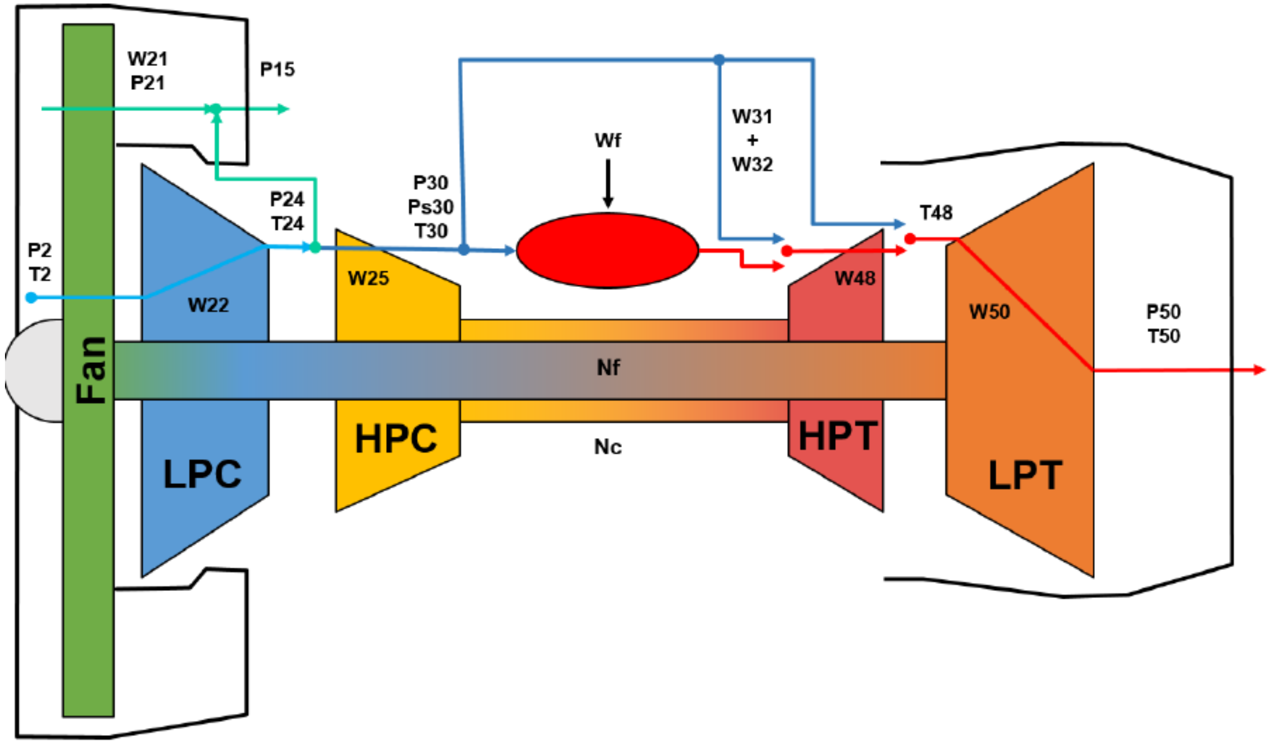
Remaining Useful Life Prediction using Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) dataset

Gregory Lim

13/3/2022

Objective

Remaining useful life (RUL) is an estimation of the leftover time or cycles that an industrial system can operate before failure. The objective of this analysis is to develop a data-driven model to predict the RUL of a fleet of aircraft engines operating under conditions of high variability in the flight envelope and multiple failure modes. Each unit of the fleet has unknown and different initial health conditions and experiences types of slowly developing faults that initiate at some time during the flight history.[1,2]



Schematic representation of the CMAPSS model as depicted in the CMAPSS documentation

[1] Manuel Arias Chao, Chetan Kulkarni 2, Kai Goebel 3 and Olga Fink. "PHM Society Data Challenge 2021". [Online] https://data.phmsociety.org/wp-content/uploads/sites/9/2021/08/2021_Data_Challenge.pdf

[2] Manuel Arias Chao, Chetan Kulkarni 2, Kai Goebel 3 and Olga Fink, "Aircraft Engine Run-to-Failure Dataset under Real Flight Conditions for Prognostics and Diagnostics". Data 2021, 6, 5. https://doi.org/10.3390/data6010005

Dataset

Turbofan Engine Degradation Simulation Data Set-2

Source: https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/#turbofan-2

Citation: M. Chao, C.Kulkarni, K. Goebel and O. Fink (2021). "Aircraft Engine Run-to-Failure Dataset under real flight conditions", NASA Ames Prognostics Data Repository (http://ti.arc.nasa.gov/project/prognostic-data-repository), NASA Ames Research Center, Moffett Field, CA

Description: The generation of data-driven prognostics models requires the availability of datasets with run-to-failure trajectories. In order to contribute to the development of these methods, the dataset provides a new realistic dataset of run-to-failure trajectories for a small fleet of aircraft engines under realistic flight conditions. The damage propagation modelling used for the generation of this synthetic dataset builds on the modelling strategy from previous work . The dataset was generated with the Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) dynamical model. The data set is been provided by the Prognostics CoE at NASA Ames in collaboration with ETH Zurich and PARC.

Files:

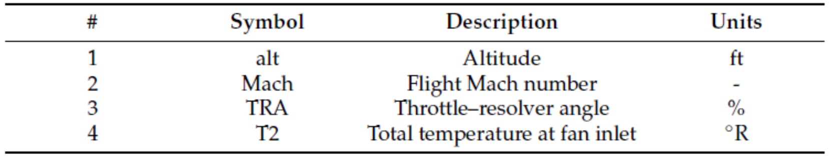
- **N-CAMPSS_DS01-005.h5** ← Dataset used in this analysis (2.8 GB)
- N-CAMPSS_DS02-006.h5
- N-CAMPSS_DS03-012.h5
- N-CAMPSS_DS04.h5
- N-CAMPSS_DS05.h5
- N-CAMPSS_DS06.h5
- N-CAMPSS_DS07.h5
- N-CAMPSS_DS08a-009.h5
- N-CAMPSS_DS08c-008.h5
- N-CAMPSS_DS08d-010.h5
- **N-CMAPSS_Example_data_loading_and_exploration.ipynb** ← Brief reference on dataset loading and description
- **Run_to_Failure_Simulation_Under_Real_Flight_Conditions_Dataset.pdf** ← Detailed description on dataset and C-MAPSS simulation environment

Dataset

N-CAMPSS_DS01-005.h5 dataset contains 10 units, 3 varying flight classes with 1 failure mode affecting the efficiency of the high pressure turbine (HPT) . The dataset contains 7,641,868 records captured during a simulated run-to-failure degradation trajectories based on 46 variables and 1 derived variable (RUL). The first 4,906,646 records relating to first six units (#1-6) are used for development while the remaining 2,735,232 records relating to the remaining 4 units (#7-10) are retained for testing purposes.

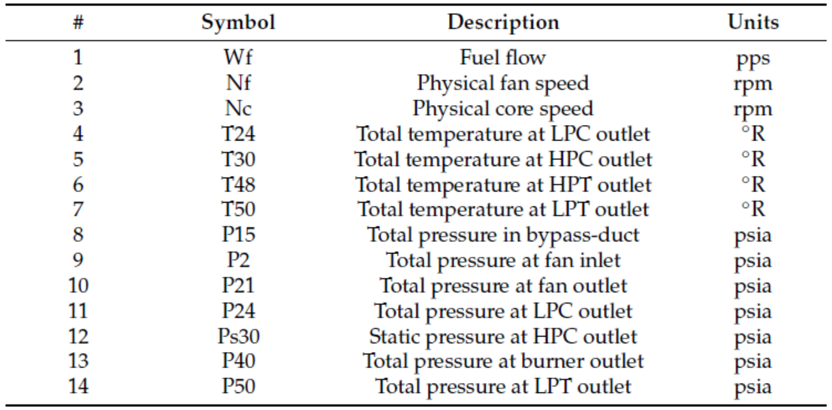
Flight classes	Flight length (h)
1	1 – 3
2	3 – 5
3	>5

Scenario descriptors



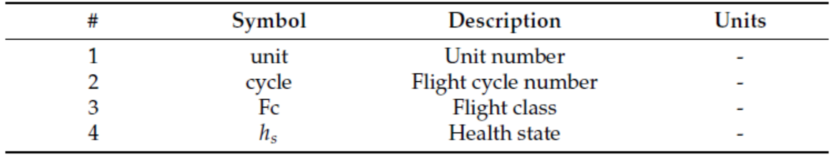
#	Symbol	Description	Units
1	alt	Altitude	ft
2	Mach	Flight Mach number	-
3	TRA	Throttle–resolver angle	%
4	T2	Total temperature at fan inlet	°R

Measurements



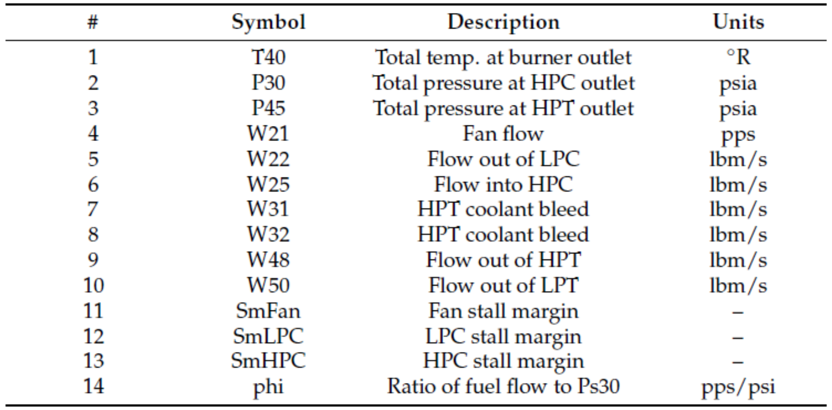
#	Symbol	Description	Units
1	Wf	Fuel flow	pps
2	Nf	Physical fan speed	rpm
3	Nc	Physical core speed	rpm
4	T24	Total temperature at LPC outlet	°R
5	T30	Total temperature at HPC outlet	°R
6	T48	Total temperature at HPT outlet	°R
7	T50	Total temperature at LPT outlet	°R
8	P15	Total pressure in bypass-duct	psia
9	P2	Total pressure at fan inlet	psia
10	P21	Total pressure at fan outlet	psia
11	P24	Total pressure at LPC outlet	psia
12	Ps30	Static pressure at HPC outlet	psia
13	P40	Total pressure at burner outlet	psia
14	P50	Total pressure at LPT outlet	psia

Auxiliary data



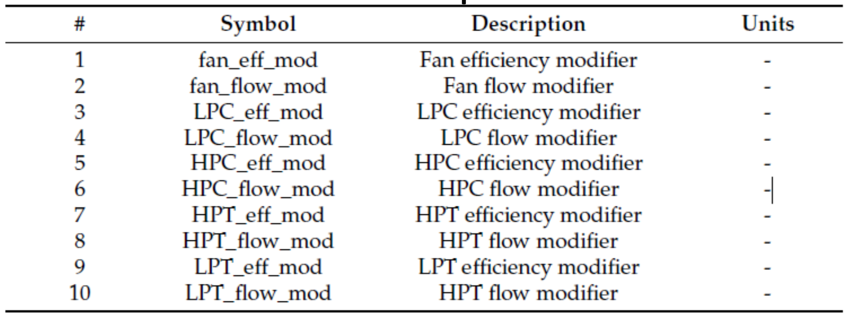
#	Symbol	Description	Units
1	unit	Unit number	-
2	cycle	Flight cycle number	-
3	Fc	Flight class	-
4	h_s	Health state	-

Virtual sensors



#	Symbol	Description	Units
1	T40	Total temp. at burner outlet	°R
2	P30	Total pressure at HPC outlet	psia
3	P45	Total pressure at HPT outlet	psia
4	W21	Fan flow	pps
5	W22	Flow out of LPC	lbm/s
6	W25	Flow into HPC	lbm/s
7	W31	HPT coolant bleed	lbm/s
8	W32	HPT coolant bleed	lbm/s
9	W48	Flow out of HPT	lbm/s
10	W50	Flow out of LPT	lbm/s
11	SmFan	Fan stall margin	–
12	SmLPC	LPC stall margin	–
13	SmHPC	HPC stall margin	–
14	phi	Ratio of fuel flow to Ps30	pps/psi

Model health parameters



#	Symbol	Description	Units
1	fan_eff_mod	Fan efficiency modifier	-
2	fan_flow_mod	Fan flow modifier	-
3	LPC_eff_mod	LPC efficiency modifier	-
4	LPC_flow_mod	LPC flow modifier	-
5	HPC_eff_mod	HPC efficiency modifier	-
6	HPC_flow_mod	HPC flow modifier	-
7	HPT_eff_mod	HPT efficiency modifier	-
8	HPT_flow_mod	HPT flow modifier	-
9	LPT_eff_mod	LPT efficiency modifier	-
10	LPT_flow_mod	HPT flow modifier	-

Data pre-processing

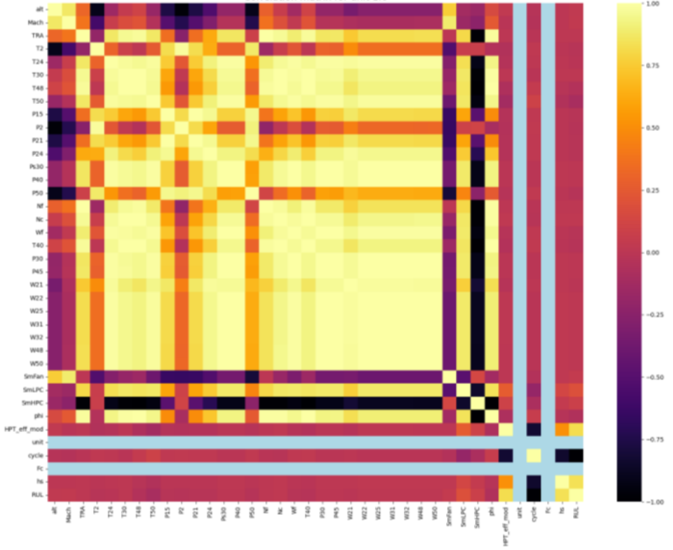
- Treat RUL as a regressive problem by predicting RUL (target variable) using all other features such as sensor descriptors, measurements, virtual sensors, model health parameters and auxiliary data
- Analysis is performed primarily using Python libraries such as Pandas, Numpy, Matplotlib and Scikit-Learn

The following actions are applied on the dataset after exploratory data analysis:

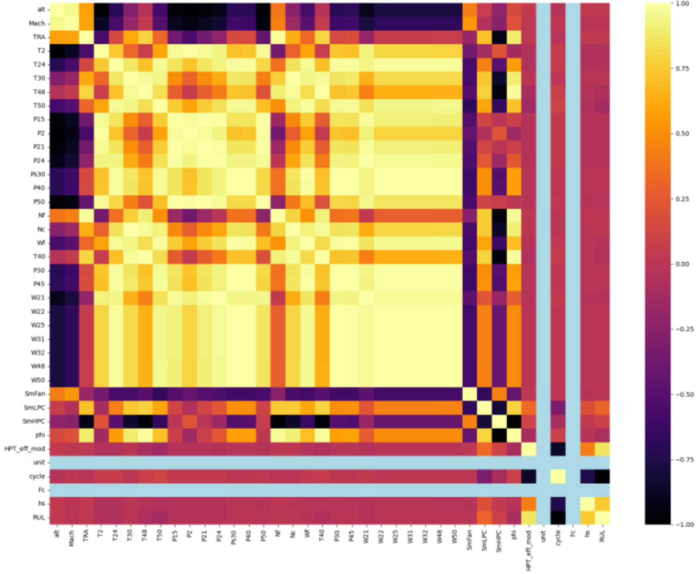
#	Observation	Action
1	No null values in dataset	Do nothing
2	Variables *fan_eff_mod, fan_flow_mod, LPC_eff_mod, LPC_flow_mod, HPC_eff_mod, HPC_flow_mod, HPT_flow_mod, LPT_eff_mod, LPT_flow_mod contains all zeroes*	Remove these variables from dataset
3	Variable *cycle* is used to derive *RUL. Strong negative correlation is observed in correlation matrix (see image in the next slide)*	Remove *cycle* and retain *RUL as target variable for prediction*
4	Variable *unit* is a categorical ID variable. Similar *unit* values do not exist in both development and test dataset.	Remove *unit* variable
5	Variables *Fc* and *hs are categorical variables while the remaining variables are continuous*	Do nothing. To be handled in data pipeline during training
6	Dataset possess temporal properties	Engineer additional lag features using target variable *RUL* (i.e. *RUL_lag1*, *RUL_lag3*, *RUL_lag5*)

Data pre-processing

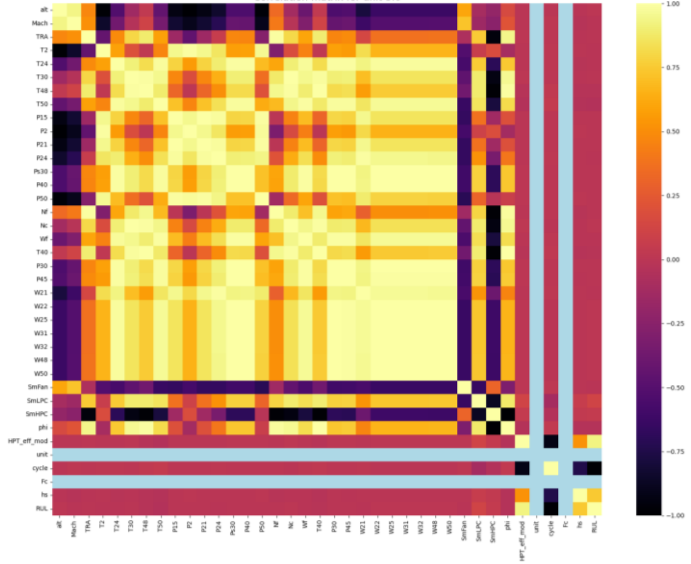
Correlation matrix for unit 1.0



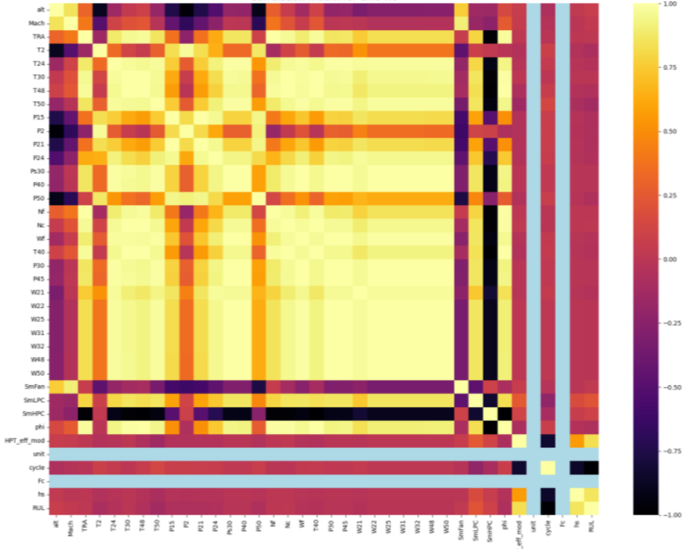
Correlation matrix for unit 2.0



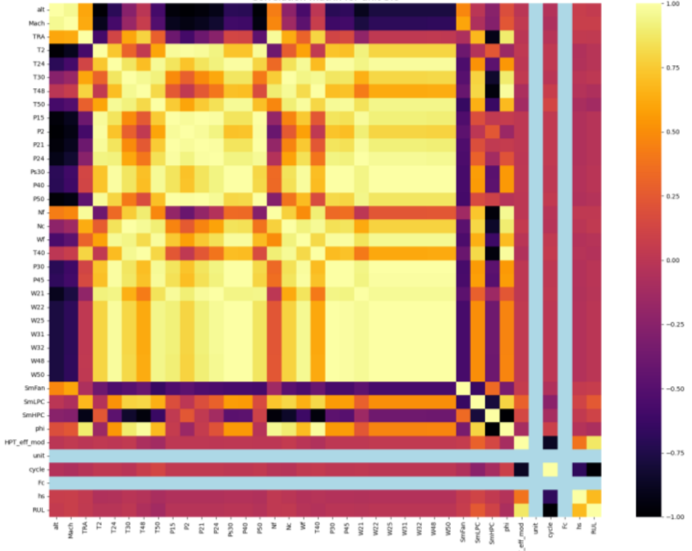
Correlation matrix for unit 3.0



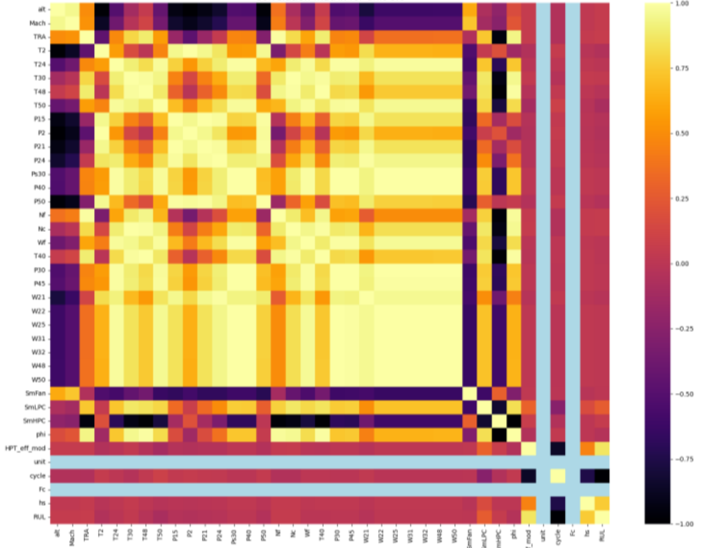
Correlation matrix for unit 4.0



Correlation matrix for unit 5.0

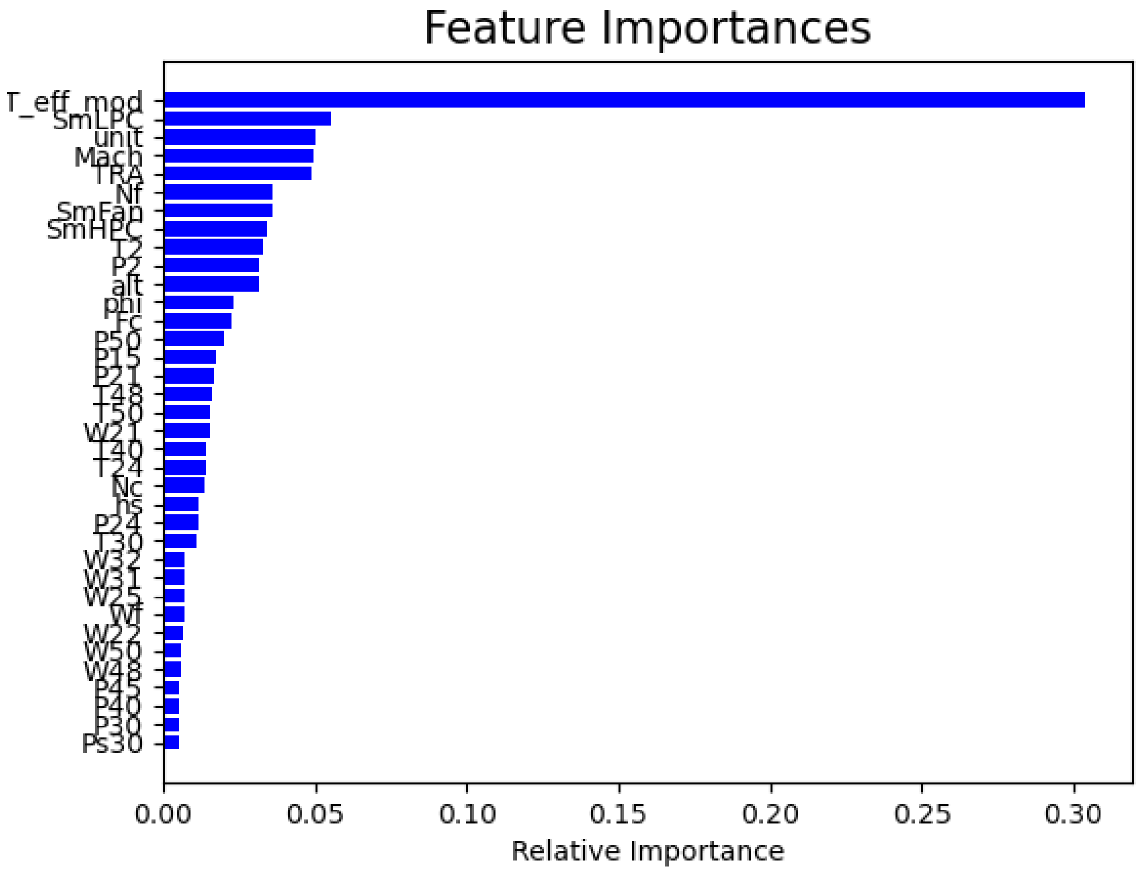


Correlation matrix for unit 6.0



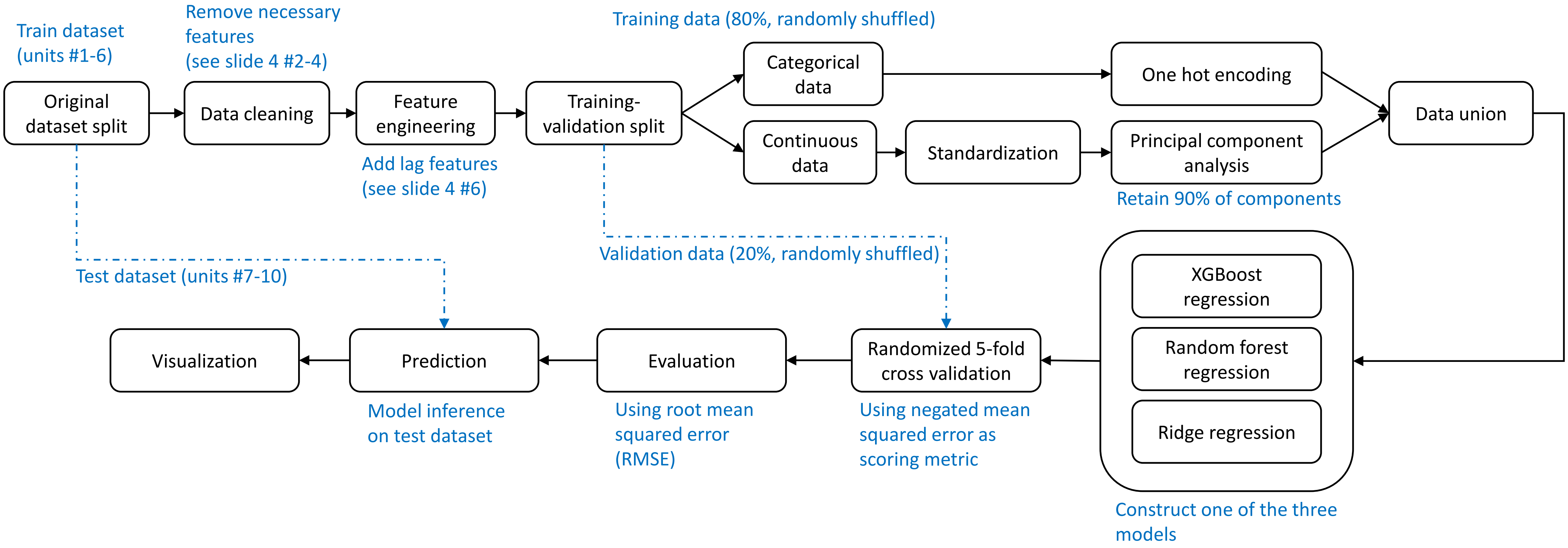
- Relatively similar correlation is observed across all 6 units in the development dataset
- *RUL* is negatively correlated with cycle which is removed during model training
- *HPT_eff_mod* and *hs* are highly and positively correlated with *RUL*, hence must be retained for model training

Data pre-processing



- Evaluation on the importance of features on *RUL* is computed using Random Forests algorithm
- *HPT_eff_mod* is shown to have the largest effect on *RUL*
- Large number of variables has little effect on *RUL*
- Principal component analysis can be included in data pipeline to create new uncorrelated variables that successively maximizes variance

Model training


Train dataset (units #1-6)
Original dataset split
Remove necessary features (see slide 4 #2-4)
Data cleaning
Feature engineering
Add lag features (see slide 4 #6)
Training-validation split
Training data (80%, randomly shuffled)
Categorical data
Continuous data
Standardization
One hot encoding
Principal component analysis
Retain 90% of components
Data union
XGBoost regression
Random forest regression
Ridge regression
Construct one of the three models
Validation data (20%, randomly shuffled)
Test dataset (units #7-10)
Randomized 5-fold cross validation
Using negated mean squared error as scoring metric
Evaluation
Using root mean squared error (RMSE)
Prediction
Model inference on test dataset
Visualization

- The data pipeline is illustrated as shown in the diagram above beginning with original dataset train/test splitting and ending with data visualization
- Three models namely XGBoost regressor, Random Forest regressor and Ridge regressor, are constructed and evaluated individually
- Mean squared error is used as an accuracy-based metric to aggregate errors in RUL estimation

Model training

Models	Hyperparameters	Range

XGBoost regressor	n_estimators
	max_depth
	Subsample
	colsample_bytree
		[100]
		randint(1, 2)
		uniform(0.25, 0.75)
		uniform(0.25, 0.75)

Random forest regressor	n_estimators
	min_samples_leaf
	max_features
	max_depth
	min_samples_split
		randint(1e1, 1e2)
		randint(1e0, 2e0)
		['auto’]
		randint(1e0, 4e0)
		randint(2e0, 4e0)

Ridge regressor	alpha	uniform(0.1, 10.0)

- Regression models are randomized 5-fold cross-validated at 3 iterations
- Hyperparameters are tuned within the distribution range as shown in the table above
- Note that distribution range and number of iterations are selected heuristically in consideration of computational time

Results

Model validation after training

Models	**Root mean squared error**		**Computation time (secs)**	
	Without lag features	With lag features	Without lag features	With lag features
XGBoost regressor	0.7770	0.9917	685.1646	309.2234
Random forest regressor	0.7502	0.9889	1041.7169	1029.1359
Ridge regressor	0.7488	0.9918	71.7706	68.2248

Model inference on test data

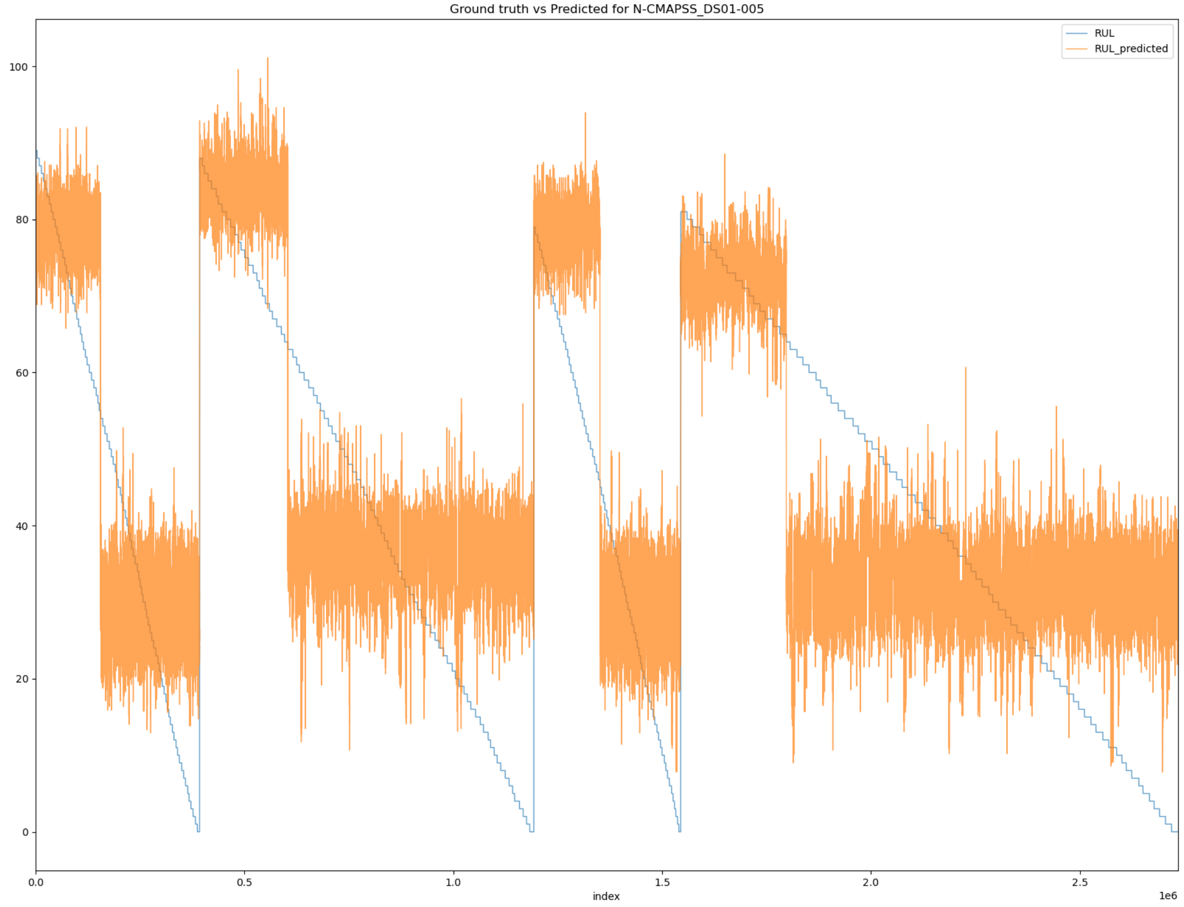
Models	**Root mean squared error**	
	Without lag features	With lag features
XGBoost regressor	16.8314	3.6441
Random forest regressor	16.9422	3.7512
Ridge regressor	16.5576	3.3004

- Three models achieve relatively similar performance based on RMSE scoring
- Addition of lag features significantly improve prediction accuracy for all three models (see diagrams in slide 11-13)
- In terms of RMSE and computation time cost, ridge regression appears to offer the best performance within the current experimental boundary

Results

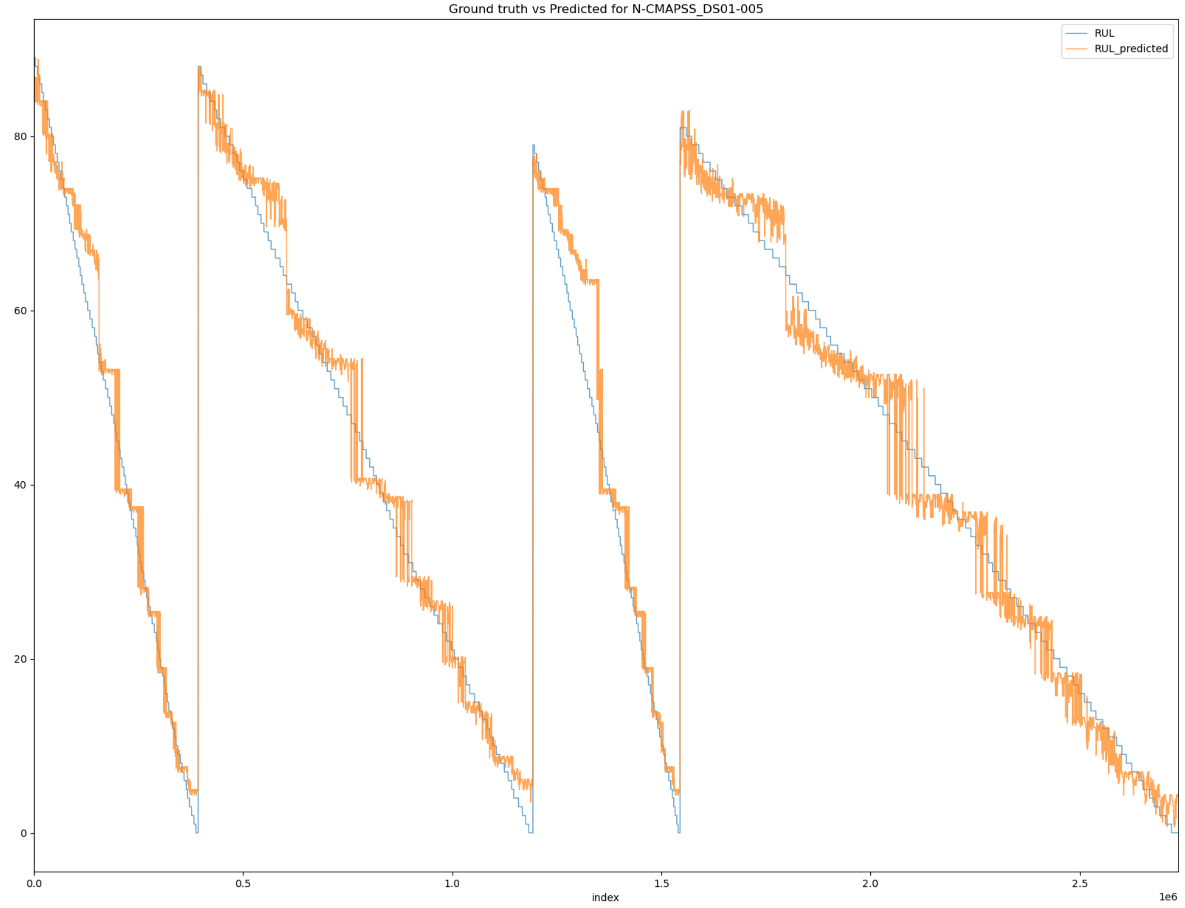
XGBoost regression

Without lag features



RMSE: 16.8314

With lag features

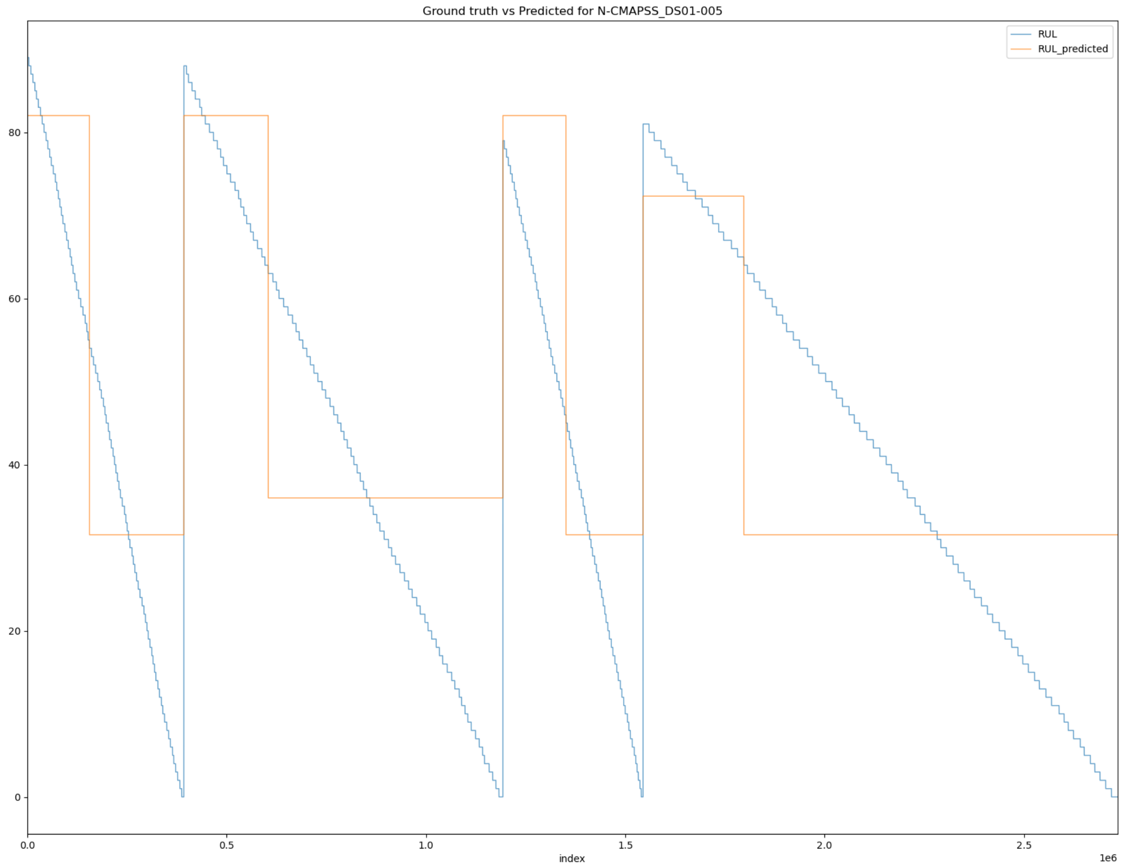


RMSE: 3.6441

Results

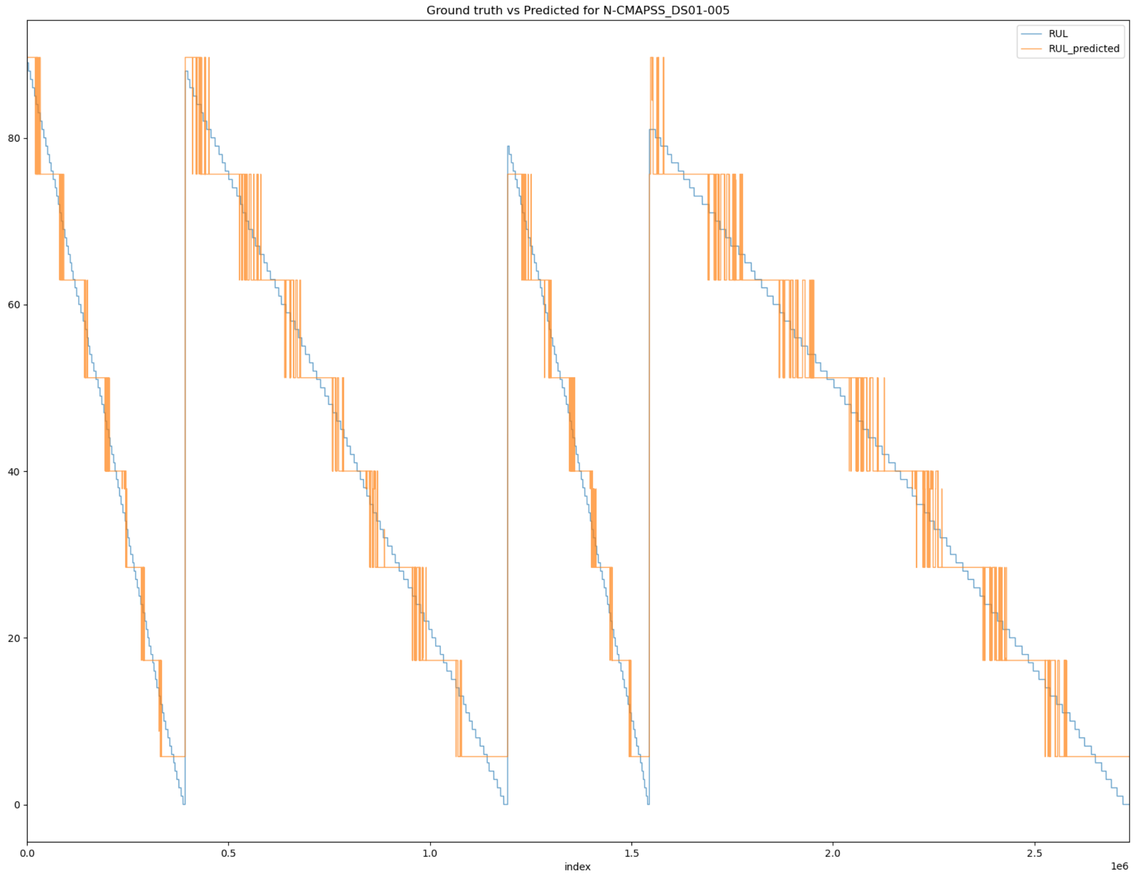
Random forest regression

Without lag features



RMSE: 16.9422

With lag features

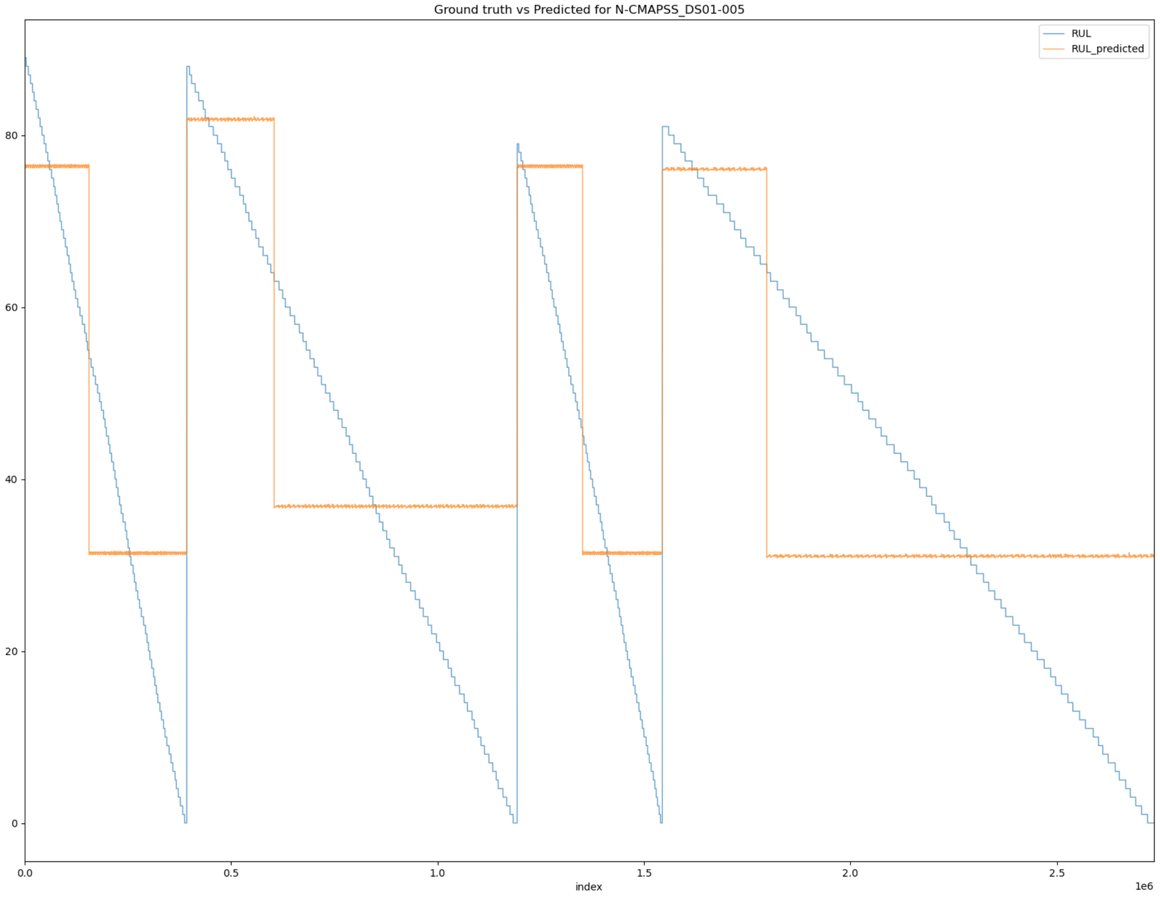


RMSE: 3.7512

Results

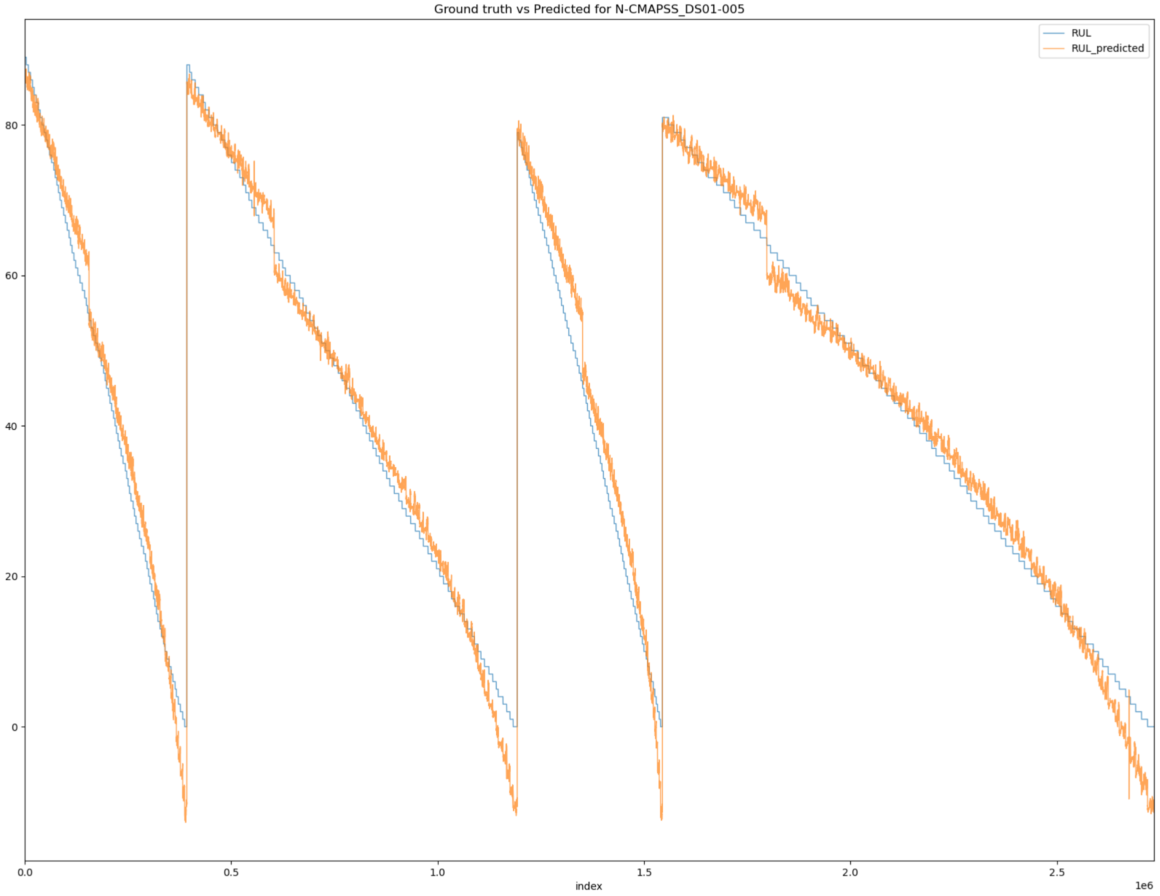
Ridge regression

Without lag features



RMSE: 16.5576

With lag features



RMSE: 3.3004

Results

Ridge regression (with lag features) on test dataset

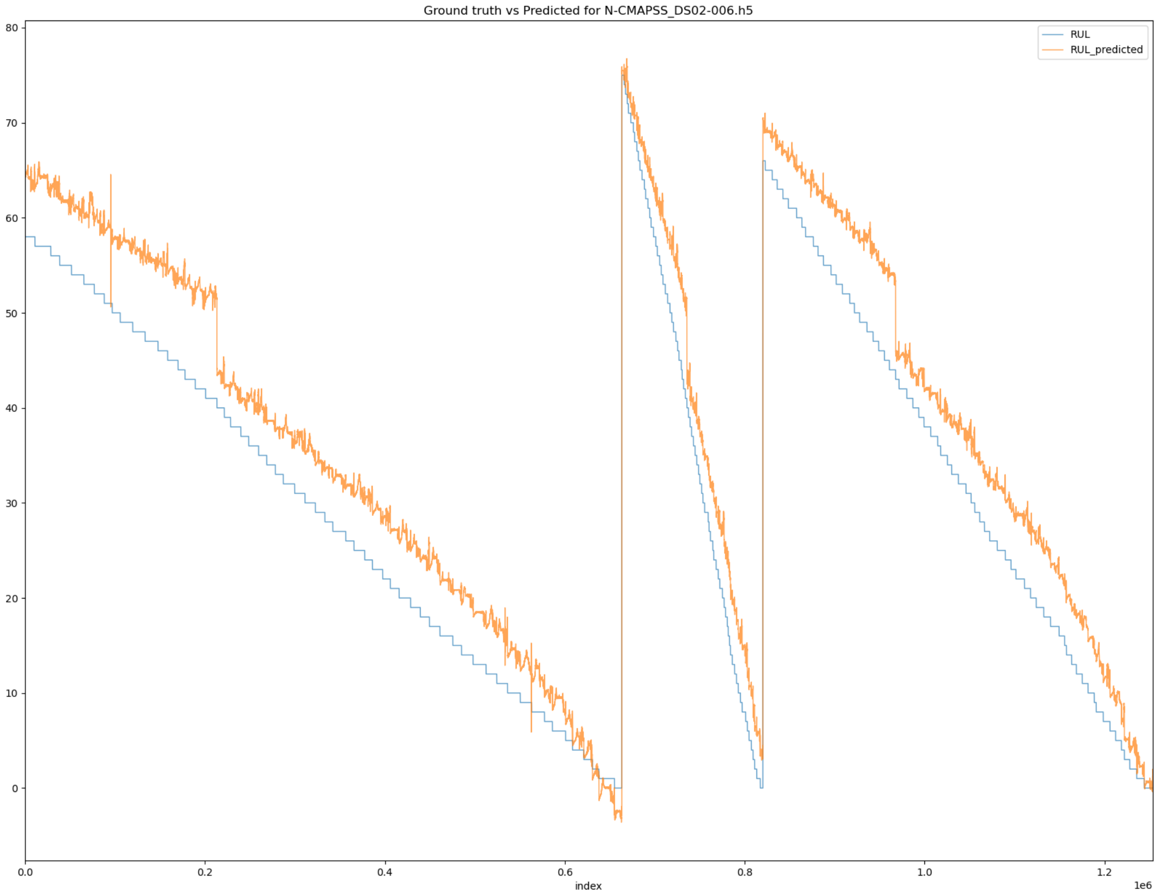
Data files	**Root mean squared error**
N-CAMPSS_DS01-005.h5 (Model is trained using development dataset from this file)	3.3004
N-CAMPSS_DS02-006.h5	6.1840
N-CAMPSS_DS03-012.h5	5.6094
N-CAMPSS_DS04.h5	8.7607
N-CAMPSS_DS05.h5	8.9196
N-CAMPSS_DS06.h5	10.9219
N-CAMPSS_DS07.h5	8.2998

- Ridge regression trained with lag features is used to infer test dataset from the remaining data files
- RMSE results are tabulated as shown in the table above and the diagrams are shown in slide 15-17
- Results seem to suggest that the model trained using a dataset (N-CAMPSS_DS01-005.h5) is able to generalize well on other test data involving different units and failure modes

Results

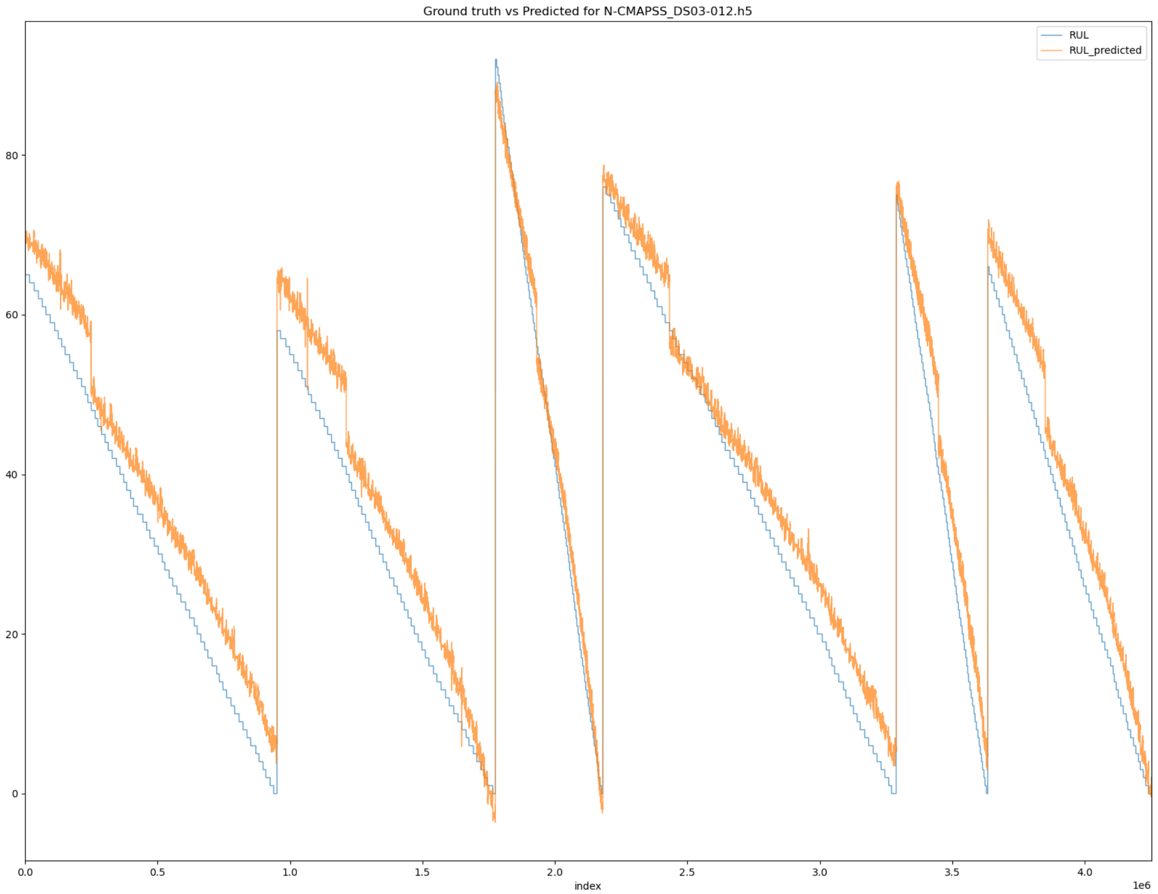
Ridge regression model (with lag features) trained on N-CAMPSS_DS01-005.h5 performing inference on test dataset from:

N-CAMPSS_DS02-006.h5



RMSE: 6.1840

N-CAMPSS_DS03-012.h5

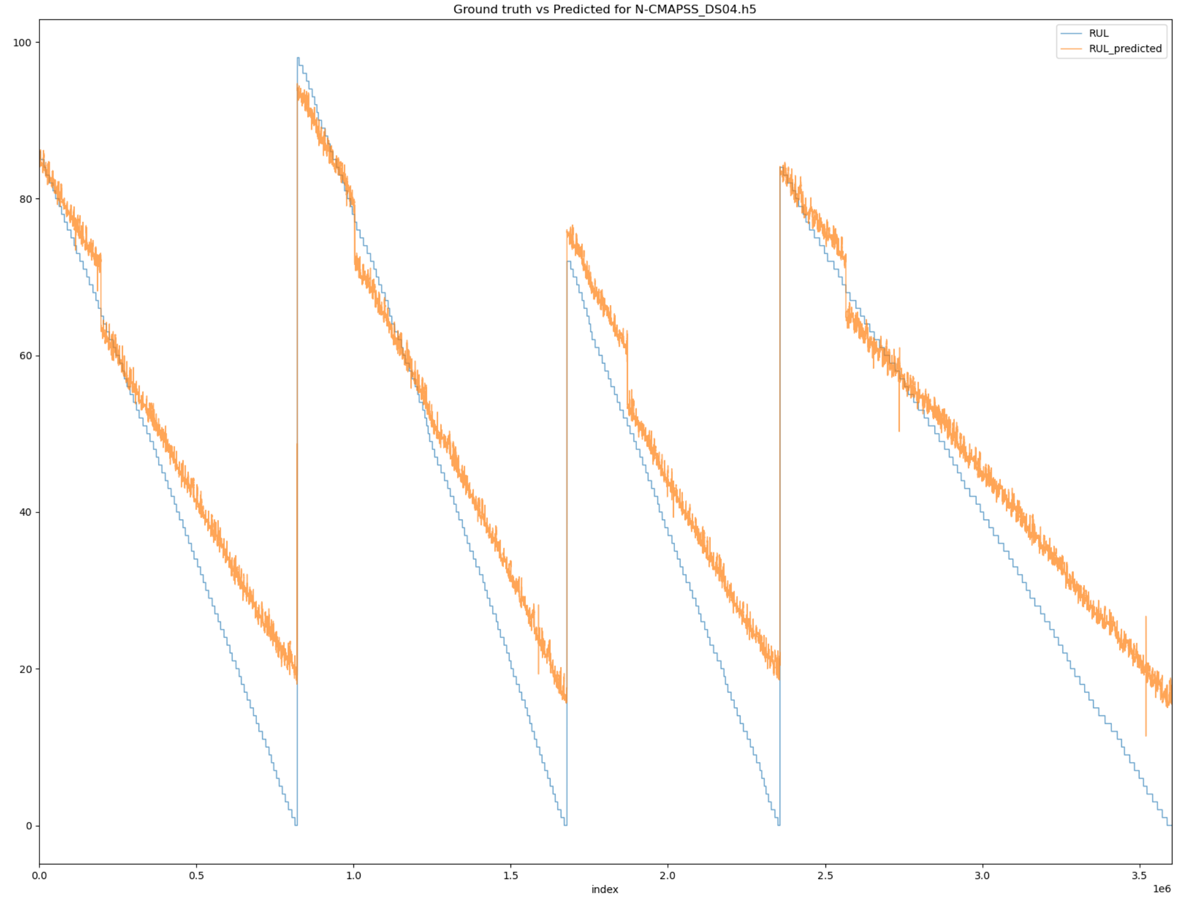


RMSE: 5.6094

Results

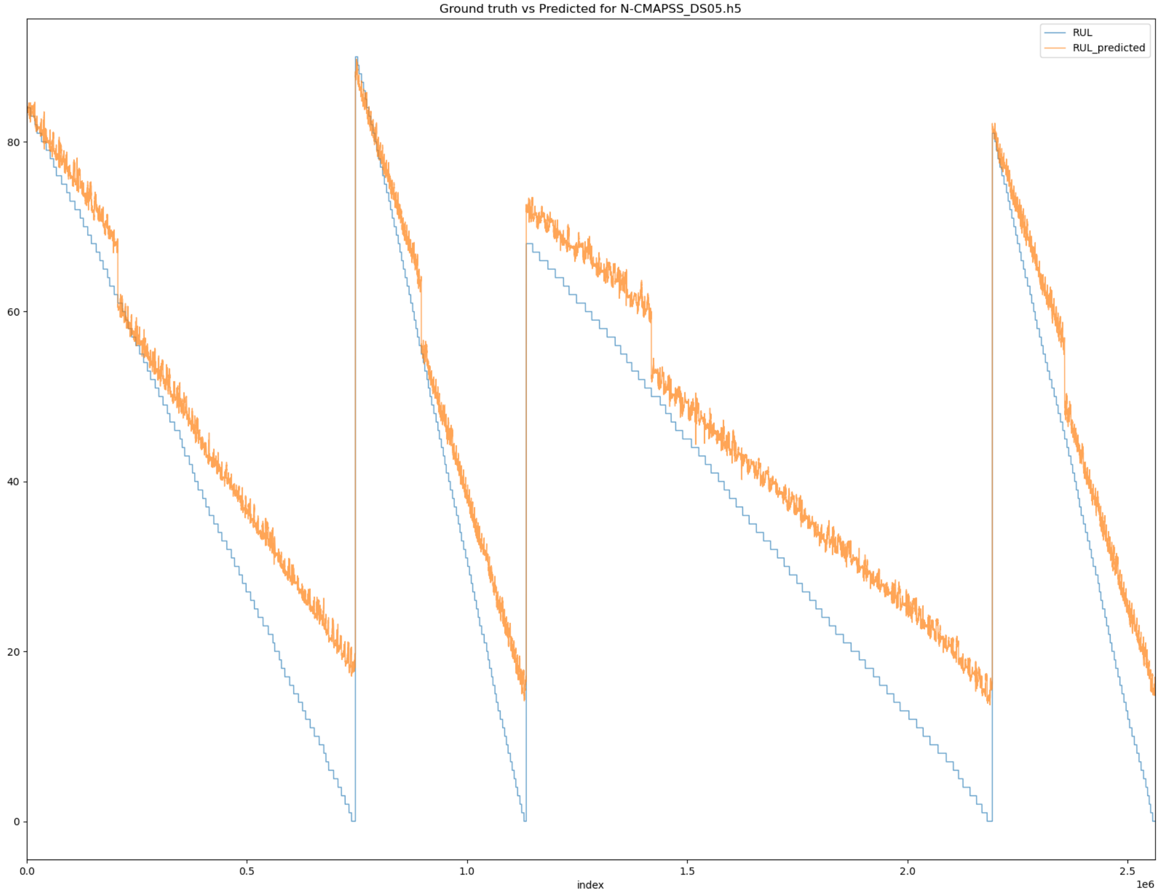
Ridge regression model (with lag features) trained on N-CAMPSS_DS01-005.h5 performing inference on test dataset from:

N-CAMPSS_DS04.h5



RMSE: 8.7607

N-CAMPSS_DS05.h5

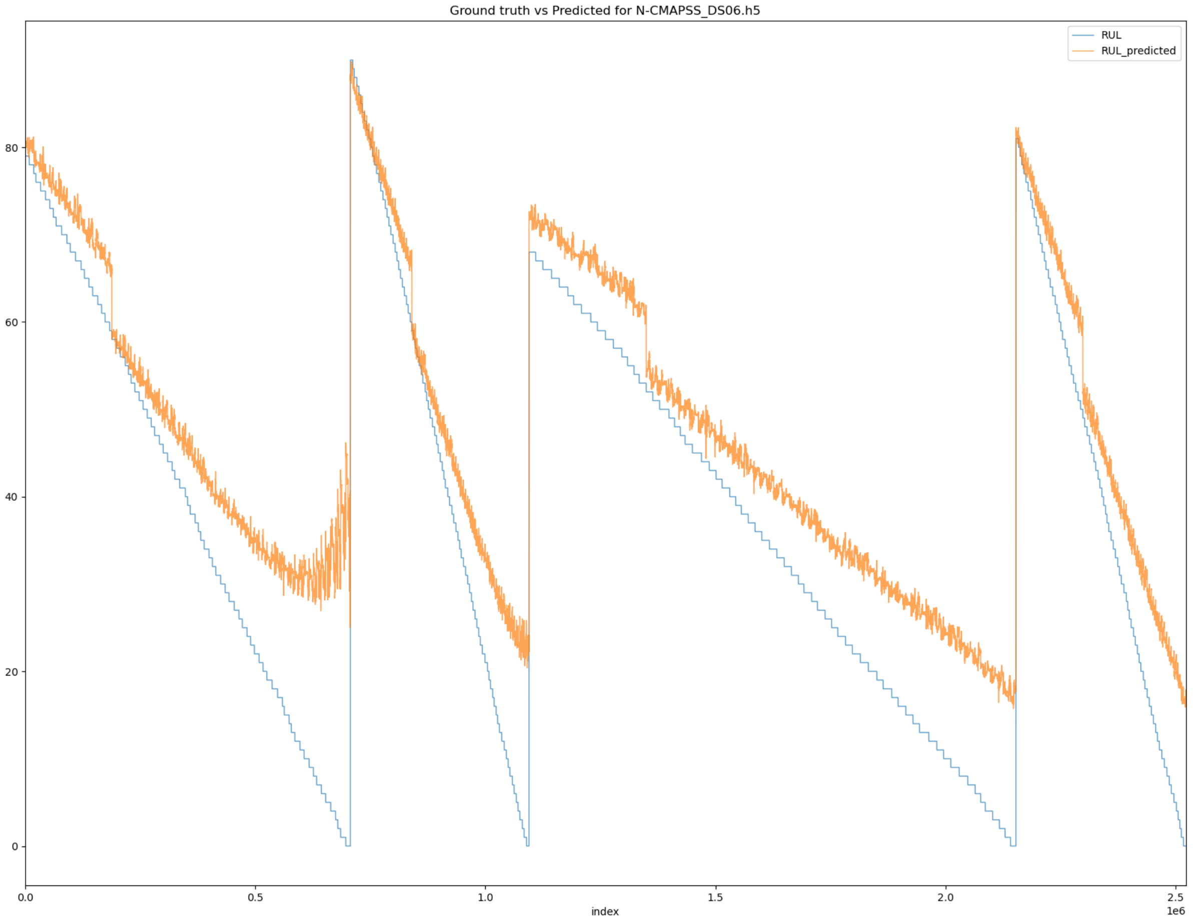


RMSE: 8.9196

Results

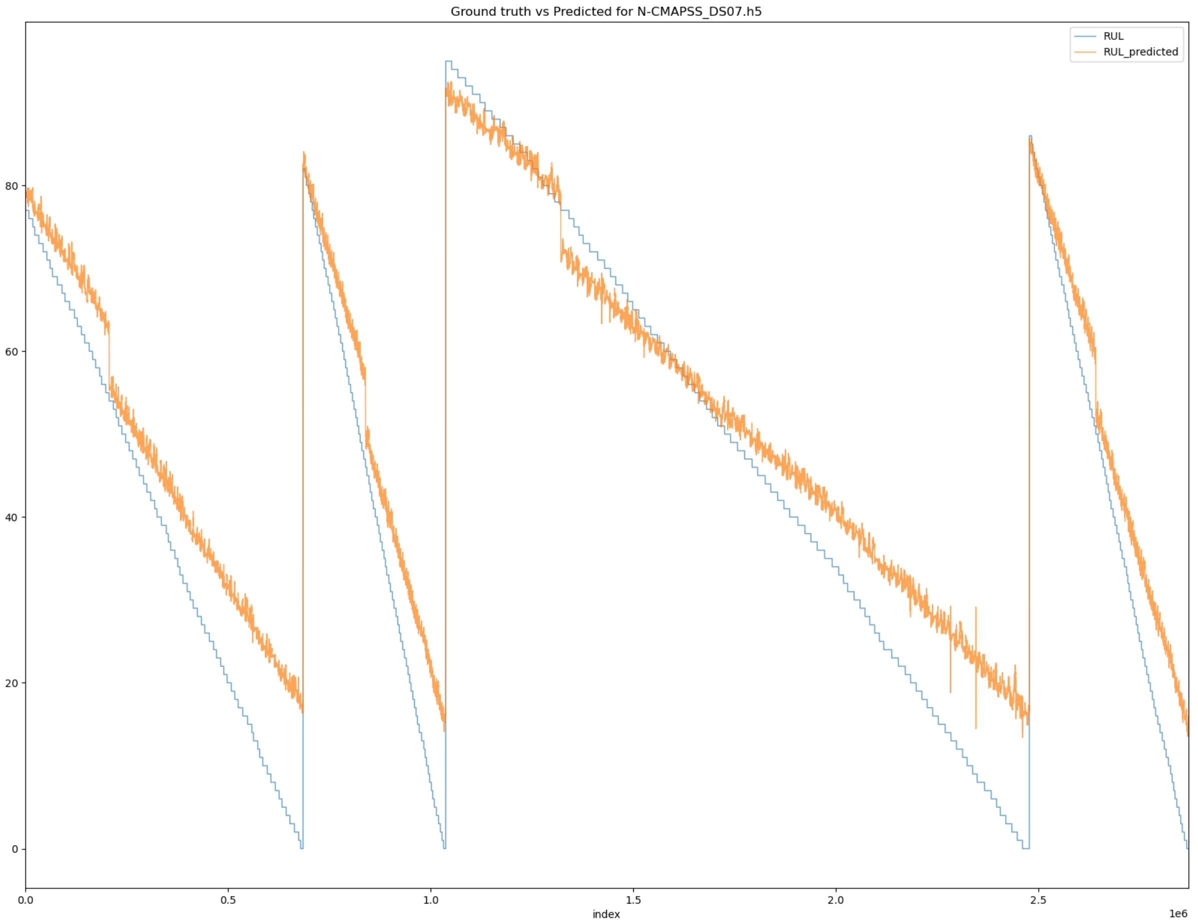
Ridge regression model (with lag features) trained on N-CAMPSS_DS01-005.h5 performing inference on test dataset from:

N-CAMPSS_DS06.h5



RMSE: 10.9219

N-CAMPSS_DS07.h5



RMSE: 8.2998

Potential exploration

- Include more datasets (e.g. N-CAMPSS_DS02-006.h5, N-CAMPSS_DS03-012.h5, etc) for modelling to achieve a model that incorporates various failure modes

- Explore data pre-processing techniques for noise filtering such as Kalman filtering and gaussian kernel smoothing

- RUL prediction is modelled based on a similarity model which requires data degradation from healthy state to failure (run-to-failure). Modelling based on survival model (only data from similar machines during failure exist) and degradation model (when a threshold of a condition indicates failure) can be explored

- Consider modelling using convolutional neural network (CNN) + long short term memory network (LSTM) to leverage upon the spatial-temporal properties of CMAPSS dataset