Delft University of Technology

Sustainable Air Transport
AE4465

Tutorial 2 of Assignment: eXplainable Artificial Intelligence (XAI) for the Maintenance of a Turbofan Engine

Lecturer:
Marcia L. Baptista
Delivery Date:
30th of June 2024

May 11, 2024

Introduction

This is a coding tutorial (02) to support the 2024 assignment. This tutorial is meant to help develop the assignment in an easy and reproducible manner.

Goal

The overall goal of this tutorial is to do some pre-processing on the data and create a running RUL prediction model.

Data

You can find the assignment data at: https://www.nasa.gov/content/prognostics-center-of-excellence-data-set-repository (Turbofan Engine Degradation Simulation - 6)

You should download the "train_FD001.txt" dataset. This is a dataset of engines that exhibit only one fault mode and one operational condition. The other datasets, which you can also use, are more complex. The "test_FD001.txt" file and the other test files have smaller trajectories and should **not** preferably be used.

Instructions

1. We assume that you have completed tutorial 1.

2. Import additional necessary libraries **at the top** of the code file or Jupyter noteboook. It is good practise to import libraries only at the top (even though it is not strictly enforced). In our case we are going to implement the machine learning of random forest regressor so we important the suitable model from the sklearn package.

```
from sklearn.ensemble import RandomForestRegressor
# import other necessary libraries (at the top)
```

3. Normalize the data employing min-max normalization or standardization. Please check how to do it in https://stackoverflow.com/questions/26414913/normalize-columns-of-a-dataframe. Either opt for one or another. How to know which one is better?

```
# Example of min-max normalization (adapt to your dataframe and case)
normalized_df=(df-df.min())/(df.max()-df.min())
```

```
# Example of standard normalization (adapt to your dataframe and case)
normalized_df=(df-df.mean())/df.std()
```

   Check the final results by printing the column or visualizing some trajectories over time for different engines (**remember** Tutorial 01). The data will be different (in scale) according to the method you chose.

4. Split the data into training and testing datasets. Be aware that you should split the data without mixing units in training and testing data. You need 4 variables in the end:

```
# Data splitting results
y_train = ... (to be completed)
y_test = ... (to be completed)

X_train = ... (to be completed)
X_test = ... (to be completed)
```

5. Create a random forest model (SVM, NN or other) based on the splitted data. Check the potential regression models at Supervised learning at sklearn.

```
# Data splitting results
rfr = RandomForestRegressor(max_depth=3)
```

Create an object of the machine learning type you want to implement. Configure the several parameters you find necessary. Parameters vary per machine learning model chosen.

6. Proceed to do two steps with the data: first we train the model on the training data (fit) and then we estimate (predict in regression) the target RUL on the test dataset.

```
rfr.fit(X_train, y_train)
predictions = rfr.predict(X_test)
```

7. Evaluate the predictions of the model.

Extra Instructions

1. Apply filtering techniques (moving average, savgol).

2. Implement cross-validation techniques.