ADVANCED DATA ANALYSIS AND MACHINE LEARNING

Group project, Practical assignment

Lappeenranta–Lahti University of Technology LUT

BM20A6100 Data Summary and Visualisation, NASA Turbofan Jet Engine

September 15, 2024

Subin Khatiwada, Joonas Pitkäniemi, Jawed Tariq

1. Data selection and communication

1.1. Data selected

As we are a B-level group we needed to only work on a single NASA Turbofan. We selected what we viewed as the “first” Turbofan FD001. The data was split into a few different files. The data will be discussed more later.

1.2. Communication Setup and code sharing

Communication will be done via Microsoft Teams with weekly meeting to discuss observations and distribute work properly between teammates. We are going to use GitHub for code sharing. This will allow us to work together fluidly and also have a remote storage for code recovery in case of equipment faults. Matlab was chosen as the working environment for data analysis and data visualization.

Application	**Usage**
Matlab	Data analysis and visualization
Github	Code sharing
Microsoft Teams	Communication

Table 1 Tools used for the project

2. Data summarization and experimental scenario overview

2.1. **Engine data and Fault development**:

The dataset consists of multivariate time series from a fleet of turbofan engines split into training and test subsets capturing the engine's performance over time. As we are working

in a Errors start to appear in the training data and get worse until the system fails. Hence, the RUL must be predicted before the system failure.

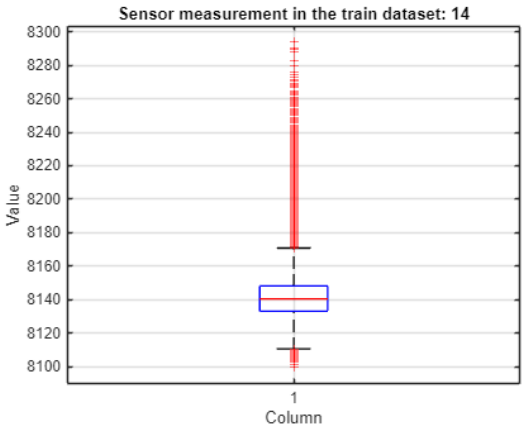
2.2. **Operational Parameters**

The performance of the engine is greatly impacted by three operational parameters. The data contains setting values (variables 3, 4 and 5 in the dataset) that have no clear indicator what they mean. There is sensor noise in the data which must be managed during analysis. The data is also split into different unit values which go through the time in cycles parameter values that can be viewed as time series data

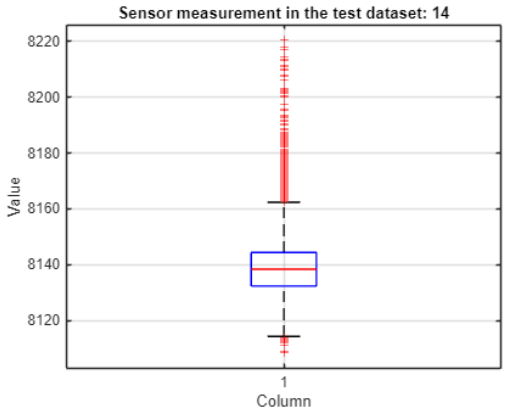
3. Data visualizations

3.1. Visualizations

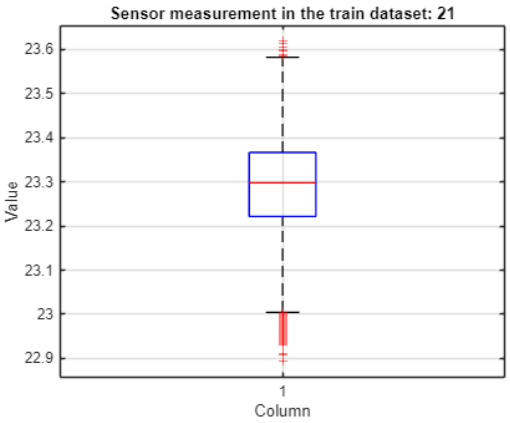
The data consists of 26 columns which have very differing measurement values, that are difficult to fit in a single plot. Some variables also don't change over the time the motor runs. Visualized below will be a few interesting occurrences and similarities between the split datasets of train and test datasets. Pictures 1, 2, and 3 show some interesting sensor measurement values with massive outliers on both sides of the regular occurrences in the data from the selected motors test data set. Visualized next to each other are the same sensors from both the test and train datasets



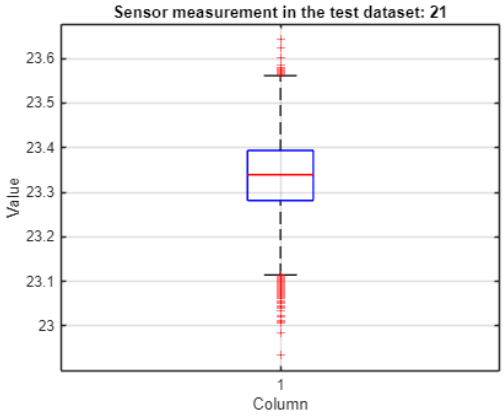
Picture 1



Picture 2

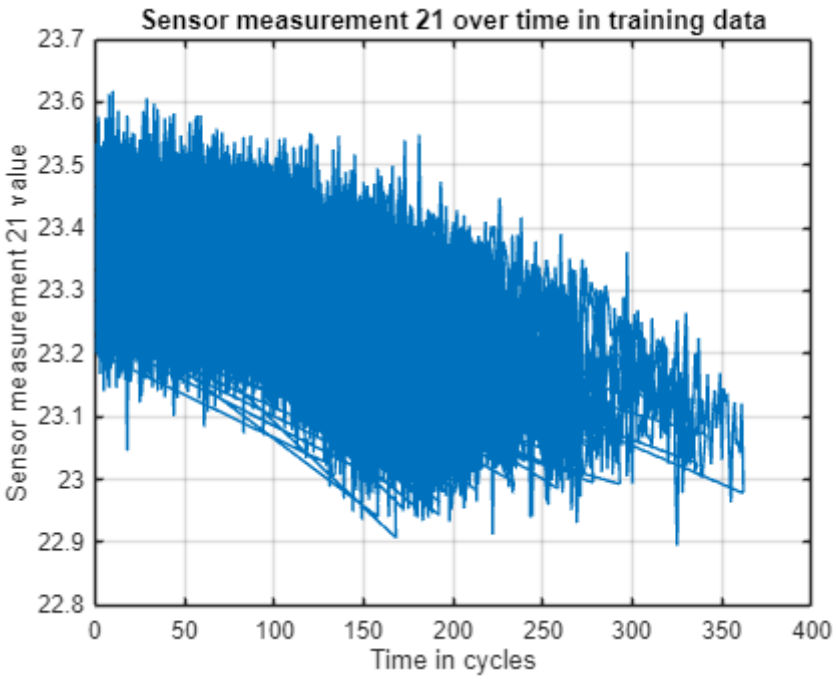


Picture 3

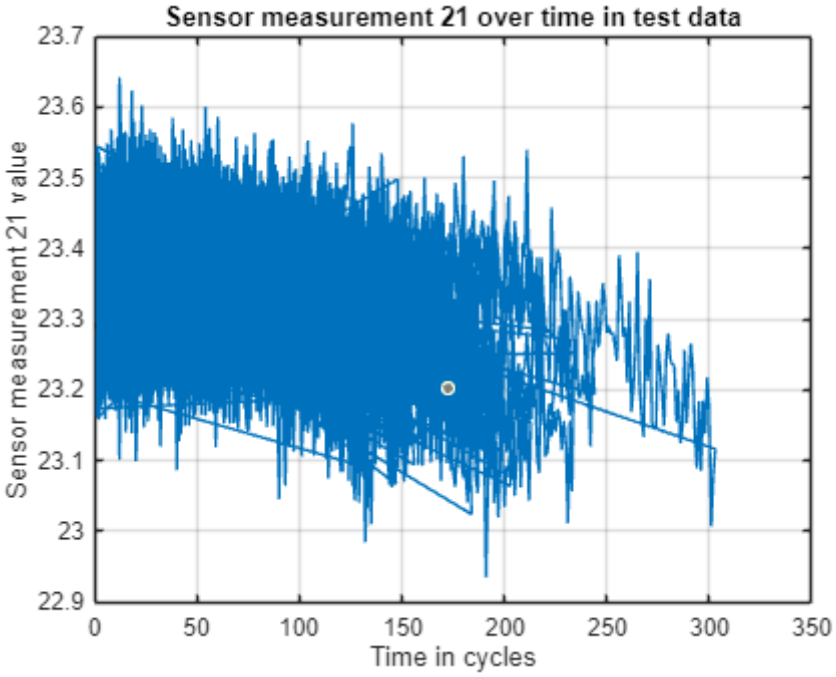


Picture 4

The boxplots help us detect a few initial points of interest in the data. Digging a bit deeper one can visualize these sensor measurements over the cycle times. Focusing on the sensor measurement number 21 (Variable 26 in the datasets) shows similar patterns. The plots below show that despite looking at all units at the same time a decrease in measurement values can be seen around the 100 to 150 marks of the time in cycles variable in both the train and test dataset (Pictures 5, 6).



Picture 5



Picture 6

3.2. Statistics and basics of the data

The test and train datasets are built similarly with the RUL-file being a single column vector.

Dataset	**Rows**	**Columns**
FD001 Train	26	20631
FD001 Test	26	13096
FD 001 RUL	1	429
Total Train + Test	26	33727

Table 2 data summaries

The data seems to be split in a 40/60 split where 40% of what could be viewed as the test dataset and 60% of the data is in the training dataset.

4. Challenges and Preprocessing Plan

4.1. **Challenges**:

The data overall isn't very well documented. The sensor measurement data doesn't really mean anything to a viewer despite it being floating point numbers or integers. The sensors and their functionality aren't explained at all.

As stated previously the data provided is split into different units that could potentially be analysed separately, or they might need to be analysed all together, there's no explanation on what units mean either.

The RUL file is also a unconnected singular vector that doesn't really connect to the main data in a consistent manner.

The largest challenge is the lack of full understanding of what one is looking at. Misunderstanding how to look at the dataset can lead to misanalysis and misguided machine learning development.

4.2. **Preprocessing Plan**:

To address these issues, the preprocessing plan involves imputing or removing missing values, detecting and managing outliers, and scaling the sensor data for consistency. The data is already split into a 40/60 split that we are ready to potentially alter if needed for a better split. Splitting the data in this manner should help us provide our machine learning models we create during the course in a reliable manner.

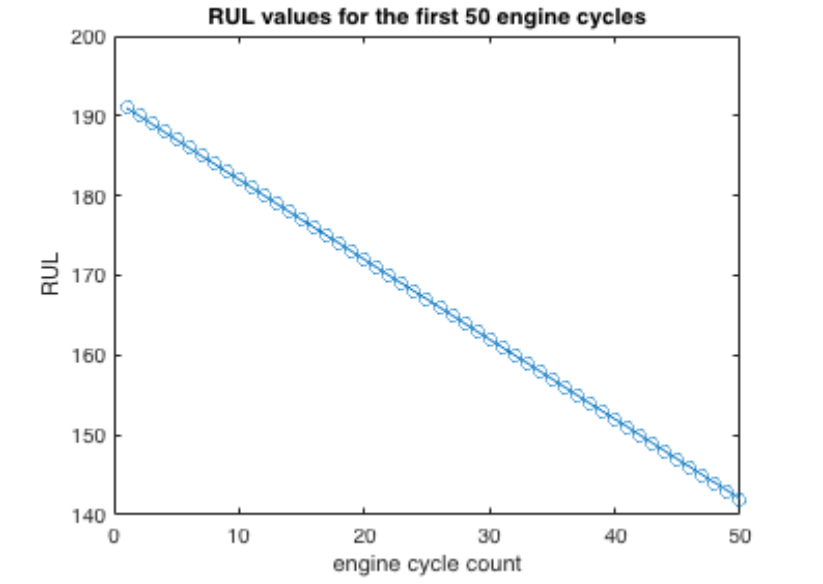
5. Remaining Useful Life (RUL) Calculation and Analysis

Using the training dataset, we determine the Remaining Useful Life (RUL) for every engine unit. The dataset is made up of several sensor measurements taken for various engine units during various running cycles and our task is to predict the RUL which is the number of remaining cycles before engine fails, for each unit.

For each Engine the RUL was calculated as:

$RUL(i) = max\_cycle(i) - current\_cycle(i)$

This approach was applied to the first 50 engine cycles for visualization purposes, as seen from the plot below.



This RUL for the first 50 engine cycles starts at about 190 and drops linearly to 140, showing a continuous decline toward failure.

The idea that each engine unit is continuously operating toward failure is supported by the linear decline in RUL for these cycles. This steady pattern will help with engine failure prediction models.

6. Train-Test splitting

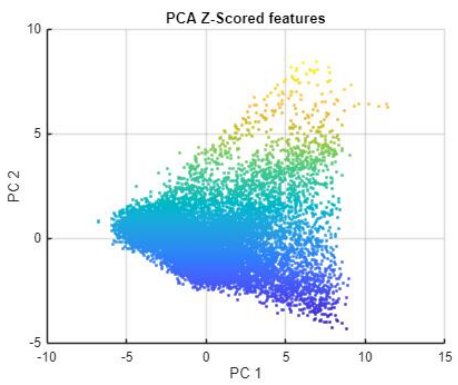
The data is a time-series dataset, so the data needs to be observed in order. Therefore, the data should be split into portions based off the unit in question. As the maximum unit value of the data was 100, the data was split into a 70/20-split. Shuffling the data would lead to the time series breaking which then would lead to the developing model making bad predictions as the time the motor is running is very relevant to the outcome.

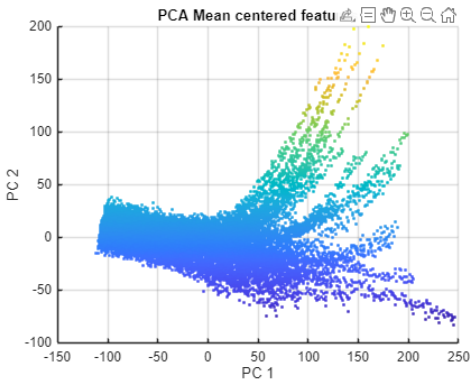
We were instructed by the TA to split the training partition of the data into calibration and test splits. This was in a bit of a conflict with the later assignment in moodle, but we chose to stick to the initial instructions and only split the data into the calibration and test splits.

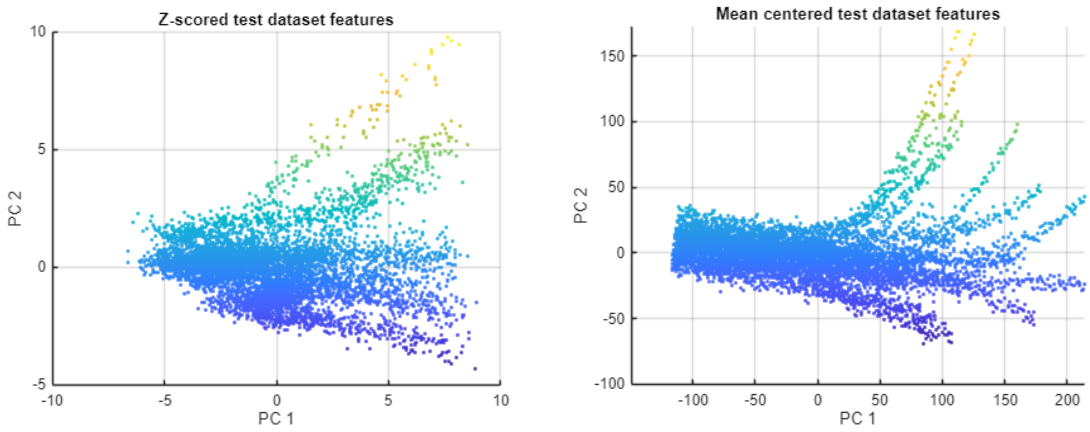
The X-variable was used to represent the features of the data nad the Y-label was selected to represent the label of the data i.e. the prediction outcome variable.

6.1. Centering and scaling the data

Two methods were used to scale and center the data just to look at differing outcomes. Z-Score and mean centering. The results would favor Z-scored values, as the data seems to be a bit more well-rounded. But in a way mean centering seems to show the necessary trends in the time-series, so that could also work here.







6.2. Outliers from the data and Re-visualize

First detect and remove outliers from a dataset using z-scores and re-visualize the cleaned data and set a z-score threshold to identify extreme values in both a mean centered test dataset and a z-scored test dataset.

For the mean-centered dataset computed the z-scores and identified the outliers using a threshold-based comparison. Similarly, calculated z-scores for the z-scored dataset and applied the same outlier detection method. After identifying the outliers cleaned both datasets by removing the rows corresponding to these outliers.

Once the data cleaned, then re-visualized both datasets. Scatter plots of the cleaned mean-centered test data and z-scored test data are generated, highlighting the distribution of data points across two principal components (PC 1 and PC 2).