NASA Jet Engine RUL

Paul Casey

29 April 2024

**Abstract**

The purpose of this paper is to document the process of analyzing sensor data of four different engine types, cleaning and processing the data, building and optimizing models for the prediction of Remaining Useful Life (RUL), assessing the best predictive model, and determining the business value of the model and subsequent findings. By conducting these steps, we should have a better understanding of how many cycles until a specific engine type fails, what sensors are necessary for this prediction, and provide business value and insights for these engines. With these insights, we hope to provide better maintenance strategies, reduce aircraft downtime, and provide suggestions for points of failure for each engine to be improved upon.

**Methods**

EDA

It is important to note that this problem required importing four data frames for training and four data frames for testing as well as four RUL vectors for the testing data. I decided to keep all of these separate and not combine them because each data frame is for a different engine type. This will mean that instead of having one model to predict RUL, I will have four separate models for each engine type.

The initial step in my EDA process was to obtain information on how many missing values were in each data frame. The last five sensors of each data set were missing values in every row, so I removed those columns. This left me with four training data frames and four testing data frames that had no missing values.

Next, I looked at the standard deviations for each column and noticed that many had 0.0 for that value. This means that that column had no fluctuation and would therefore have no statistical significance. The image below shows how many columns were removed from each data frame due to not having any standard deviation:

A screenshot of a computer screen

Description automatically generated

This is when I first noticed a pattern in our data sets, with the pattern being engine types one and three were heavily related and engine types two and four were heavily related. Now that I had removed the columns with no statistical significance, I looked at the distribution plots as seen below:

A screenshot of a graph

Description automatically generated A screenshot of a graph

Description automatically generated

I only show two here because the same patterning pops up for the other data frames, but we can see that training data frames one (and three) have a normally distributed data set, but training data frames two (and four) are mostly not normally distributed. What this means going forward is that when I start doing transformations on these data sets, I will have more success fixing skew in data frames one and three and have little success in two and four because they are so far from being normally distributed. I then plotted correlation heat maps for all four, but again I will only show two since the trends are the same for each data frame pair:

A screenshot of a computer

Description automatically generated A screenshot of a computer screen

Description automatically generated

As shown in the plots above, we see that data frames two (and four) have predictors that are heavily correlated while data frames one (and three) have some correlations among predictors, but not as severe as the first two. Going forward, dimension reduction techniques such as PCA will be greatly beneficial for the latter two pairs of data frames and might have some benefit to data frames one and three. This way, I will not run into multicollinearity problems during the modeling process.

Transformations

As seen by the EDA, all four data sets needed to be treated individually instead of as a whole. What this meant going forward was that all transformations, feature engineering, and modeling needed to be done on each data set. Data sets one and three were similar while two and four were similar so most likely what would work on one of the components of the pair would work on the other component with some minor adjustments.

Data sets one and three were mostly normally distributed, but I did need to apply square root and log transformations on some of the predictors, more specifically three transformations for data set one and seven for data set three. At this point, data frames one and three were prepared for modeling, but two and four needed to be transformed.

Data frames two and four were mostly not normally distributed and had such strange distributions that any type of log, square root, or power transformation did little to nothing. There was also the issue of multicollinearity that needed to be addressed as well. Having these two pieces of information, I decided it was best to do some type of dimensional reduction and settled on Principle Component Analysis (PCA). This would solve my multicollinearity issue and hopefully allow the models to fit better. I decided to stop the creation of principal components once 99% of the feature space was explained. Three components were selected for both data sets and below we can see the explained variance for the principal components.

A screen shot of a computer

Description automatically generated

Modeling

My strategy for modeling involved passing my data frames through many different models and then comparing the mean squared error (MSE) for each model to see which one performed the best. Once I found the best model, I would pursue it more thoroughly to see how well it performs by calculating the R-squared value and the root mean squared error (RMSE). The R-squared value tells me how much vairance my model is explaining and the RMSE would show me on average how close I am to the true RUL value. The trickiest part of this process was trying to balance the bias-variance trade-off.

A computer screen shot of a program code

Description automatically generated

**Conclusions**

Modeling Results

For data frame one, the best-fitting model was a Random Forest model with a depth of ten. It was only able to explain 51% of the variance and on average it predicted within 41 cycles of the true RUL. Time in cycles was the most powerful predictor followed by sensor measurement 11.

A screenshot of a computer program

Description automatically generated

Data frame two’s best-fitting model was the LightGBM (Gradient Boosting) which was able to explain 45% of the variation in the test data set and be within 47 cycles of the true RUL. To see what features have the biggest impact we can look at the loadings of the first principal component and see which values have the highest absolute value. The top two are sensors nine and eighteen.

A screen shot of a computer

Description automatically generatedA screenshot of a computer program

Description automatically generated

Data frame three’s best-fitting model was a gradient-boosting model with a depth of five. It was able to explain around 39% of the variance in the test data set and was within 66 cycles of the RUL on average. The most important features were time in cycles and sensor four.

A screenshot of a computer program

Description automatically generated

Data frame four’s best-fitting model was a LightGBM (same as data set two) and it explained 31% of the variance in the testing data set and was within 77 cycles of the RUL on average. Again, we need to look at the loading values for the first principal component for this data frame to see which predictors are the most important which are sensors nine and eighteen.

A screen shot of a computer

Description automatically generated A screen shot of a computer

Description automatically generated

Overall

The biggest limitation in all of these models is that there is not enough data. Since we were not able to just have one model for the entire fleet because in the EDA process, we discovered that each engine type had its uniqueness of data, we simply do not have enough data to build strong predicting models. This is not to say that the models we have are not promising, but they need to be trained on more data so that they can be optimized appropriately. Once we have more data, I believe that through more testing of models, we can create a model that accurately can predict the RUL for its engine type.

What is interesting is that even with the small amount of data, we can see that there is a very complex relationship between the features and the targets as no linear models performed well to include lasso and ridge. I believe this also supports my conclusion that for us to output a model worth deploying, we need more data.

**Statement of Business Value**

In each of our models' current states, I cannot conclude that they bring business value right now, but with more data and training, they could become invaluable. As of now, on average, our models predict within 58 cycles of the true RUL which is not useful right now. With more data though, it would allow me to evaluate more complex models better and potentially get into the single digits of predicting RUL. This would be of enormous value to the business because downtime due to engine failure would be dramatically more predictable.

Our models as of right now are somewhat interpretable, so the importance of each explanatory variable can be known. If I can keep the complexity of the model down to where it is now, we could also potentially get rid of sensors that have no value in predicting RUL whose sole job is to do that. As I come from a background in aviation maintenance, the fewer things that can break on the aircraft, the better the reliability of the aircraft.

Going forward, if this project is allowed to move forward and I can get the data needed, I see this model being very useful to production control and quality control teams to track expected RUL. When reports are pulled for the status of the aircraft as it enters phase maintenance, having a predicted RUL for the engines could help determine if they need to be replaced or not. This also gives production control and quality control more information to make these very costly time and money decisions. In conclusion, I do not doubt that the pursuit of better models would save time, money, and headaches for the company.