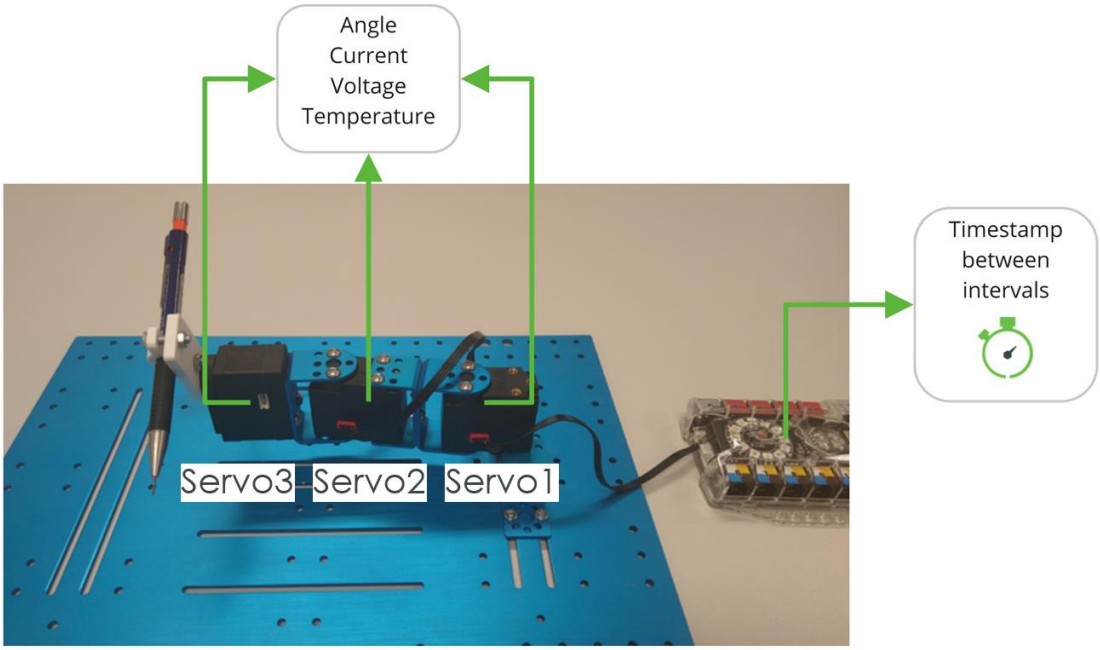
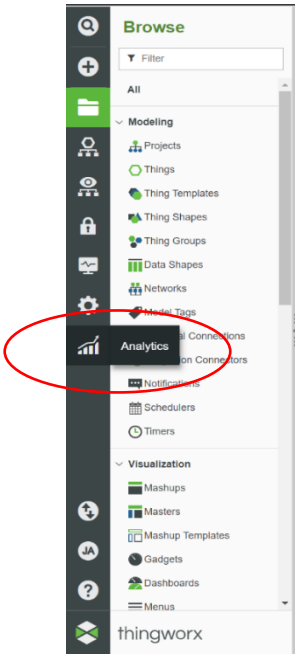
Create a predictive maintenance project using time series analytics with Thingworx Analytics

1. Demonstrator

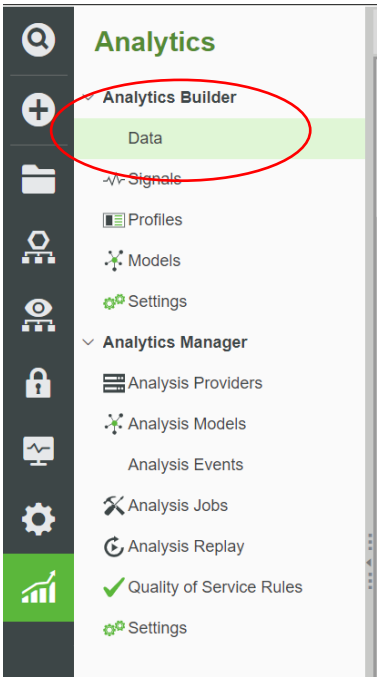


   a. Dataset: data/ThingworxAnalyticsTimeSeriesDataset.csv
2. This guide serves as an introductory exercise for a predictive maintenance use case using Thingworx Analytics. This example is intended to familiarize new users with the necessary steps for a predictive maintenance project and time series Analytics with the Analytics builder. Before you start the project please read through the guideline for a predictive maintenance project before. For more information on the demonstrator and the use case please refer to the README.md in this subdirectory.

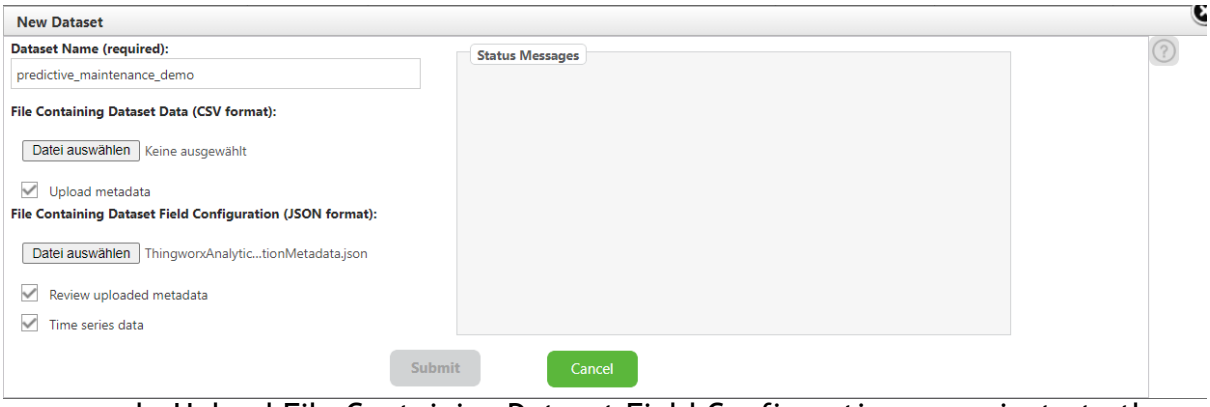
3. Navigate to Analytics on the menu bar on the left side of composer:



4. Select Data



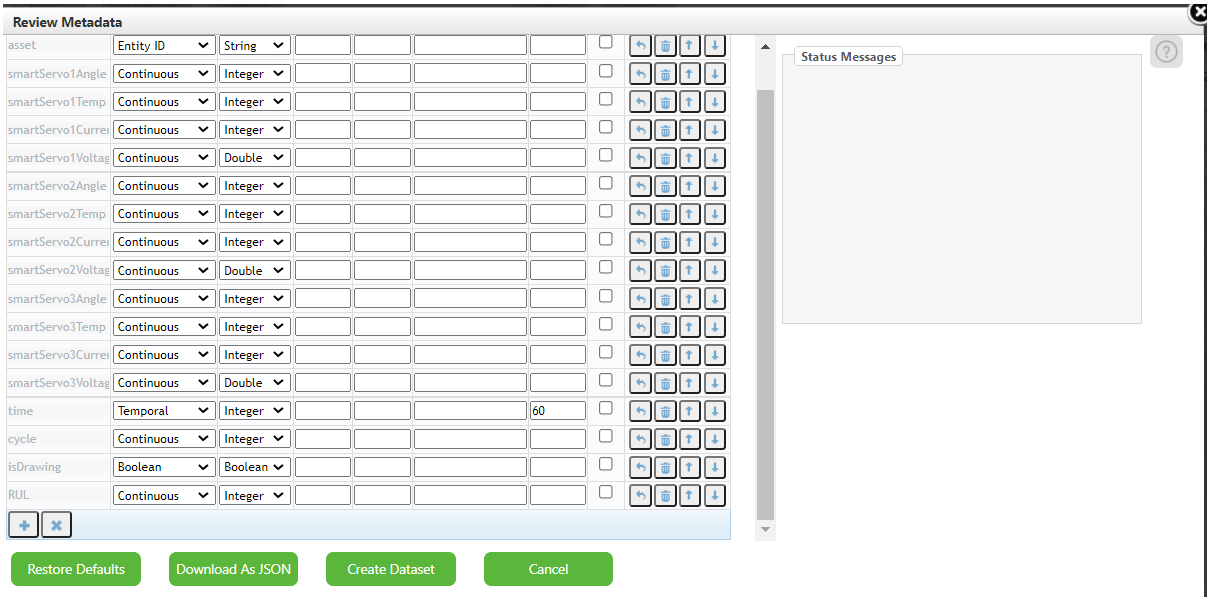
5. Create New dataset
    a. Name: predictive_maintenance_demo
    b. Choose File -> navigate to the ThingworxAnalyticsTimeSeriesDataset.csv file on your PC
    c. Check Upload metadata, Review metadata and time series data



    d. Upload File Containing Dataset Field Configuration -> navigate to the ThingworxAnalyticsTimeSeriesPredictionMetadata.json file on your PC
    e. Submit

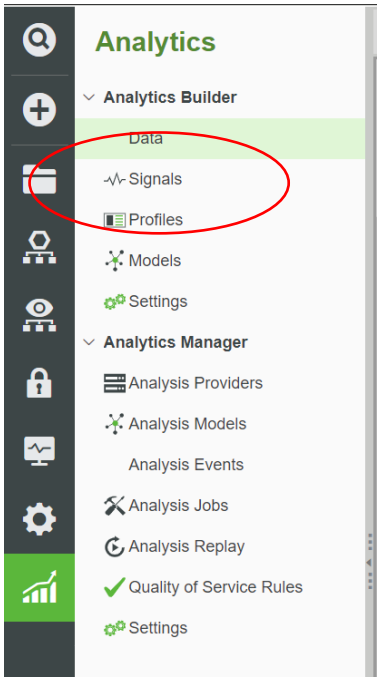
6. Check the metadata

a. It should look like this:



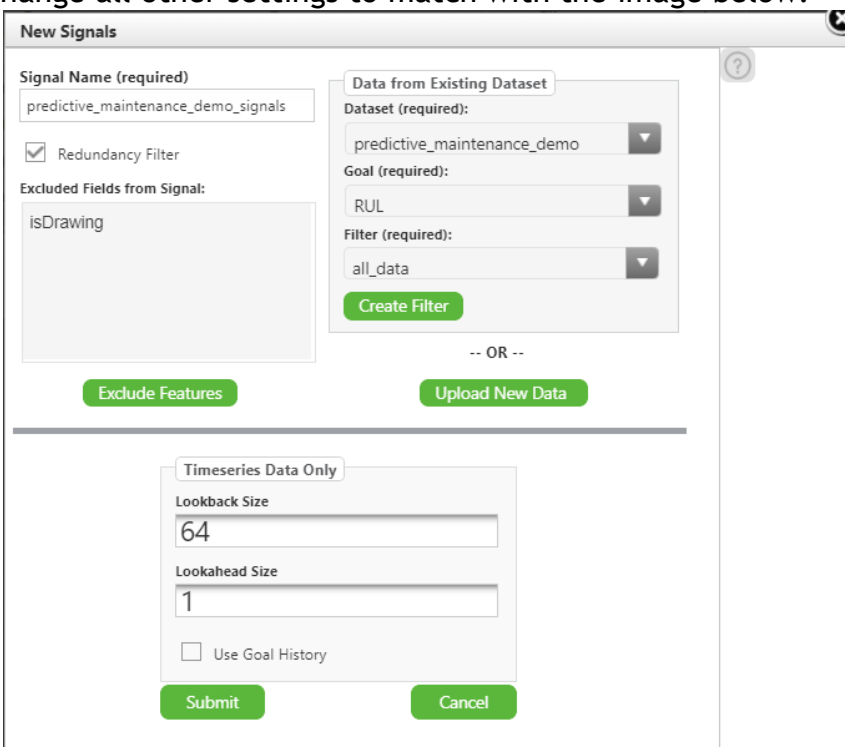
b. Click Create Dataset
c. For detailed information on data types please refer to: Review and Edit Metadata

7. Navigate to Signals
   a. Create a new signal



8. Enter Signal settings
   a. Name: predictive_maintenance_demo_signals
   b. Dataset: Select your previously created Dataset
   c. Goal: Select RUL (Abbreviation for Remaining Useful Life)
   d. Filter: all_data

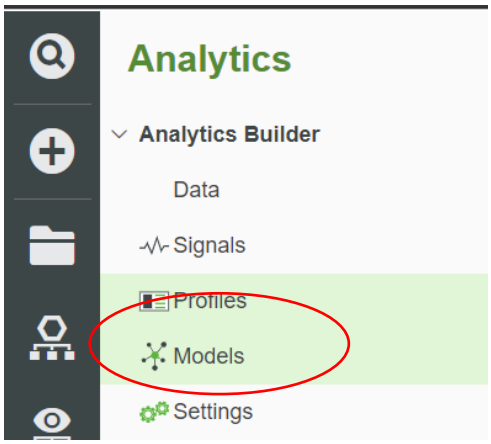
e. Exclude Features -> exclude isDrawing
   This feature was manually labeled during the data collection (see scripts/time_series_log.py) and is not available for live predictions.
f. Change all other settings to match with the image below:



g. Submit

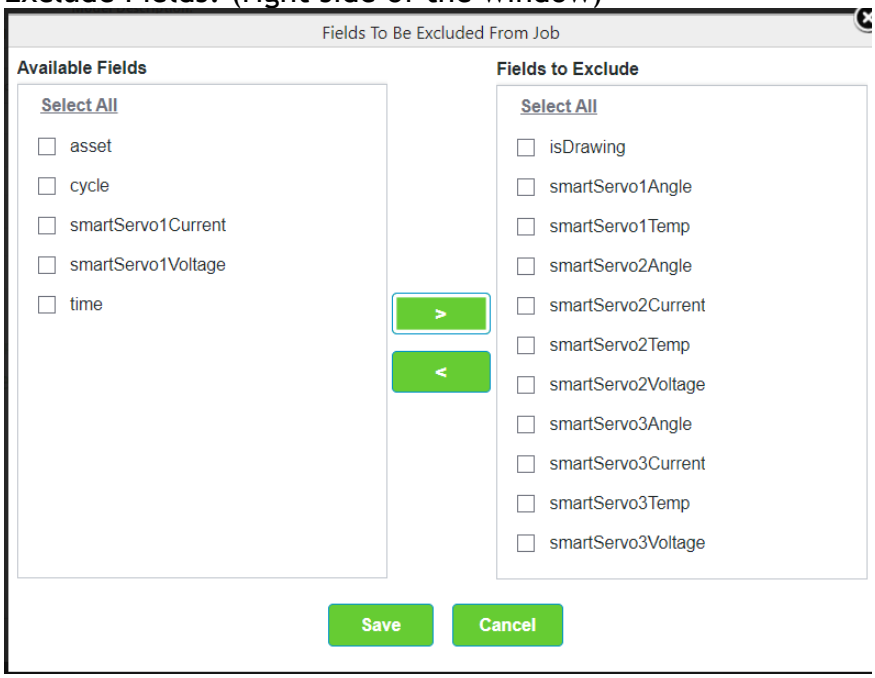
9. Review Signal results
   a. On the left side you can see the Feature Name and how much information it provides for the value of RUL (from 0 to 1), 0 means no mutual information
   b. For more information to Signals please refer to Working with Signals
   c. For more information on feature engineering please refer to Feature Engineering Community Post

10. Navigate to Models
    a. For more information please refer to Predictive Models



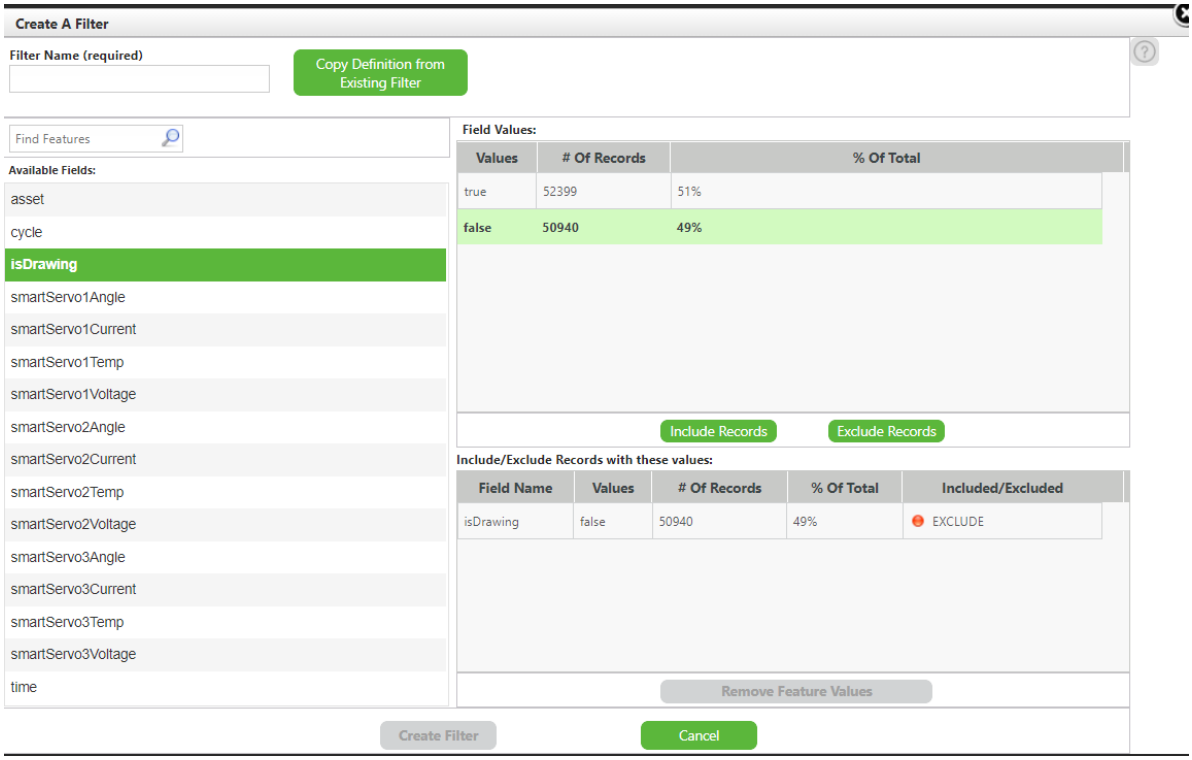
11. For this guide time series analytics is used, for information about this please see [Time Series Prediction]

12. Create new Model
    a. Name: predictive_maintenance_demo_model
    b. Dataset: name of the previously created dataset
    c. Goal: RUL
    d. Exclude Fields: (right side of the window)



Save

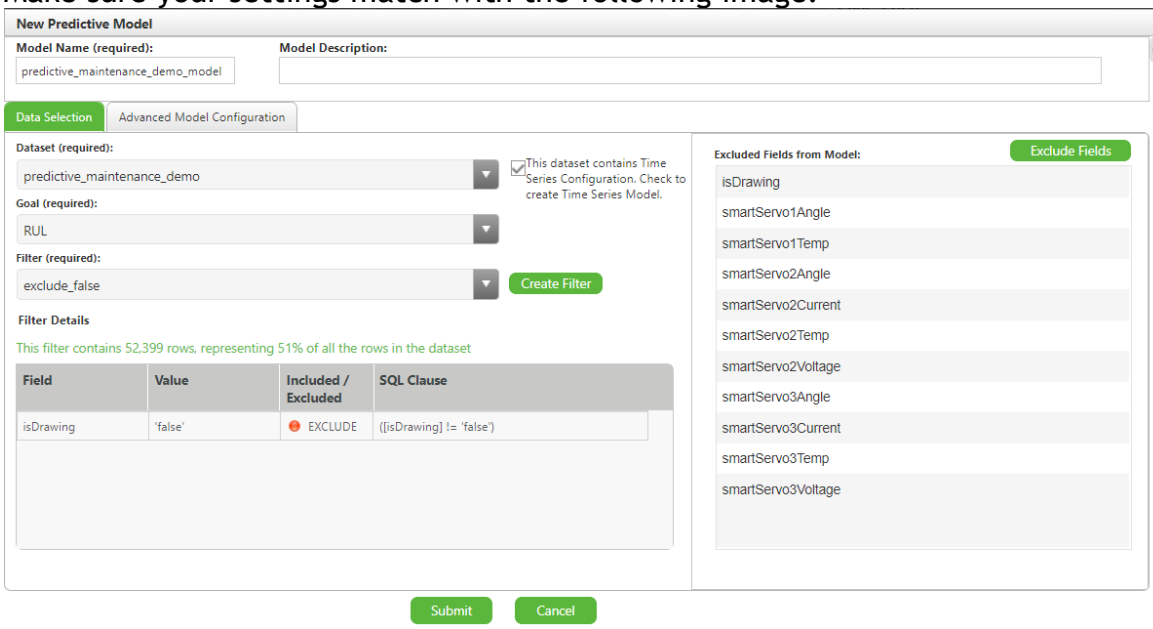
e. Exclude following values:



Those values need to be excluded in order to not have too many samples with cycle 0 in the dataset. This would bias the model output.

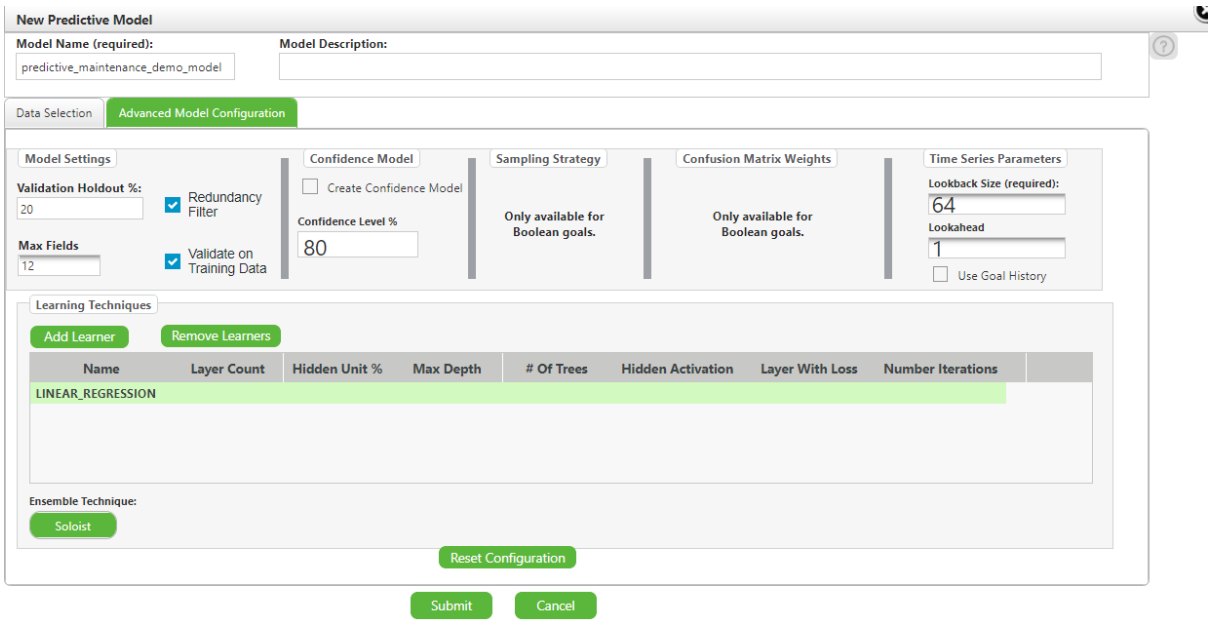
f. Enter a filter name and create the filter, after that you need to select your filter from the filter dropdown menu

g. Make sure your settings match with the following image:



h. Click on Advanced Model Configuration (refer to step 10 for more information on the settings)

i. Please enter following details:



Validation holdout: For time series analytics the last 20% (if you select 20%) of data samples in your dataset are used to test the trained model to ensure the data integrity. Non time series models use 20% random samples.
Max Fields: Limit how many features should be used for training (Thingworx Analytics creates features, so if your dataset only contains 4 features this limit still applies, because averages, derivatives, … are calculated and used as additional features)
Lookback size: 64 is the limit with Thingworx Analytics and those samples contain an entire cycle of the robot-program
Learning Techniques: Remove all learners except Linear_Regression, you can experiment with different learners later and compare them

j. Submit

13. Wait until the model state changes to Completed
14. Double click on created model
    a. Review the results for the training and test data
    b. For more information on the model results please refer to View Model Results

15. Congratulations, you have done your first time series analysis. Now you can either work on your own projects or continue experimenting with this dataset. The following points will show you how to create a new feature from your dataset, how to label the RUL and use the hardware for data collection.
    a. For more information on feature engineering please refer to Feature Engineering Community Post
    b. If you want to use real time predictions please refer to the ThingworxAnalyticsIntroduction-guide (starting at step 14) and the thingworx_analytics_scripts in this git.

16. The current version of the scripts are customized for following workflow:
    a. Collect data from the hardware
       One run includes drawing with the robot and approximately the same amount of cycles after the pencil does not draw a complete line anymore
    b. Calculate the RUL for the small dataset
    c. (Optional) calculate the average values for the SmartServo1Current
    d. Merging the individual runs into one large data set
       The "asset" column indicates the number of the run (JA_Demonstrator_1 to JA_Demonstrator_30)
       if you want to try the scripts, please only use them on one run (split the dataset and don't forget to copy the header in the first row) or edit the script to work on the full dataset -> create a loop
17. To start the data collection with the hardware run the predictive_maintenance_data_collection.ino code on the Arduino (you can find the script in the physical_demonstrator directory in this git) and afterwards run the **time_series_log.py** script. If you want to collect multiple datasets please look at line 30 and change the number after each run. Follow the instructions on the command line. If the pencil tip is not drawing a full line on the paper anymore (move the paper a little bit to check) press ctrl+c. From this point on the data will be labeled as "isDrawing = false". If you have approximately the same amount of cycles drawing on the paper and not drawing press ctrl+c again. This is done to get the same number of samples for each state of the variable isDrawing. The data set for boolean goals should contain 50% true and 50% false values. For continuous goals it is important to ensure an equal number per value.
18. Now you can run the **calculate_RUL.py** script for any individual run. Make sure to use the correct csv name in line 6 (if you changed the output name of the data collection script). The script will count the number of cycles until the isDrawing variable switched to false.
19. (Optional) If you want to experiment with feature engineering you can use **current_feature_engineering.py** script. For each run you can execute the script. During each cycle of the robot arm, data points are taken when the robot is drawing and which ones when the pen is in the air and returns to the starting point. This script sums the values for the current of the SmartServo1 during the drawing process and outputs a new csv file containing those values. If you want to use these for training, make sure to only include following features:
    a. asset
    b. time
    c. AVG
    d. RUL

    Select RUL as goal again and the only input will be the AVG value.
20. Merge the datasets. Attention: only use one header on top of the merged file and remove the headers from the other runs.
21. Upload the dataset and go through the Analytics process again.