**Devising A Predictive Maintenance Model For The Study Of Turbofan Engines- A Dive Into Machine Learning**

Project Description & Overview

This project takes a more defined look into highly innovative and emerging field-predictive analytics, which exists as a counterpart to preventive maintenance. While quality control methods like Monte Carlo simulations and sampling studies can help remove discrepancies from machines, they still lack the formidable nature to help predict an oncoming breakdown and help technicians interpret more about the remainder of the machine’s life. Lack of knowledge about such parameters can cause financial losses, depreciation cuts and even human losses. The dataset has been collected from the engine degradation simulation experiments that were conducted at NASA for a turbofan engine using C-MAPSS. Four different sets are simulated using a mix of different combinations of operational and faulty modes along with the recorded RUL(Remaining Useful Life). The aim of this model is to split up the sensor data into appropriate training and testing fragments and perform exploration tests to better interpret the relationship between normal processing and failure trends for fault evolution. The dataset was obtained from the Prognostics CoE at NASA Ames.

The dataset has been collected into a number of txt files along with the resultant RUL values. These datasets play an important role in assessing the health of the machines and devising solutions for maintenance checks and mitigate changes in time. Model creation from the collected data also helps in reduce the depreciation costs associated with salvaging the machines and formulate better scheduling periods annually for shutting down the machines for the purpose of quality control checks. A more accurate predictive model would help in achieving this, all the while reducing the possibilities of damages incurred by constant use.

Objectives and Purpose

The data set has been split into sections that include the trajectories captured from the turbine’s use along with the conditions and fault modes. Each of the datasets is unique due to the variation of these modes and faults. The first dataset consists of a sea level condition with one fault mode(HPC degradation). The second dataset consists of six conditions with the same fault mode. The third dataset is based around one condition and two fault modes(HPC degradation and fan degradation). The final portion of the dataset has four conditions with the same two fault modes mentioned previously.

Each of the records represent a multiple multivariate time series for different engines in a fleet of testing units. One of the key advantages of this dataset is that the engines have different degrees of initial wear and manufacturing variation which is unknown to the user. Three operational settings, left to the user’s choice can be implemented. The data is also filled with sensor noise.

Based on the experimental readings, the engines are operating in normal work environments at the start of each time series, but rapidly gains fault readings which increase in size and magnitude until the system breaks. The engine is operating normally at the start of each time series, and develops a fault at some point during the series. In the training set, the fault grows in magnitude until system failure. In the test set, the time series ends some time prior to system failure. The objective of this project is to develop modular interpretation of the relationship between these sensor readings and the resultant remaining useful life by predicting the number of remaining operational cycles before a failure is reported. Column 1 represents the unit number, column 2 represents the time in cycles, columns 3 to 26 are the sensor measurements for all the settings and modes.

**Solution Statement**

By processing through the signals and building predicting models, the results can assist in creating more augmented controllers that can prevent the destruction of important machinery, save capital and ensure safe working environments. The objectives of this capstone project can be thus summarized in the following points:-

1. Creating models that connect the signalling data to the RUL output and understanding how breakdowns occur.

2. Segmenting data trends and performing signalling analysis to remove trends, means and visualize results in a more uniform manner.

3. Create PCA plots to connect the relationship between parameters and outputs.

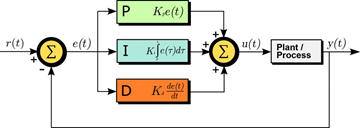
4. Generate features if possible from parameters based on timing analysis and sensor information.

5. Building model predictive controllers using neural networks or well-defined logical gates with feedback loops.

6. Additionally, use time series analysis methods like ARIMA, ARMAX and BJ models for the trends.

**PID & MPC Controllers**

Industries have conventionally used error reduction and prediction based models in PID circuits for reducing the variations between the controlled variables and predicted variables. More advanced model predictive models however can account for unexpected and unwarranted errors which significantly reduce biases, with more uniform connections to the output. One of the objective of this project will be to use the Python framework to create an MPC influenced control system for the sensor information and connect it to the output with the aim to properly pin point when the turbine will undergo damage.

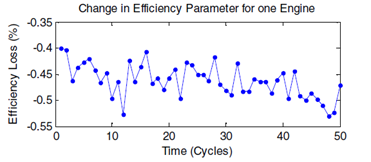


**PID Controller Breakdown.**

It should be noted that building MPC control systems does require an initial set of weights and biases which can be optimized as the model continues to learn the system and improve its coefficients.

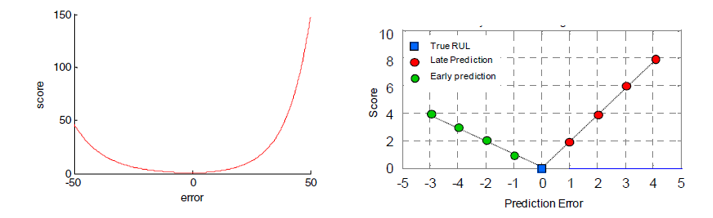
**Benchmarking and Solution Comparison**

The benchmark solution for RUL prediction is done through PID controls and for the purpose of this project, the model results will be compared against the MPC control systems. Ideally, the project will create a typical PID control unit system and track variations for a small sample of the inputs. While public benchmark models for the turbine are not available, alternatives can be generated through Python and checked for efficiency by comparing the resultant values from literature.



**Example of Prognostics data for efficiency losses in the engine.**

As per the research paper that discusses the turbine’s function and components, RUL prediction models were developed by technicians working on the system and certain results for modular accuracies can be compared. As seen from the images below, the predictive threshold for each data point along with the scores(represented as RMSE and MSE), a benchmark comparison can be made through checking how quick it takes for the model to return the system to stability. Theoretically, a PID system is not compatible for building reactive systems, but will be compared for the benchmark tests. Conventional neural network programs use classical regression programs for the models which will be replaced with a host of models including LinearSVC, Xgboost, bagging, boosting, SVM, Naïve Bayes.



**Plot of scoring losses(r2\_score) with error as well as prediction errors for true RUL, later predictions and early predictions**

**Evaluation Metrics**

There will be three major evaluation metrics for the models for this project that will cover accuracy, response and adaptability. The project will consider root mean square error, mean square error, response time, accuracy based on training to testing ratios as the major evaluation metrics. These metrics have been influenced partially by the criteria used in the research paper(listed in the references section). The remaining useful life of a turbine is integral to know in order to prevent large mishaps and as a result response to sudden changes and adaptability are important to consider as well. In order to incorporate these metrics, standard difference in response time will be compared against the conventional algorithms mentioned in the paper and compared in terms of adaptability to deal with new data.

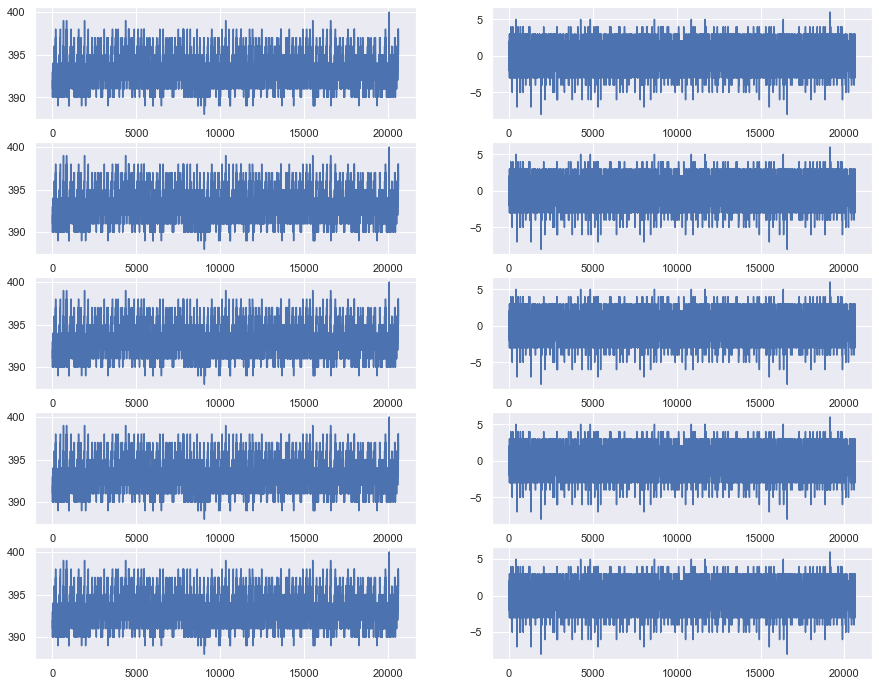
Project Design & Flow

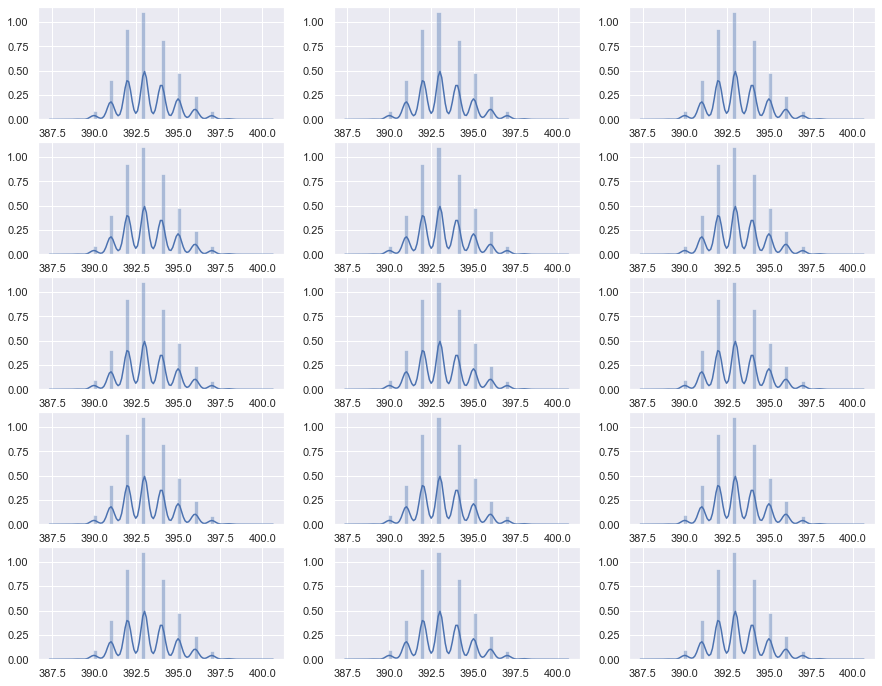
The project will first begin by parsing the values to a single file with appropriate labels and columns. Necessary columns for engineered features would be generated and a correlational plot would be generated to study the relationship between them and the RUL. The dataset will be cleaned, scaled, normalized and standardized before being analyzed using a PCA algorithm which will capture as much variance as possible. The inputs will then be regenerated as time series plots and tested for accuracy using conventional ARIMA, ARMAX, BJ models. These models would test the output RUL against a single column such as sensor reading or mode to mimic realistic testing environments.

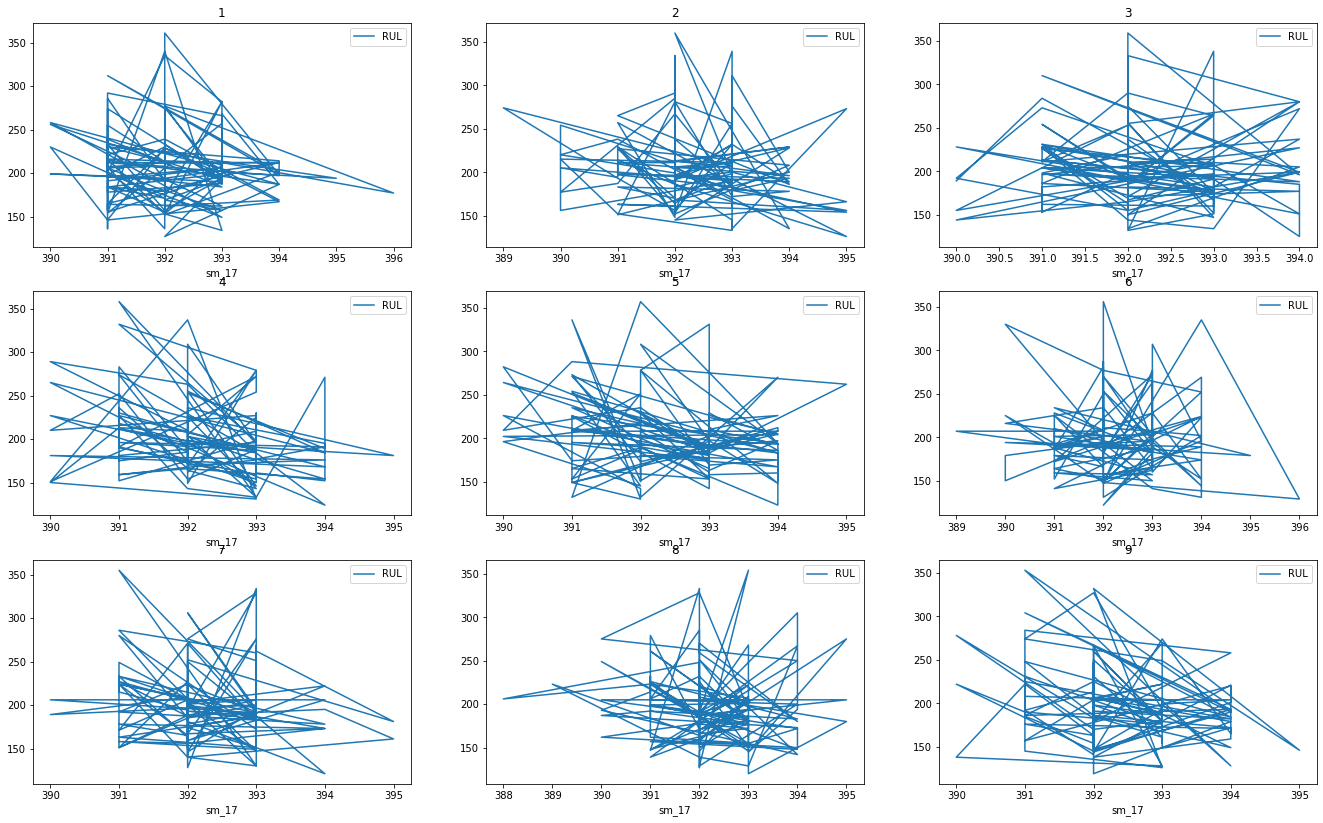
The data would then be split into a training and testing set with tests performed using a collection of algorithmic models. These models would compared the resultant metrics and come to a conclusion about which model can best predict the timeframe of a machine breakdown with the fastest response. Attention will also be put on the number of modes and conditions that each of tests use for model prediction. The final project result will be to ideally create a MPC controller either on MATLAB or Python which can be used by technicians and engineers to test their turbine data and confirm the possibility of a machine breakdown along with more significant statistics like response time and coverage of variations. The results will be deployed finally through Sagemaker and showcased on a github repo.

Exploratory Analysis

Exploratory analysis of the dataset was carried out with different columns and a correlational plot was generated to better interpret the relationship between the inputs. A sample RUL column was generated for the training set by using a conventional calculation from the time cycle values. As seen from the initial PCA tests and visualizations, not all sensor columns vary with the time cycles but still play a decisive role in determining the final RUL. More images can be compared through the repo link and the python notebook files(referenced at the end).

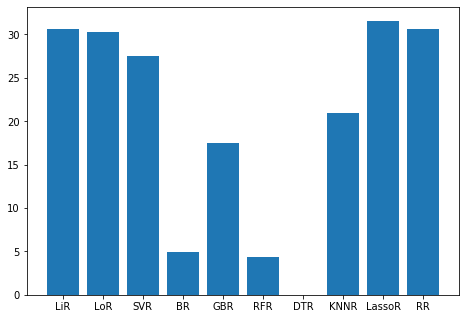


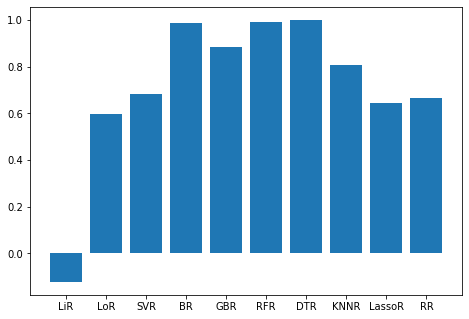




Regression Comparison Tests

Ensemble and conventional regression techniques fail to capture the changing nature of the signal strengths and predict the RUL statistics, even after the addition of addition of new features. Certain denoising tests for making the data more streamlined did not produce the intended jump in accuracy scores or reduction in mean squared errors. Tests were conducted using linear regression, logisitic regression, bagging regression, gradient boosting, random forest regression, knn regression and lasso regression among others. Descaling the inputs and outputs and removal of less feature important inputs did produce a systematic increase, but not enough to validate their use. In some cases such as decision tree regression(DTR), the model produces a near 0.0 in terms of MSE statistics, indicating an overfit case that fails to work on test data.

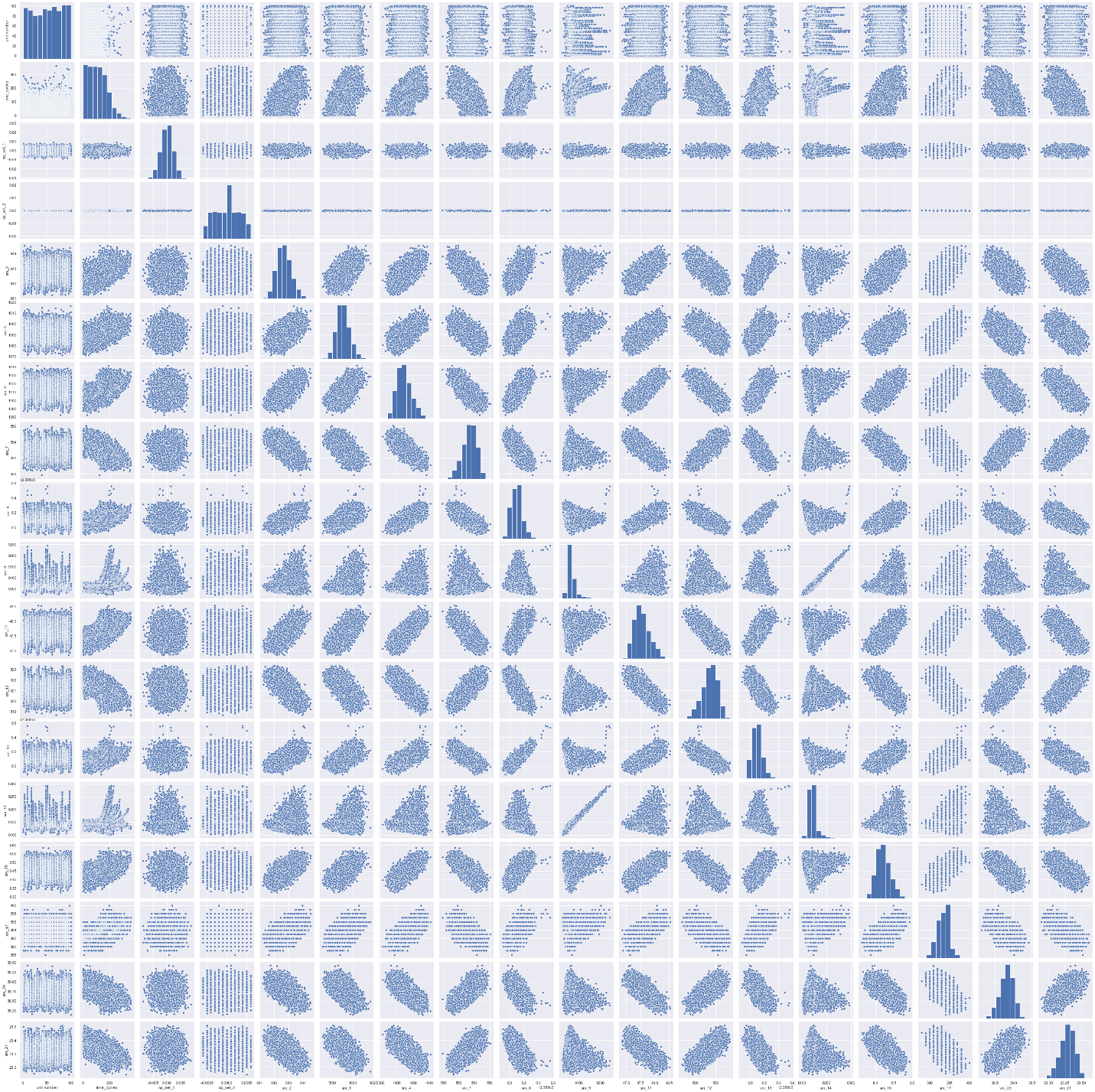




**r2score and comparison statistics for various regression models.**

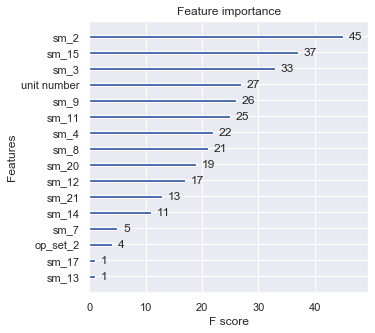
**Feature Generation and Addition to Models**

Among the methods that were included to generate new models, autoregression, feed forward transforms, Fourier series transforms and wavelet transform proved to be the most useful in capturing new information about the signals and the various modes.



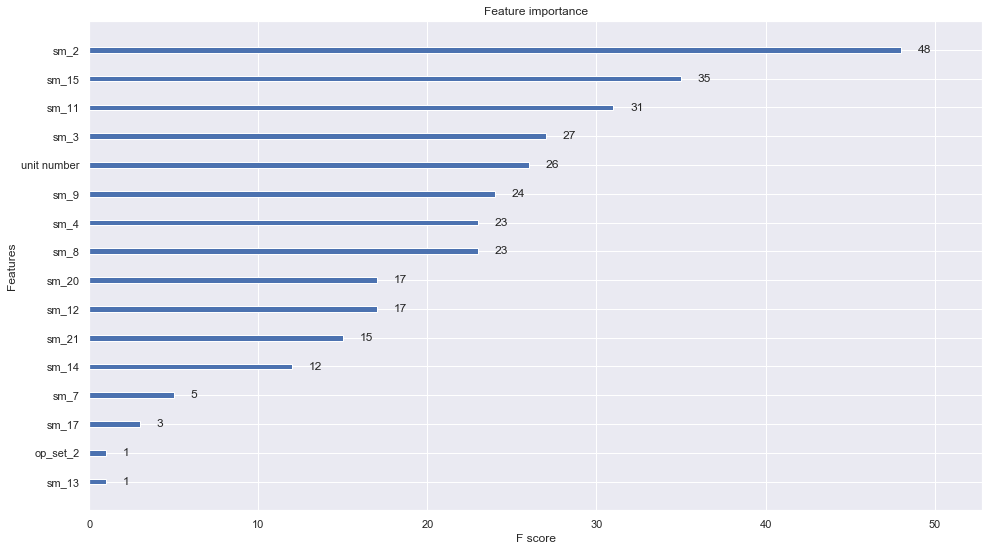
**Seaborns plot for original features**

The newly generated features were imposed for all signals and modes, resulting in a far larger input matrix, consisting of 279 features(including the time series). Due to the lack of information from the academic references about the frequency and time periods of each of the cycles, testing values were used instead.

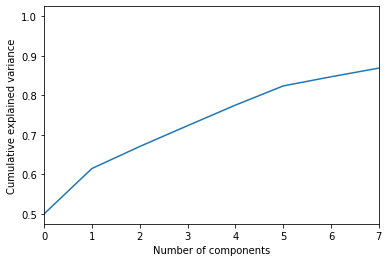


**Original feature importance plot generated from the first regression tests**

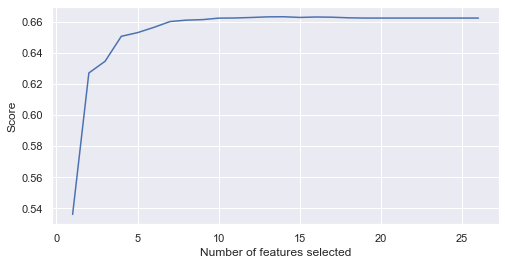
Some of the signal values were rescaled and remodeled to fit with the general indexes of the datasets. Further tests conducted from PCA plotting and correlational tests showed the number of features needed to capture reasonable variances.

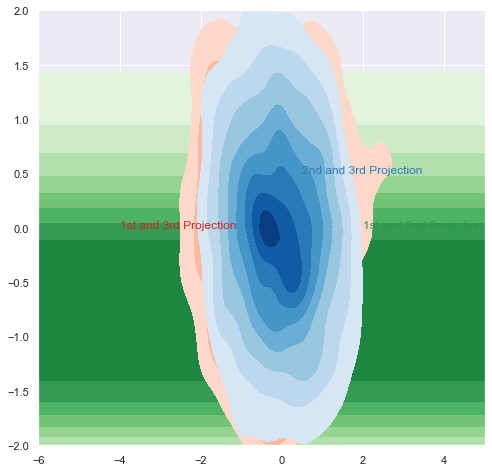


**F score feature importance obtained from Random Forest Regressor(Best model so far)**



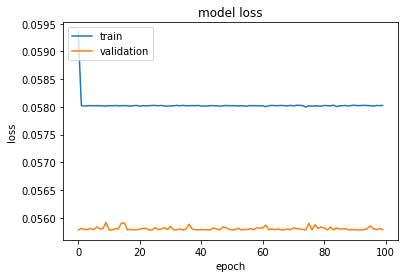
**Number of components needed to capture cumulative explained variances(PCA results)**



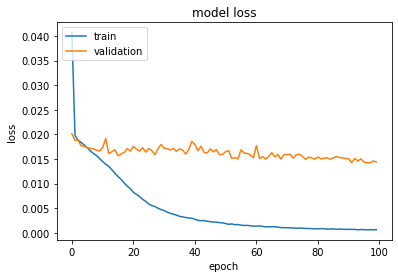


**Neural Network-LSTM, CNN and RNN Modelling**

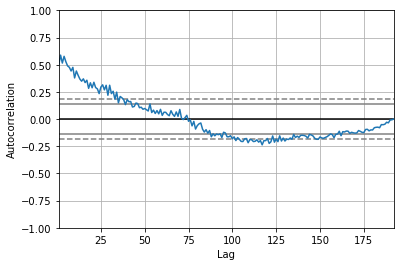
To move forward with more advanced regression modelling, one that could account for inconsistencies in the signals and random outliers, a series of tests were conducted to find the best possible configurations for building a network to predict RUL statistics. Dense layers with neurons and different number of inputs were tested against the output to arrive at the most accurate models that can be built on top of the baseline models(Code can be seen in the Tuning.ipynb file in the repo). Extensive tests pointed towards the obvious fact- LSTMs are definitely superior in this context. The question however was how fine tuned could a network be made. Adam, ReLU and tanh regressor proved to be best activation functions in terms of their ability to cut down on model losses. The project further narrowed down to using ReLU and tanh due to their speed in recovering outputs with testing data.



**Model losses using relu activation function.**

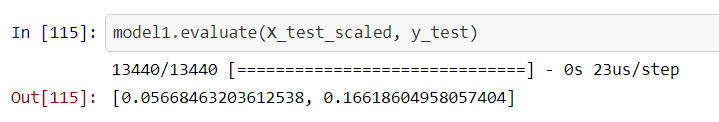


**Model losses using tanh activation function**

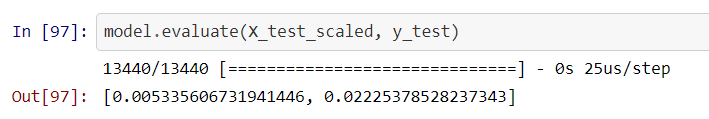


**Autocorrelation corrected signaling data for sensor 17.**

On first glance, it would appear that the ReLU configuration is far superior as the system reaches a constant loss value but is still not that strong in predicting outputs for testing data, making tanh the best possible model activation function for outputs. As a comparison, the images below show the MAE and MSE values for both cases.



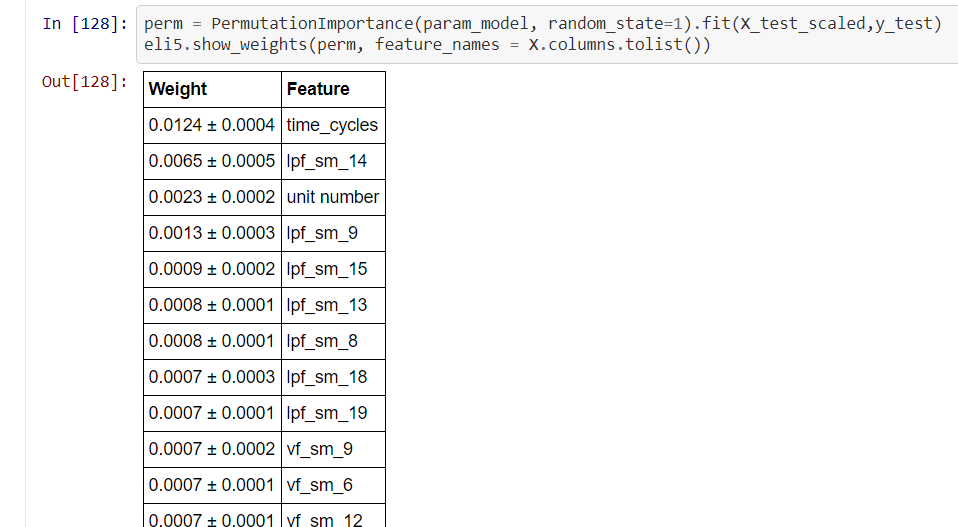
**RELU model statistics on testing data.**



**TANH model statistics on testing data.**

Conclusions

We can thus confirm from the modelling and various tests performed on the signaling data, that feature generation plays a decisive role in determining how strong the model becomes in predicting numeric outputs. A simplification of this problem that would improve the accuracies of conventional regression model would be to convert the RUL statistics to non-numeric outputs, essentially creating a classification problem that assigns a value of 1 or 0 depending on whether the machine will break. This will however make it more complicated to know more about the machine’s break point. Thankfully, LSTM and neural network models serve as great replacements for the conventional and ensemble regression techniques that are often used for baseline models. The future of this project will now look into the use of these statistics to build more accurate PID systems. Work in this area is already being done and some notebook files for PID modelling have been included in the repo. Additional work is also being done on the use of neurofuzzy systems if applicable. The final feature importances from the tanh model can be seen below for the best predictors of RUL in the image below.



**Permutation based feature importances and weights obtained from testing data(More can be found in the LSTM.ipynb notebook in the github repository).**

**Acknowledgements**

I would like to express my gratitude towards the machine learning community online, Udacity and AWS for this unique opportunity. Further gratitude towards data scientists and machine learning engineers referenced in the repository.

Resources and Links

**Citation Source**

A. Saxena and K. Goebel (2008). “[Turbofan Engine Degradation Simulation Data Set](https://ti.arc.nasa.gov/publications/154/download/)”, NASA Ames Prognostics Data Repository (http://ti.arc.nasa.gov/project/prognostic-data-repository), NASA Ames Research Center, Moffett Field, CA

**Additional Links**

**Project Github repository link-**<https://github.com/AmDeep/ML_Nanodegree_Capstone_Project>

[**NASA PCoE Datasets Source**](https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/#turbofan)