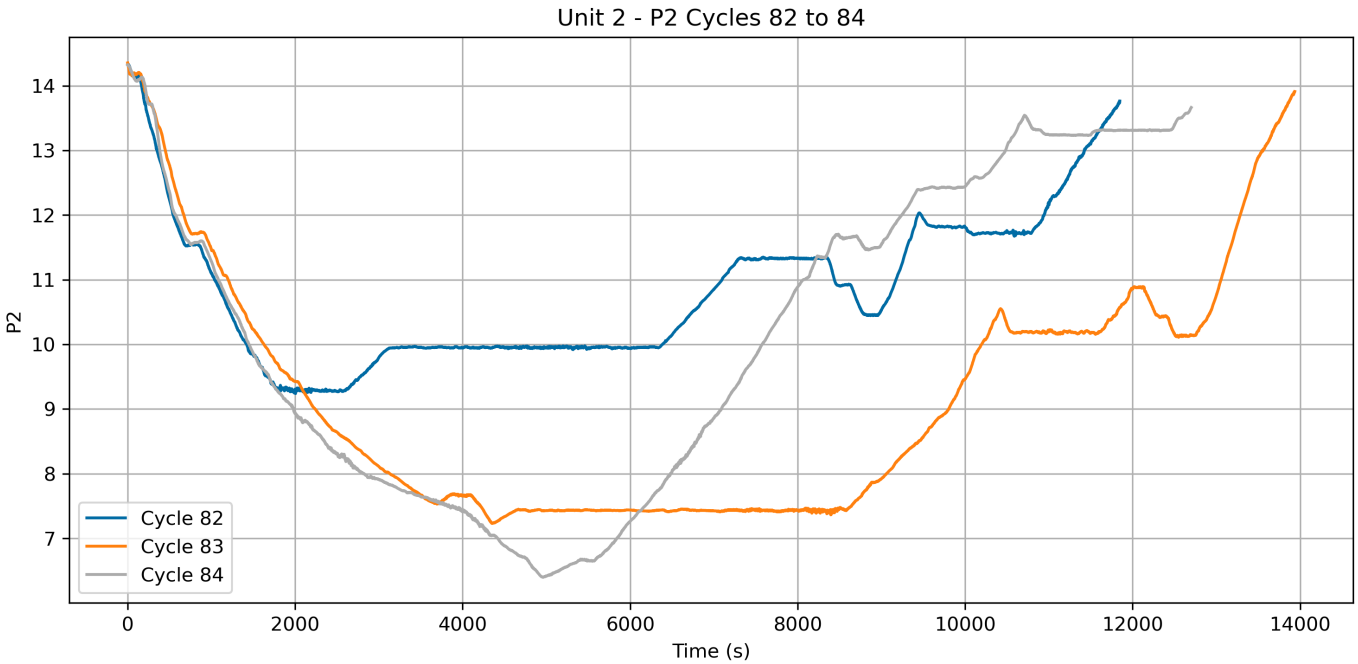


Figure: Total Pressure at fan inlet (P2) for the last 3 cycles of unit 2

Smoothness degradation with a presence of new plateaus near the end of the flight

Methodology

How are we going to do it?

Methodology

Proposed Methodology

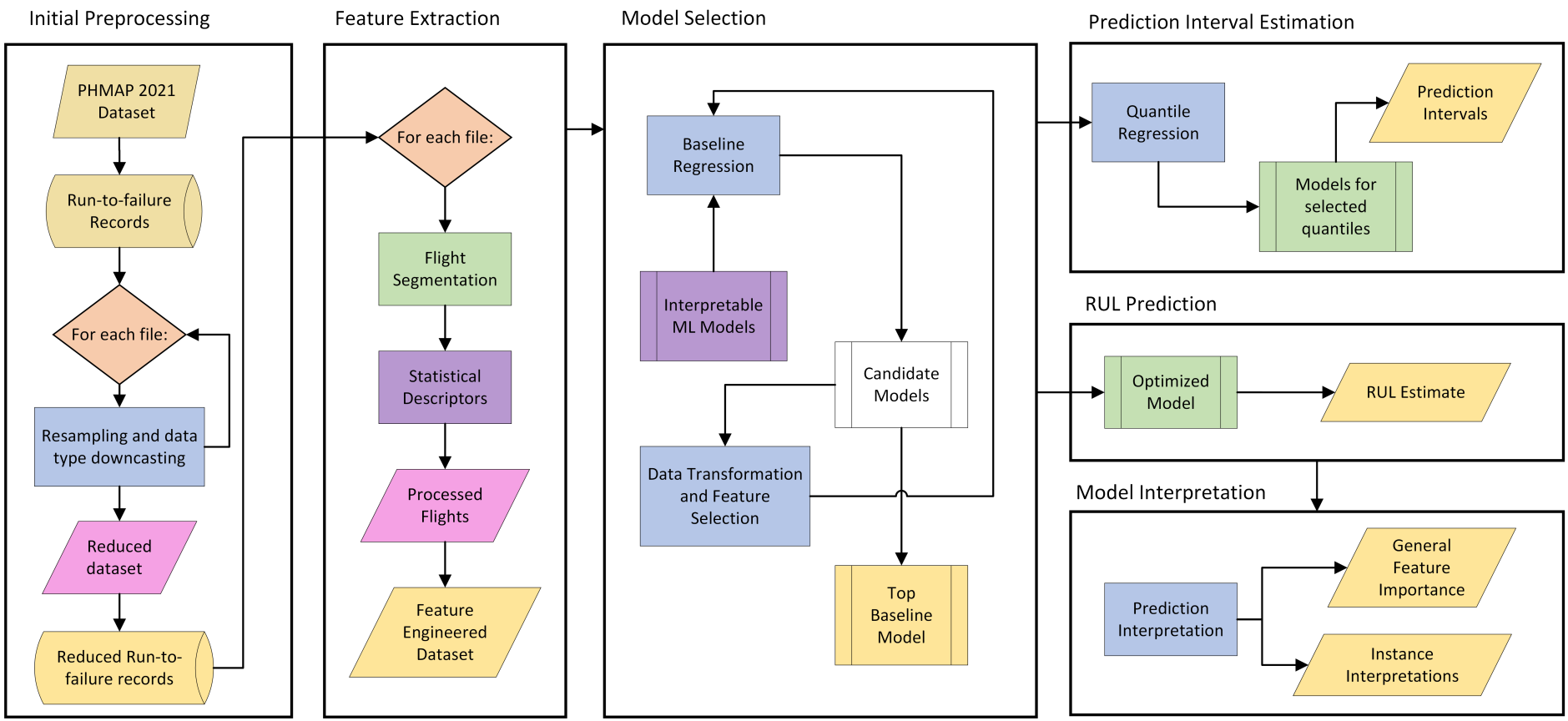


Figure: Proposed Methodology

Initial Preprocessing

Data Resampling and Downcasting

Sensor readings every second can lead to large arrays, think about sampling frequency (1 Hz, 1 KHz, 1 MHz, etc. . .) and data types.

- Resampling to catch the general shape
- Downcasting to a datatype with lower memory footprint (64bit float to 32bit float)

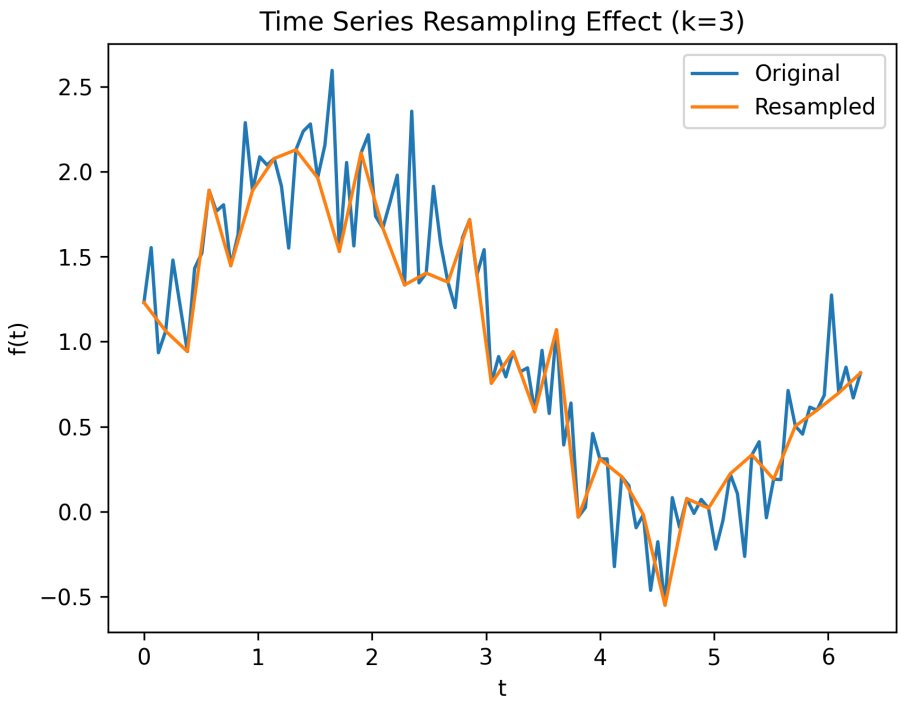


Figure: Time Series Resampling and Downcasting Effect

Problem

Dataset consists of a set of multivariate time series of varying length with a distinct number of events per series

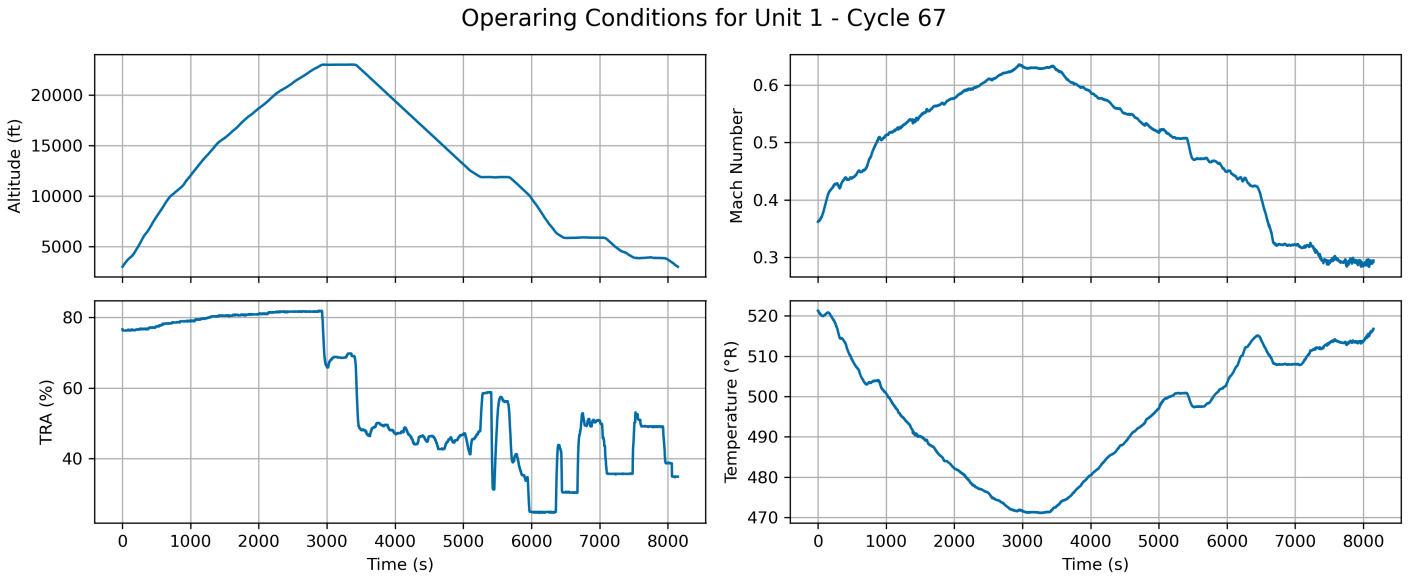


Figure: Operating conditions of Plane 1 during its 67th flight

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 $\mathcal{T} = \{t_1^*, t_2^*, \ldots, t_n^*\}$ coding the $n-1$ stages of Y

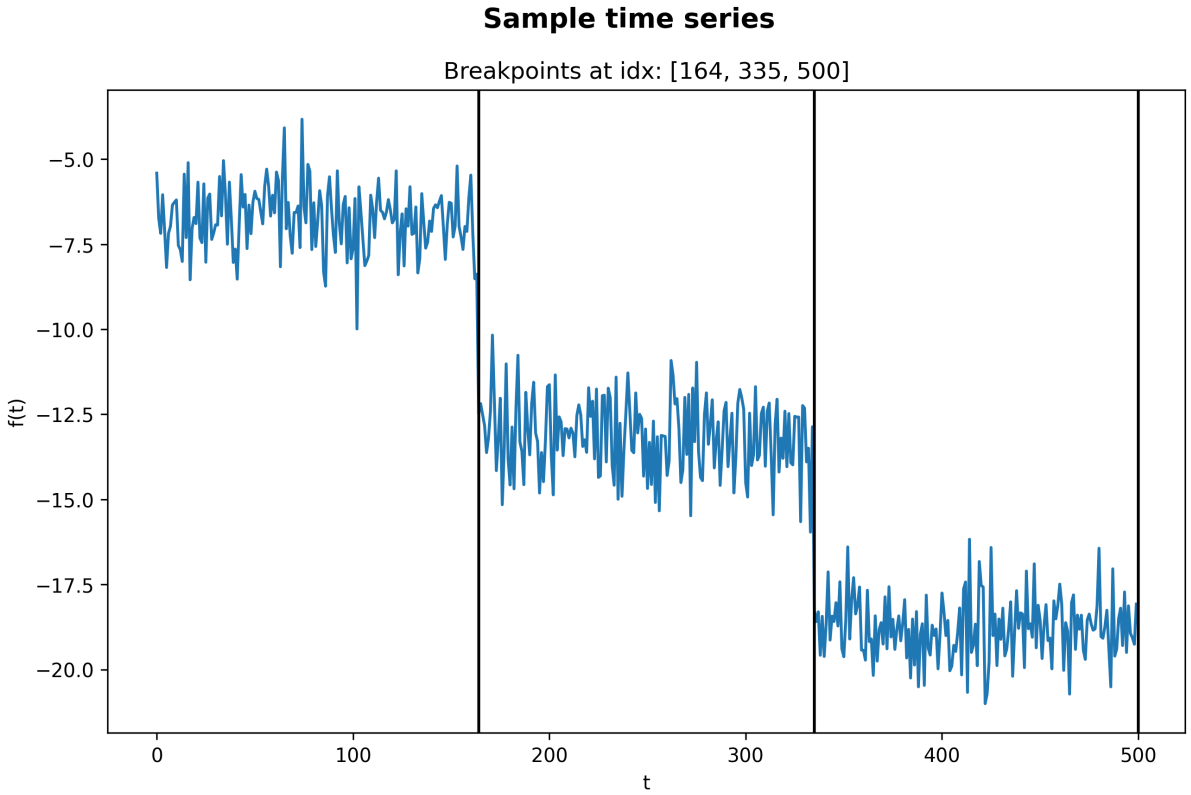


Figure: Piecewise constant signal with normal noise

Binary Segmentation as a greedy sequential algorithm that estimates 1st change-point as:

$$\hat{t}_1 := \underset{1 \leq t < T-1}{\arg\min} \; C(y_{0\ldots t}) + C(y_{t\ldots T}) \tag{2}$$

It then repeats the operation on both left and right segments until the specified number of splits is achieved.

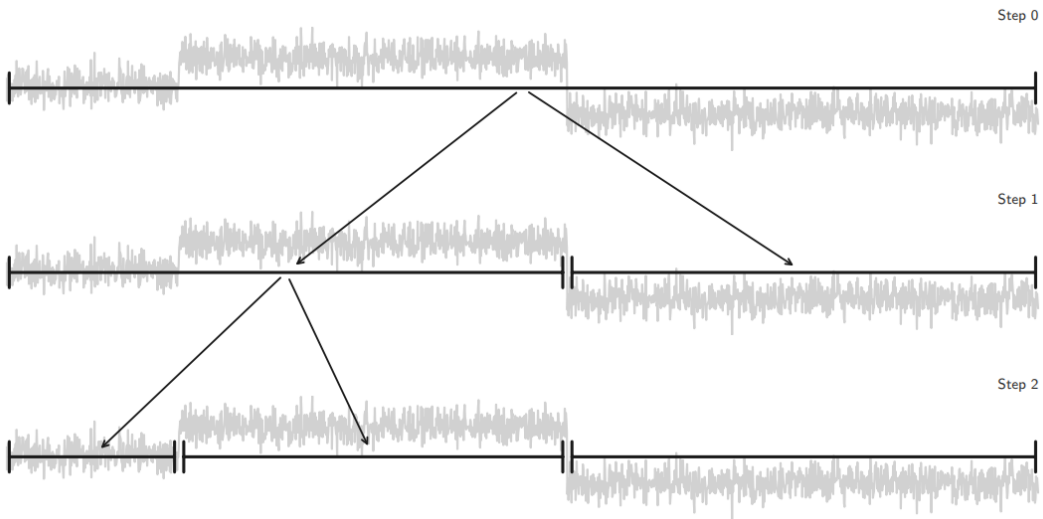


Figure: Overview of Binary Segmentation retrieved from [6]

Take the **L2 norm** An estimator of shifts in the central point of a distribution. Given a series $\{y_t\}_{t\in\mathcal{I}}$ where $y_t \in \mathbb{R}^d$:

$$C(y_{\mathcal{I}}) = \sum_{d}\sum_{t\in\mathcal{I}} ||y_t - \bar{y}||_2^2 \tag{3}$$

where $\bar{y}$ is the component wise mean of $y_{\mathcal{I}}$

- Needs scaling, considering not to penalize on segment length

Feature Extraction

Statistical Descriptors

Given a segment S_i 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. We then use a set of descriptors $F_s = \{f \mid f : \mathcal{M}_d^{T_i}(\mathbb{R}) \to \mathbb{R}^d\}$ as an example:

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

Applying each element of F_s onto S_i results in a feature vector $\hat{S}_i \in \mathbb{R}^{d \cdot |F_s|}$

A flight $F_i \in \mathcal{M}_d^{T_i}(\mathbb{R})$ contains $d \cdot T_i$ elements, the feature vector $\hat{F}_i \in \mathbb{R}^{d|F_s|\tau}$ contains $d|F_s|\tau$ elements, where $|F_s|$ is the number of statistical descriptors used and τ is the number of segments chosen. As an example, consider a flight with 5,000 timestamps, 10 features, 10 statistical descriptors and 4 segments:

$$|F_i| = 10 \cdot 5,000 = 50,000 \tag{4}$$

$$|\hat{F}_i| = 10 \cdot 10 \cdot 4 = 400 \tag{5}$$

$$\frac{|F_i|}{|\hat{F}_i|} = 125 \tag{6}$$

Feature Extraction

Pros & Cons

Pros

- Large data reduction, see (6)
- Lower computational load
- Ease feature interpretation
- Reproducible

Cons

- Mixed or uniform MVT segmentation?
- Not a fixed number of stages in each flight
- Which descriptors to use?

Model Selection

Interpretable ML models

Given the reduced dataset, one now must select a model to estimate RUL. But how can you choose without prior information?

1. Define a common loss function
2. Apply data transformation techniques
3. Select a suite of **easy to train** ML models
4. Baseline testing (w/o tuning) and ranking
5. Candidate model selection
6. Repeat!

Model Interpretation

Importance and Benefits

Interpretability enables **transparency** and **accountability**

- Analyze the weights of each feature in linear regression
- Check the splits in a decision tree
- Marginal changes in features affecting log-odds for logistic regression
- Shapley and LIME for "higher models"

Figure: Amazon removes its recruitment algorithm due to gender bias towards men [7]

Model Interpretation

Properties of Shapley values

- **Efficiency:**

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

- **Symmetry:**

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

- **Dummy:**

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

- **Additivity:**

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

Prediction Intervals

Estimating uncertainty

All predictions have an inherent level of uncertainty. "A prediction interval provides an estimate range within which a future observation is likely to fall" [8]

Machine Learning Perspective

In ML, they represent the range within which a predicted observation is likely to fall, they estimate the uncertainty of point estimates.

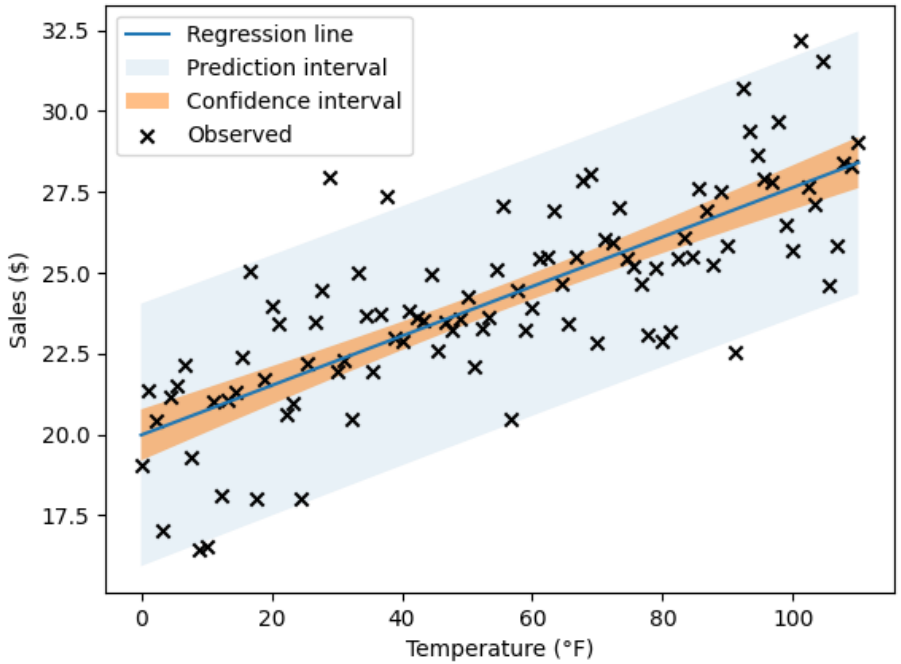


Figure: Difference between CIs and PIs [9]

Let τ be the selected quantile, y the target variable and $\hat{y}$ the predicted quantile. The **Pinball Loss** is defined as:

$$\mathcal{L}_\tau(y, \hat{y}) = (y - \hat{y})\tau \mathbb{1}\{y \geq \hat{y}\} + (\hat{y} - y)(1 - \tau)\mathbb{1}\{\hat{y} > y\} \tag{7}$$

$$\mathcal{Q}_\tau(Y|X) = \arg\min_{q(X)} \mathbb{E}[\mathcal{L}_\tau(Y, q(X))] \tag{8}$$

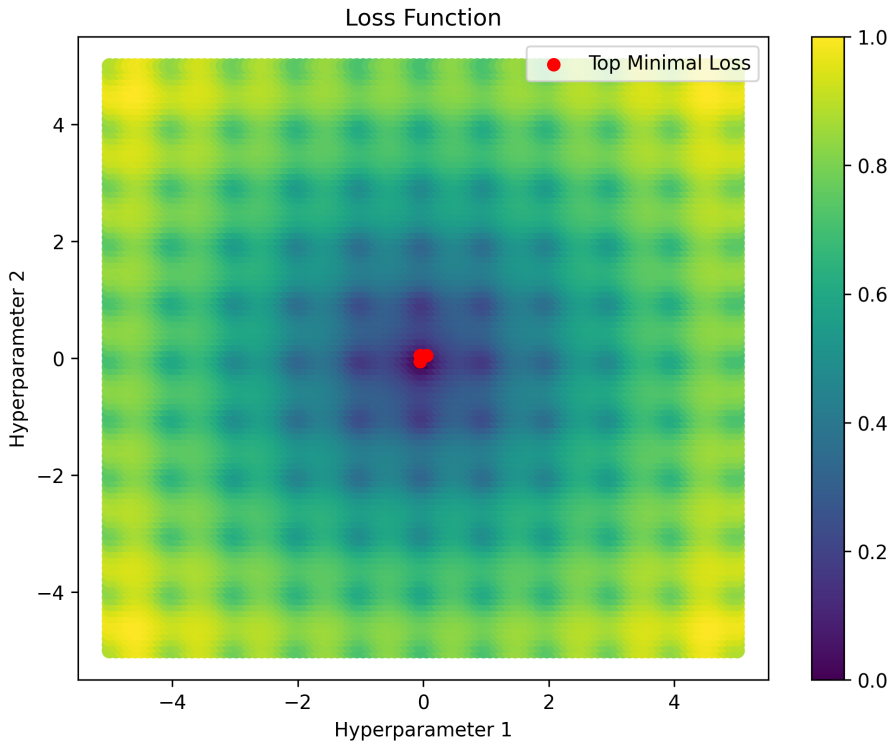
Koenker introduced the concept of quantile regression in 1978, and continued its development up to the Conditional Quantile Functions [10] referenced in (8) which minimizes (7)

Hyperparameter Optimization (HPO)

Overview

Each dataset and model is different, and there is no rigorous proof on how to determine the optimal hyperparameters for a given problem. We adopt approximate methods divided into 3 categories [11]:

- Grid Search
- Random Search
- **Bayesian Optimization**



Hyperparameter Optimization

Tree-structured Parzen Estimators

Given a search history, it suggests a hyperparameter for the next trial.

- Treats each hyperparameter **independently**
- Processes search history as tuples of (parameter, loss)
- Updates the definition of "good" and "bad" losses
- Defines $g(x)$ and $b(x)$ for good and bad losses

Heuristic

Select the hyperparameter which maximices:

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

Hyperparameter Optimization

Tree-structured Parzen Estimators Overview

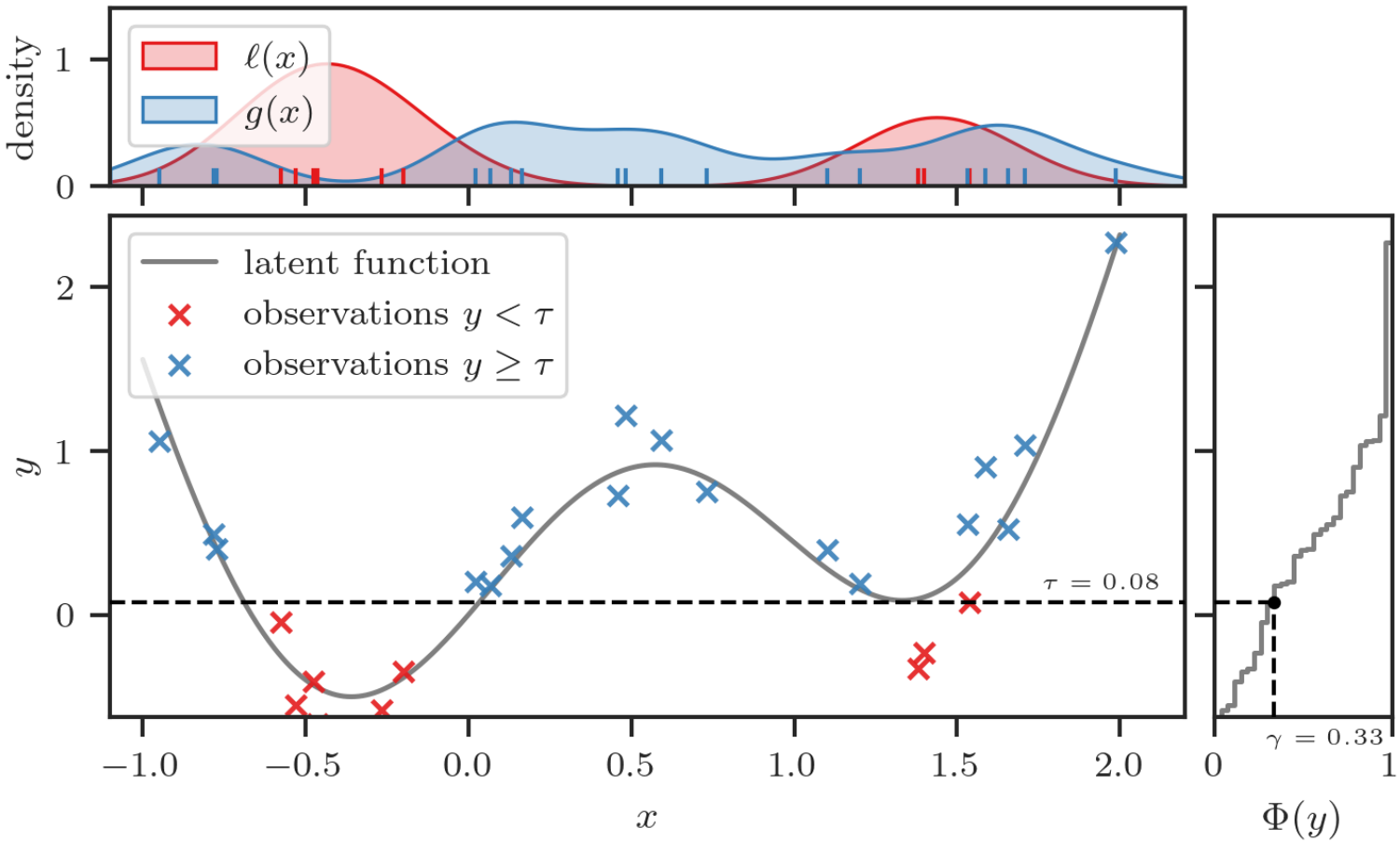


Figure: Example of TPE trial history [12]

Experimental Results

What insights emerge?

Segmentation of Environmental Descriptors

For a test case we apply Binary Segmentation with L2 norm and min. size for segment of 20% the length of the array

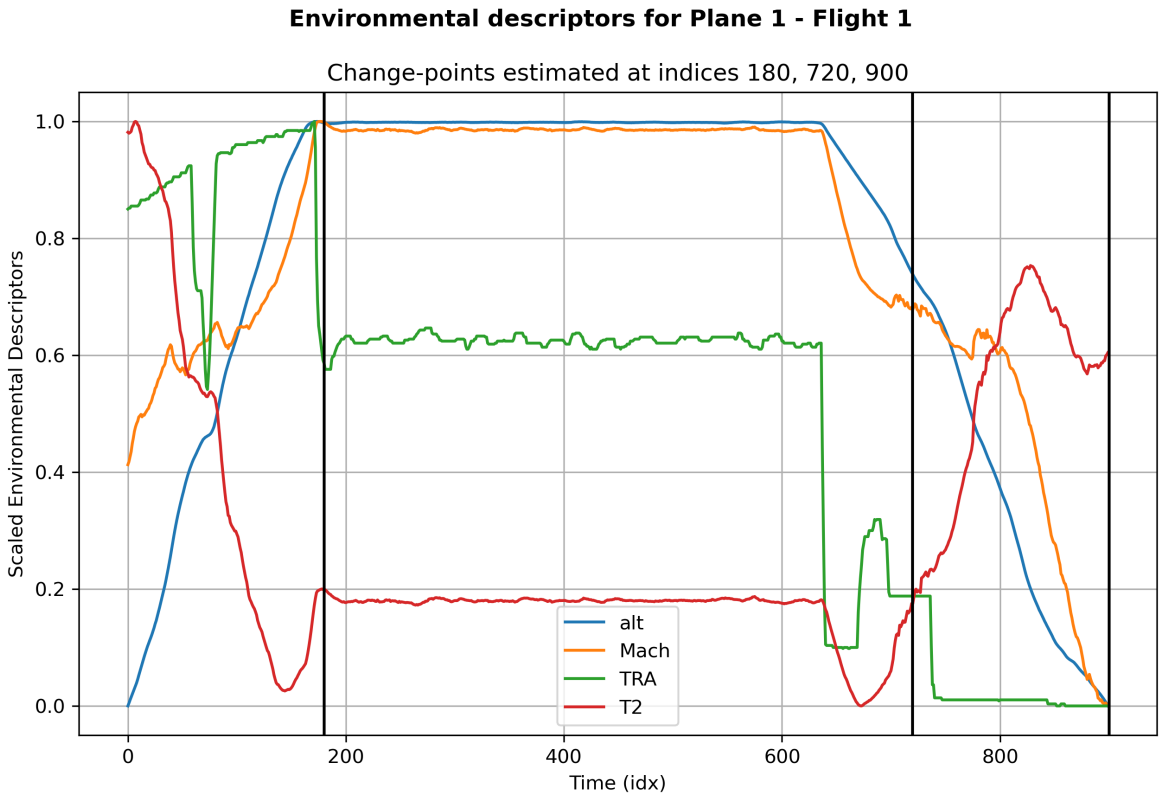


Figure: Environmental descriptors (scaled) for the first flight for plane 1

Flight History of a Plane

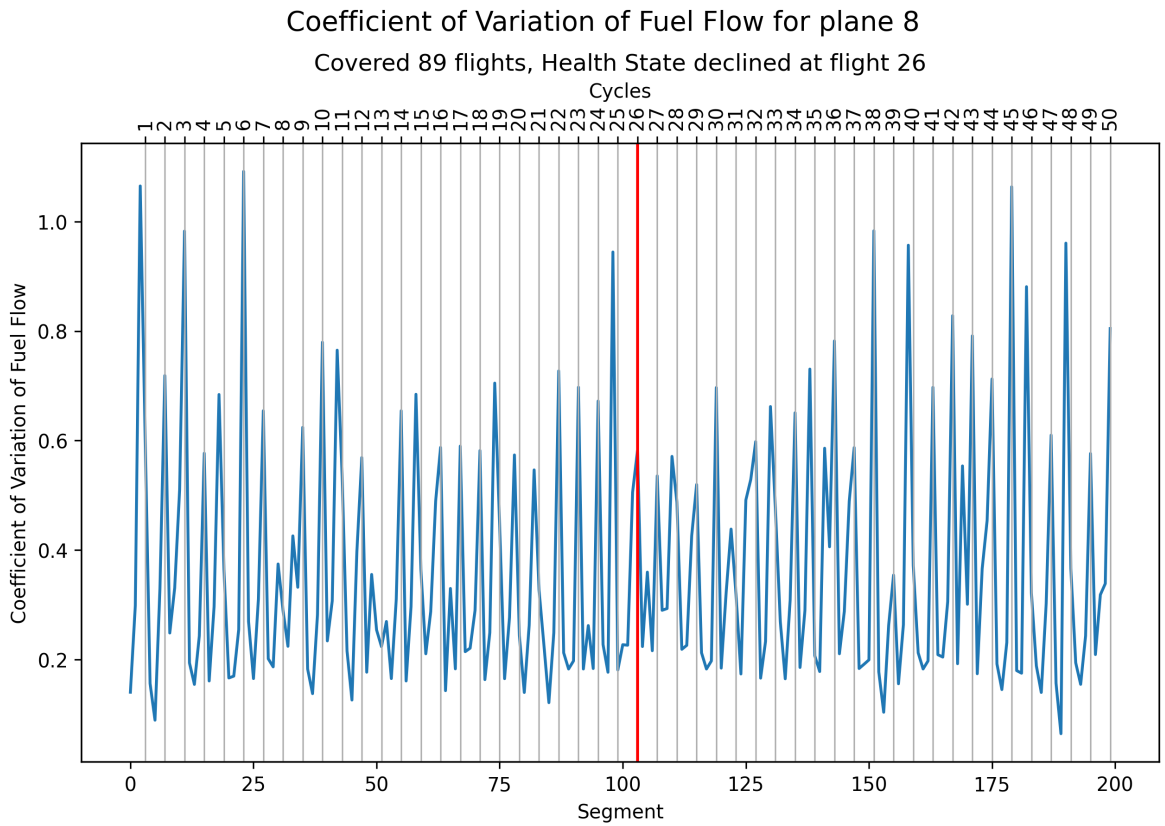


Figure: Coefficient of variation of the Fuel Flow for Unit 8. Flew 89 flights

Could there be a correlation between the early spikes and the relatively early health decline?

Baseline Regression Models

A suite of ML models is fit to 80% of a PCA processed dataset [2], tested on the remaining 20% (no GPU/TPU):

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	0.028
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	9.472	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

1. Gradient Boosting family achieves overall the same performance
2. Linear models present less overfitting but comparatively lower accuracy

[2] See Table 5 to compare model performances with raw data

TPE Hyperparameter Optimization

- Optimizes the following models: RUL estimator, Lower and Upper bounds for prediction intervals
- 500 trials per model
- Minimizes the 5-fold cross-validated test PHMAP loss

Parameter	Range
Variance Threshold	$U(0.1, 1)$
Learning Rate	$\exp(U(-4.5, 0))$
Boosting Rounds	$U_{\mathbb{Z}}(100, 10000)$
Max Depth	$U_{\mathbb{Z}}(2, 30)$
Min Child Weight	$U_{\mathbb{Z}}(2, 50)$
Subsample	$U(0.01, 1)$
Gamma	$U(0, 100)$
Alpha	$U(0, 100)$
Lambda	$U(0, 100)$

Table: Proposed hyperparameter search space

TPE Results

The total optimization procedure took *around* 8 hours (considering all models).

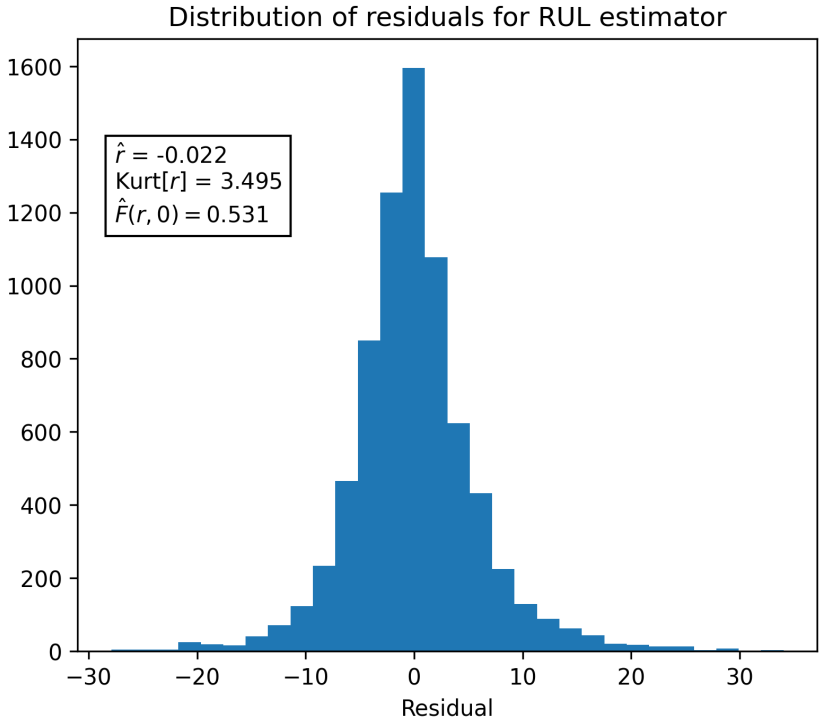
Parameter	Range	RUL	Lower Bound	Upper Bound
Variance Threshold	$U(0.1, 1)$	0.102	0.1	0.944
Learning Rate	$\exp(U(-4.5, 0))$	0.021	0.116	0.042
Boosting Rounds	$U_{\mathbb{Z}}(100, 10000)$	1963	5859	3830
Max Depth	$U_{\mathbb{Z}}(2, 30)$	30	15	20
Min Child Weight	$U_{\mathbb{Z}}(2, 50)$	29	41	38
Subsample	$U(0.01, 1)$	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

Table: TPE results after 500 trials

Residual Overview

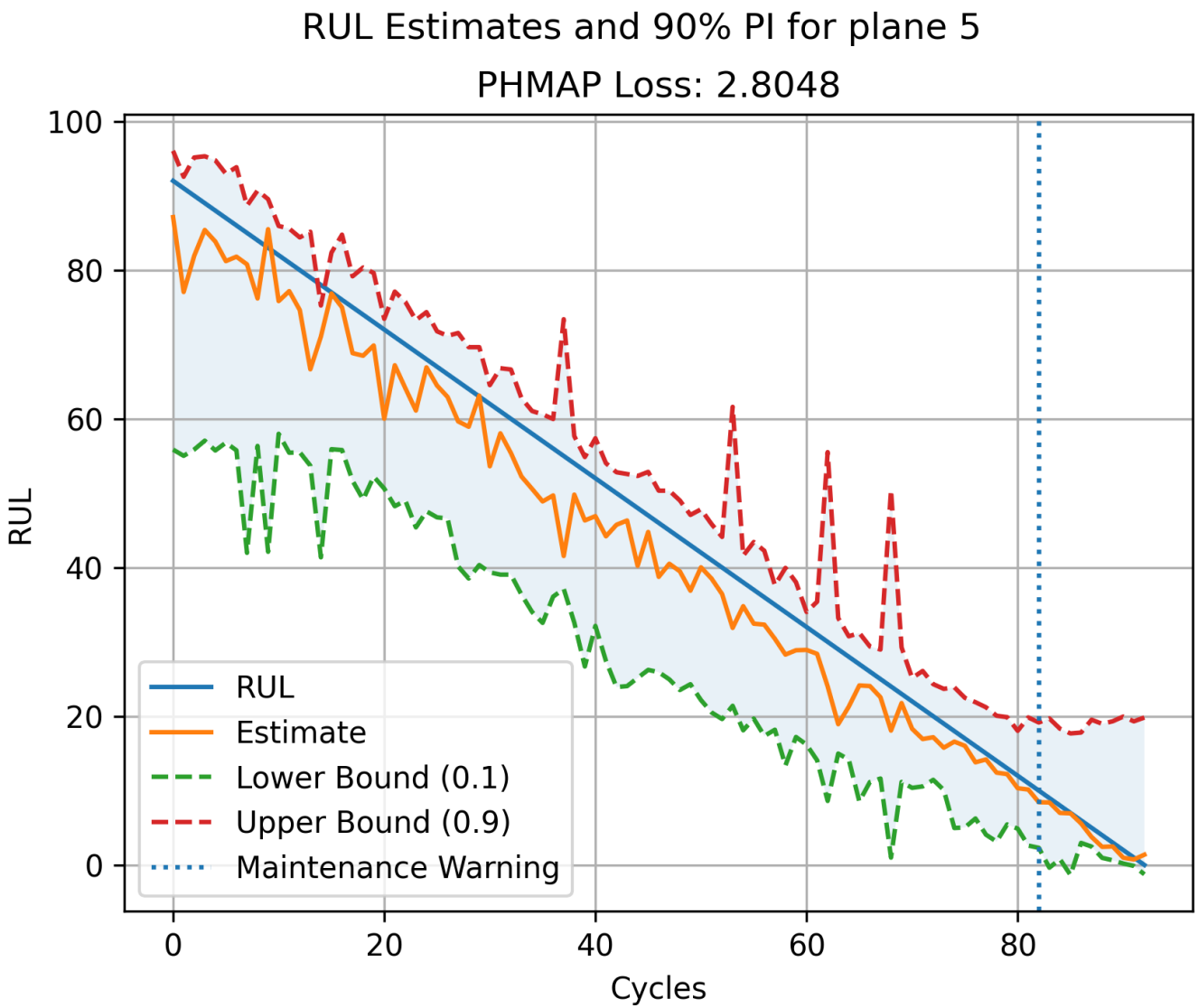
Looking deeper into the RUL estimator:

- Residuals centered at 0
- Relatively fat distribution (higher kurtosis than normal)
- ECDF at 0 is 53%



Residuals with zero mean are expected for models minimizing MSE, but *maybe* for this scenario a positive mean would be better (model which underestimates)

Sample Predictions



Model interpretation

Next we show the overall importance of each variable in RUL regression:

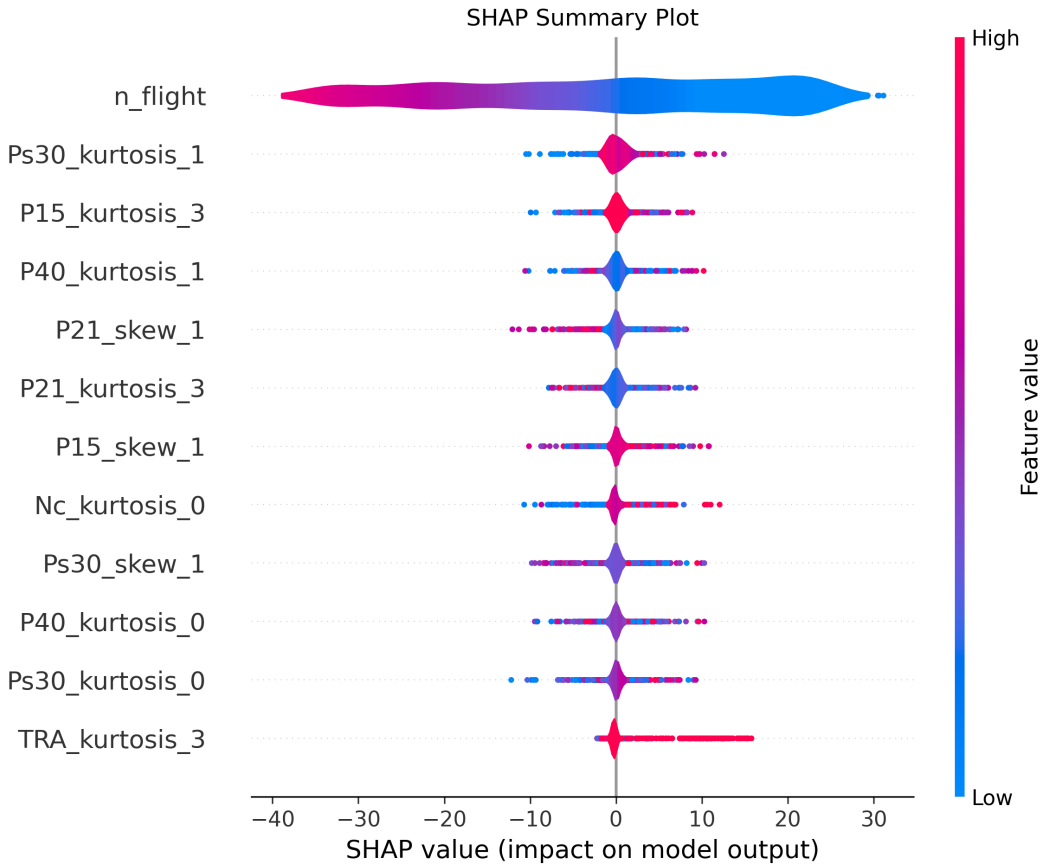


Figure: SHAP values for the Top 12 features (selected by the mean absolute value across all predictions)

Instance Explanation

We can also analyze in detail a single RUL estimate:

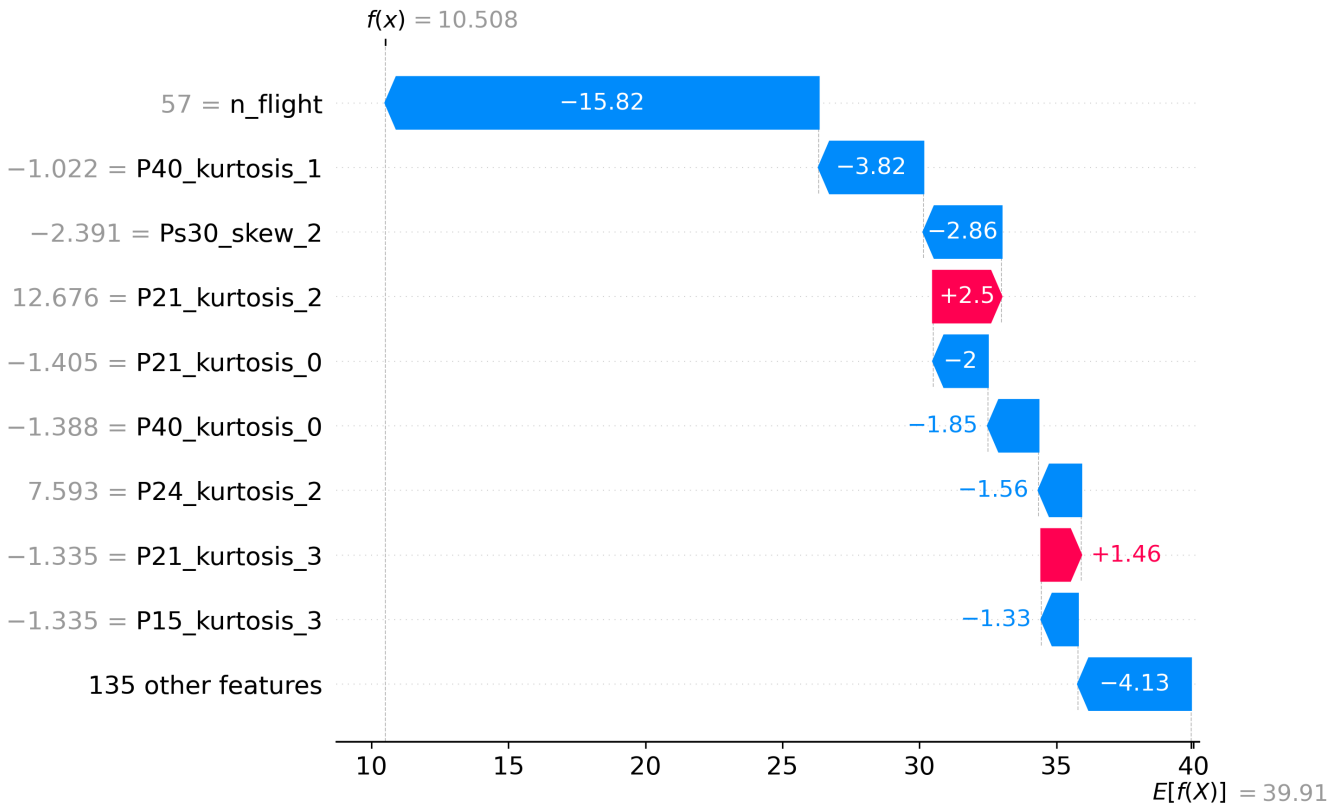


Figure: SHAP values for the Top 10 features

Instance Explanation

Shapley Values also allow to check for dependencies:

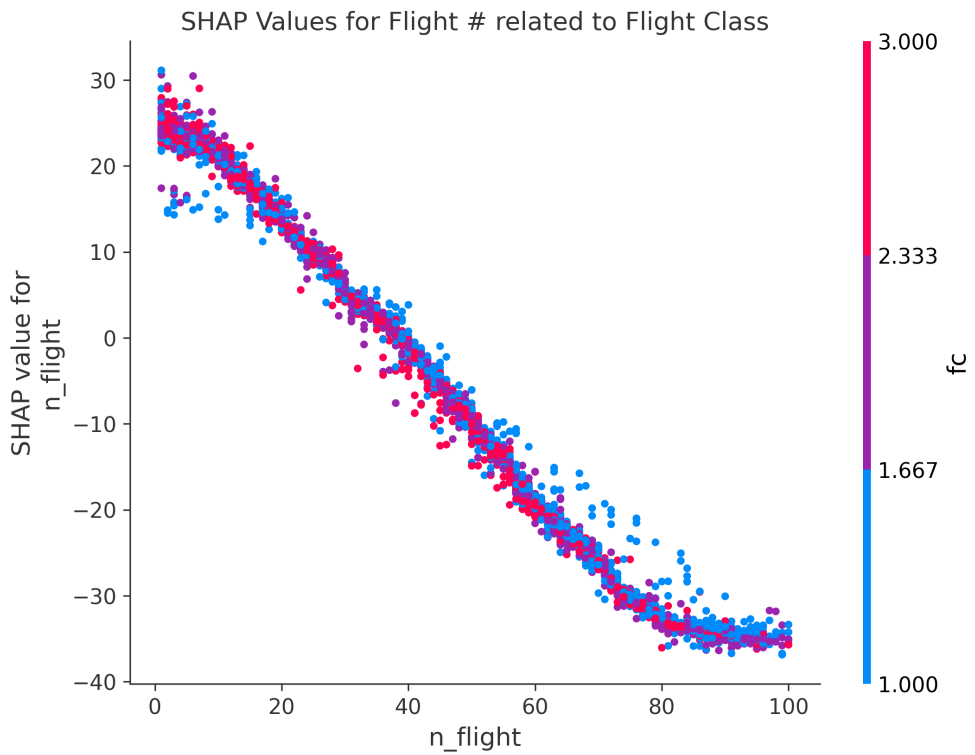


Figure: SHAP dependence plot for flight number (color coded by flight class)

Conclusions and Future Work

What have we achieved?

Conclusions and Future Work

Comparison with previous work

Slight decrease in performance but much less compute time for training and inference.

User	Model	Score
IJoinedTooLate [13]	Dilated CNN	3.006
YellowJackets [14]	Inception CNN	3.327
DatrikUS [15]	Stacked DCNN	3.651
*	**Bayesian Optimized XGBoost**	**5.687**

Table: Comparison of model performances

Conclusions and Future Work

The research has mainly accomplished:

- PdM framework with measures of **interpretability** and **uncertainty** built into it
- Extendable toolbox to apply HPO for the suite of the proposed models

But there's still work to do:

- Incorporate time series segmentation into HPO
- Use different feature selection techniques
- Apply Deep Learning?

Q&A

Questions?

References I

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

BBC News, "Boeing 777: Dozens grounded after denver engine failure," , Feb. 22, 2021. [Online]. Available: https://www.bbc.com/news/world-us-canada-56149894.

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.

References II


M. Arias 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, no. 1, p. 5, 2021.


PHM Society. (Oct. 21, 2021), 2021 phm conference data challenge - phm society data repository, [Online]. Available: https://data.phmsociety.org/2021-phm-conference-data-challenge/.


C. Truong, L. Oudre, and N. Vayatis, "Selective review of offline change point detection methods," *Signal Processing*, vol. 167, p. 107 299, 2020.


G. S. Reporter, "Amazon ditched ai recruiting tool that favored men for technical jobs", Oct. 11, 2018. [Online]. Available: https://www.theguardian.com/technology/2018/oct/10/amazon-hiring-ai-gender-bias-recruiting-engine.

References III

H. Peters, "Prediction Intervals in Machine Learning - Heinrich Peters - Medium,", Jul. 2023. [Online]. Available: https://medium.com/@heinrichpeters/prediction-intervals-in-machine-learning-a2faa36b320c.

L. Cialdella, *Understanding the difference between prediction and confidence intervals for linear models in Python*, Sep. 2020. [Online]. Available: https://lmc2179.github.io/posts/confidence_prediction.html.

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

AWS. (), What is hyperparameter tuning? - hyperparameter tuning methods explained - aws, [Online]. Available: https://aws.amazon.com/what-is/hyperparameter-tuning/#:~:text=Hyperparameter%20tuning%20allows%20data%20scientists,the%20model%20as%20a%20hyperparameter..

References IV

L. C. Tiao, A. Klein, M. W. Seeger, E. V. Bonilla, C. Archambeau, and F. Ramos, "BORE: Bayesian optimization by density-ratio estimation," in *Proceedings of the 38th International Conference on Machine Learning*, M. Meila and T. Zhang, Eds., ser. Proceedings of Machine Learning Research, vol. 139, PMLR, Jul. 2021, pp. 10 289–10 300. [Online]. Available: http://proceedings.mlr.press/v139/tiao21a.html.

A. Lövberg, "Remaining useful life prediction of aircraft engines with variable length input sequences," *Annual Conference of the PHM Society*, vol. 13, 1 Dec. 2021, ISSN: 2325-0178. DOI: 10.36001/phmconf.2021.v13i1.3108.

References V

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, 1 Dec. 2021, ISSN: 2325-0178. DOI: 10.36001/phmconf.2021.v13i1.3109.

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, 1 Nov. 2021, ISSN: 2325-0178. DOI: 10.36001/phmconf.2021.v13i1.3110.

Raw Baseline Comparison

Model	Train Loss	Test Loss	Train RMSE (cycles)	Test RMSE (cycles)	Training Time (s)
Catboost	4.426	6.319	7.919	11.037	27.665
XGBoost	3.069	6.428	5.593	11.188	6.404
LightGBM	4.362	6.484	7.814	11.285	1.881
Random Forest	2.323	6.693	4.264	11.571	310.103
Ridge	6.341	6.775	11.033	11.746	0.086
Lasso Lars	6.807	6.92	11.755	11.974	0.085
Lasso	6.807	6.92	11.755	11.974	0.13
Elastic Net	6.863	6.972	11.835	12.048	0.27
SVM	6.869	7.11	11.882	12.282	12.76
Decision Tree	0.0	9.336	0.0	14.967	4.78
Dummy (mean)	16.78	17.325	23.689	24.352	0.001
Linear Regression	5.142	87743.232	9.112	14.664	0.407

Table: Comparison of model performance without PCA

Dimensionality Reduction?

PCA on raw descriptors needs 28 components to explain 99% of the total variance:

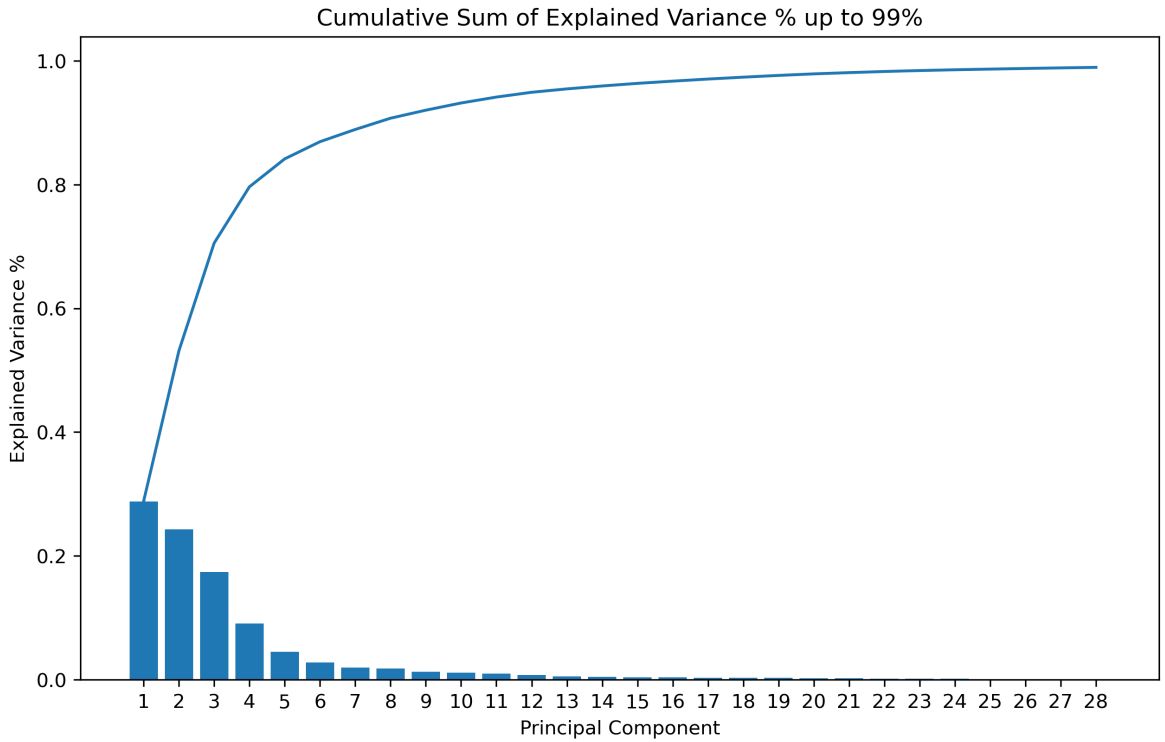


Figure: Explained variance ratio per Principal Component added up to 99%

Scaled PCA needs 93, nonetheless weights should be distributed fairly, doesn't imply better prediction performance!

Effects on Model Accuracy subject to prior Standarization

Using a XGBoost regressor as a baseline model, the following treatments were tested on a validation dataset:

Transformation	Test Loss
No PCA	6.402025
PCA	5.726742
PCA(99% Var.)	6.963744
Standardized PCA	6.746479
Standardized PCA(99% Var.)	7.448121

Table: Comparison of PCA and Standardization Treatments

PCA enables an overall improvement on model loss, but the model doesn't rely on the *most* important components.