Remaining Useful Life Prediction

Link for recording: PresentationRecording

By: Guru Manikanta Innamuri

UMID: 68448892

The Problem

- Predicting RUL
- It’s Estimation/prediction
  - Physics based
  - Data driven
  - Hybrid
- Usage
  - Prognostics and health management (PHM)

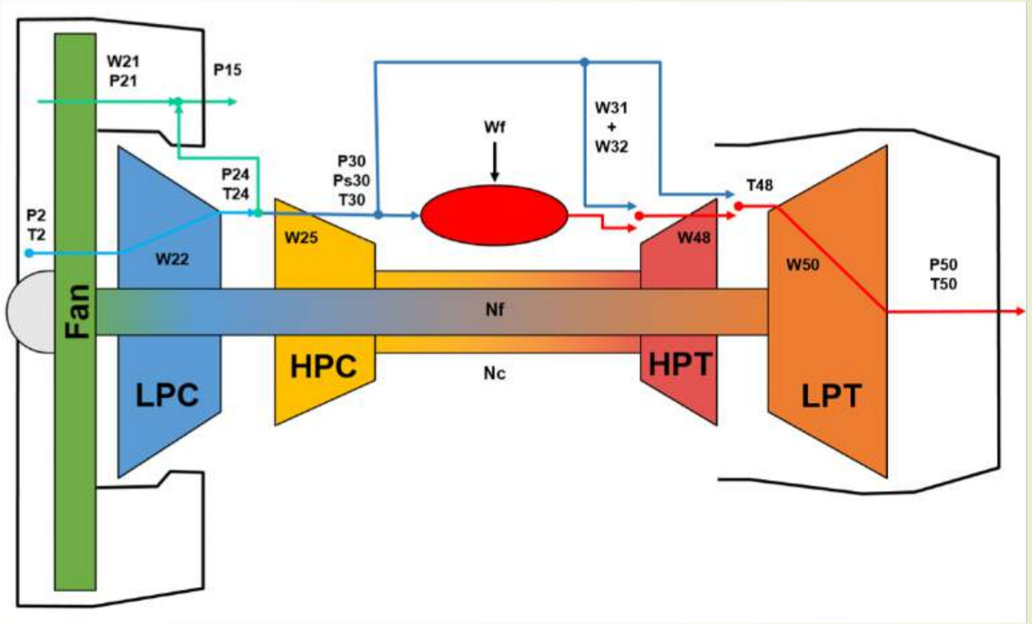


Image from:
Citation: Arias Chao, M.;
Kulkarni, C.; Goebel, K.; Fink, O. Aircraft Engine Run-to-Failure Dataset Under Real Flight Conditions for Prognostics and Diagnostics. *Data* **2021**,*6*,5.
https://doi.org/10.3390/ data6010005

Data Driven Model

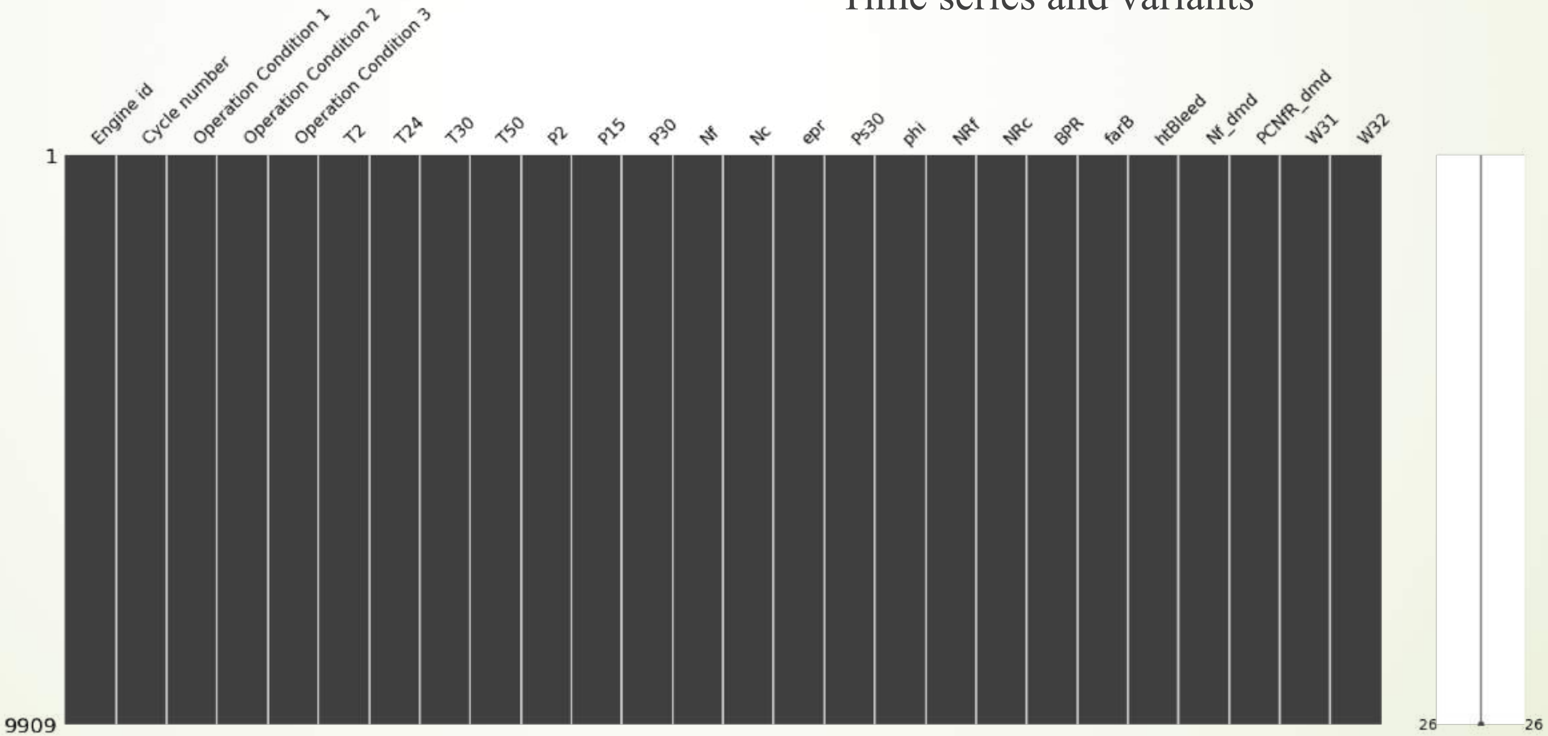
- Statistics based
  - Regression analysis
  - ARIMA (forecasting)
- Artificial Intelligence
  - Neural Networks
  - Random Forests

Dataset and Formulation

- NASA turbofan dataset
- Sensors, operation conditions and cycle
- Design Matrix with sensor information

Multiple ways to define target

- Using Health Index
- **Remaining cycles as regression**
- Classifying the number of cycles to bins
- Time series and variants

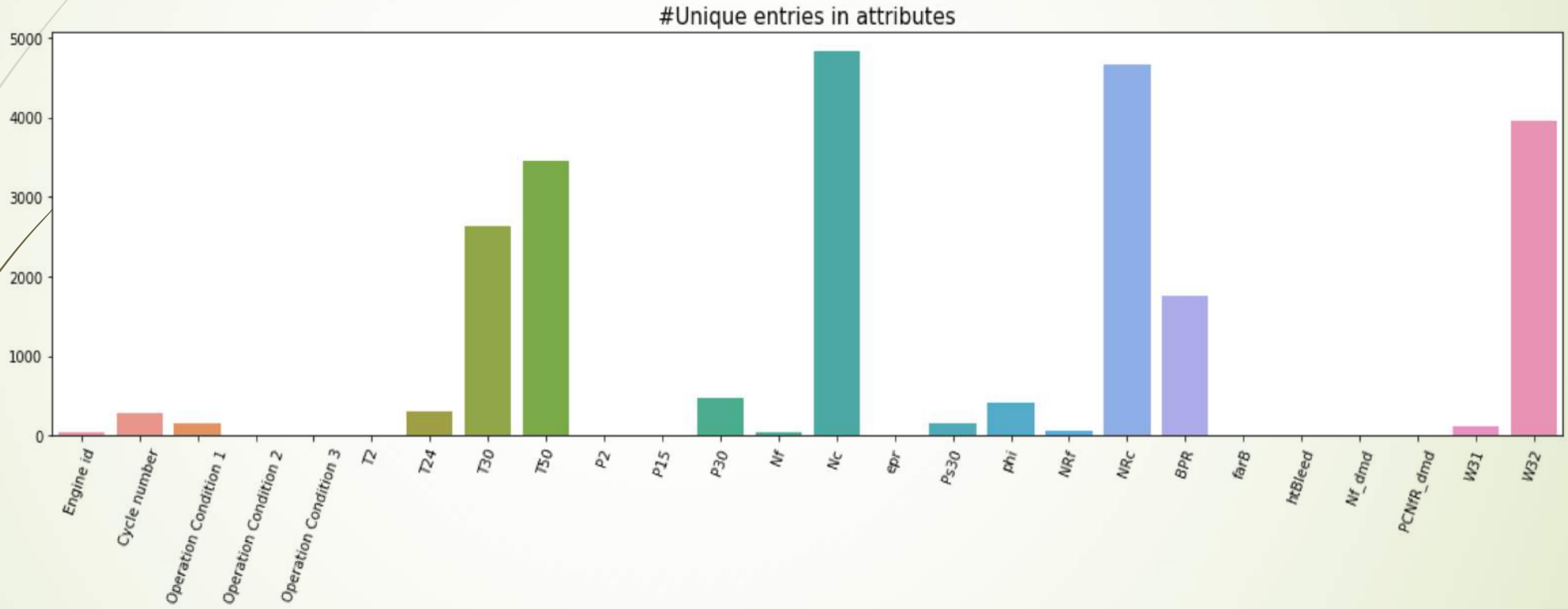


Missingno Matrix

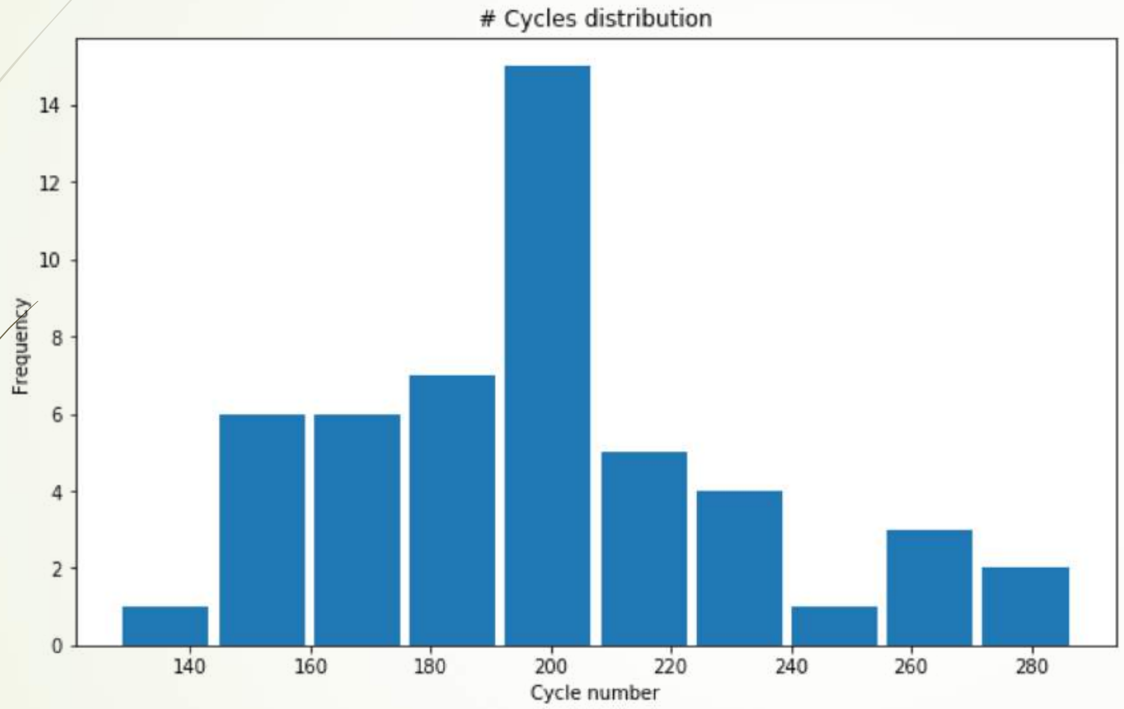
Proposed Method & Rationale

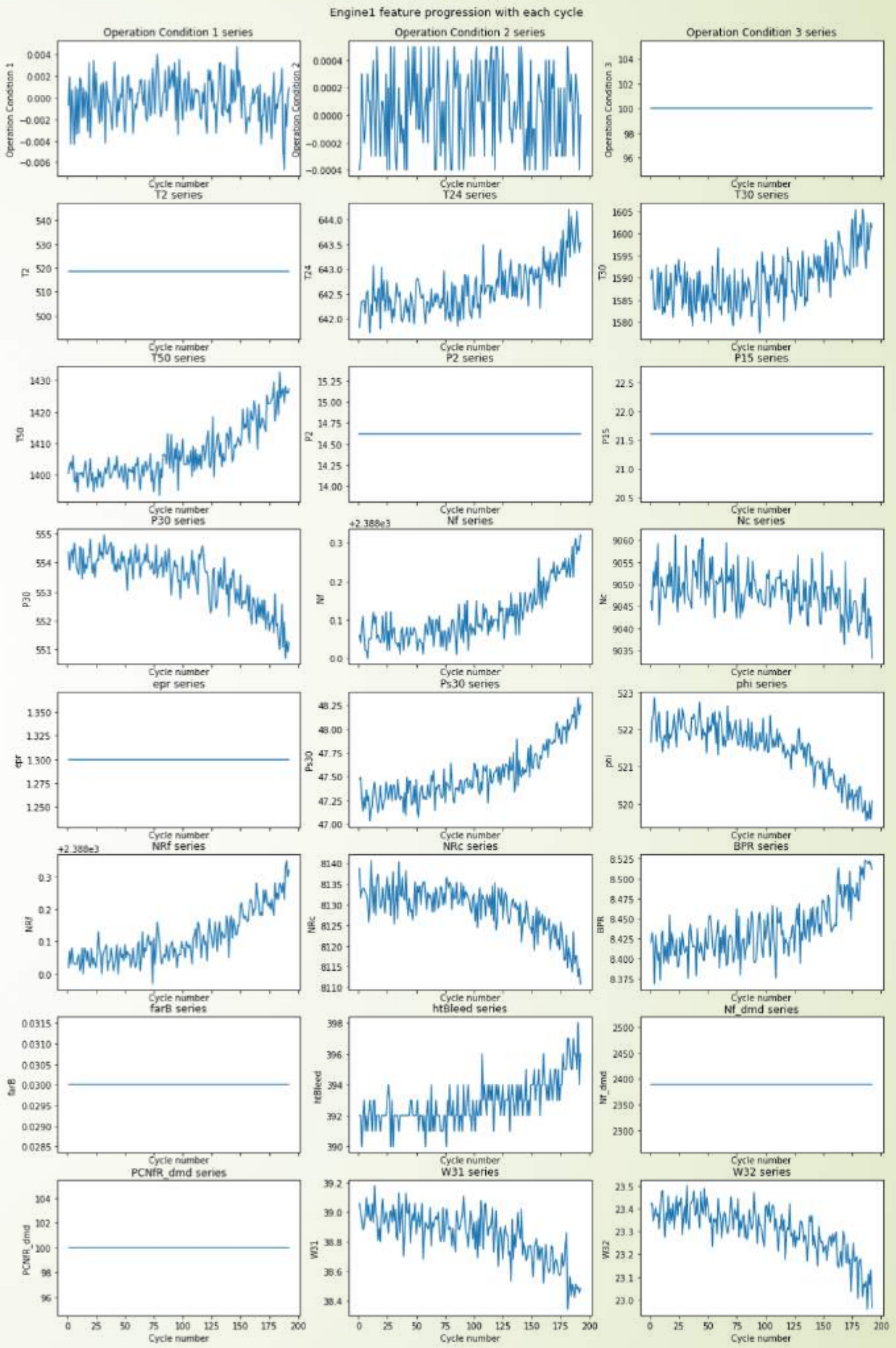
- Number of remaining useful cycles
- Designated target variable
- Difference between current cycle and max cycles from training
- Validation strategy
  - Random grouped
  - Last from group
- Cycle of interest is the last given from test
- RMSE, R-squared

Understanding the Data

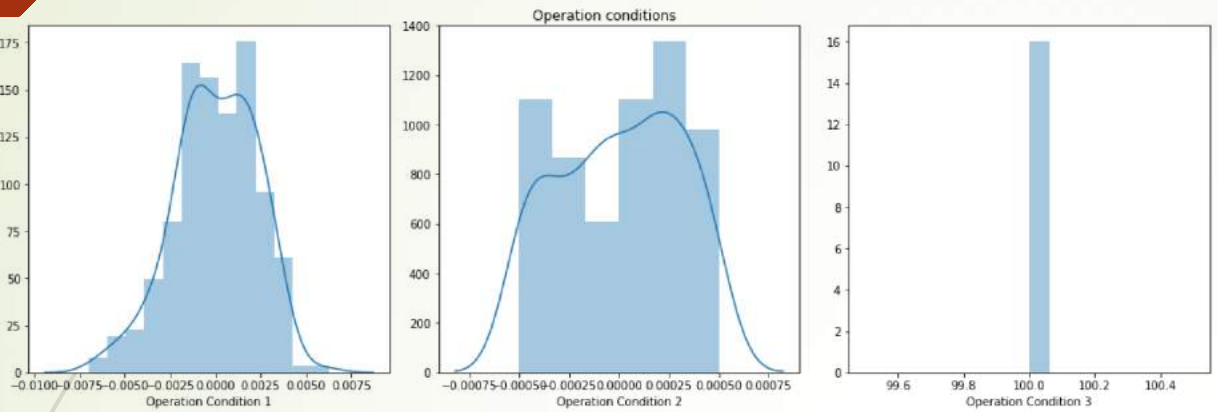


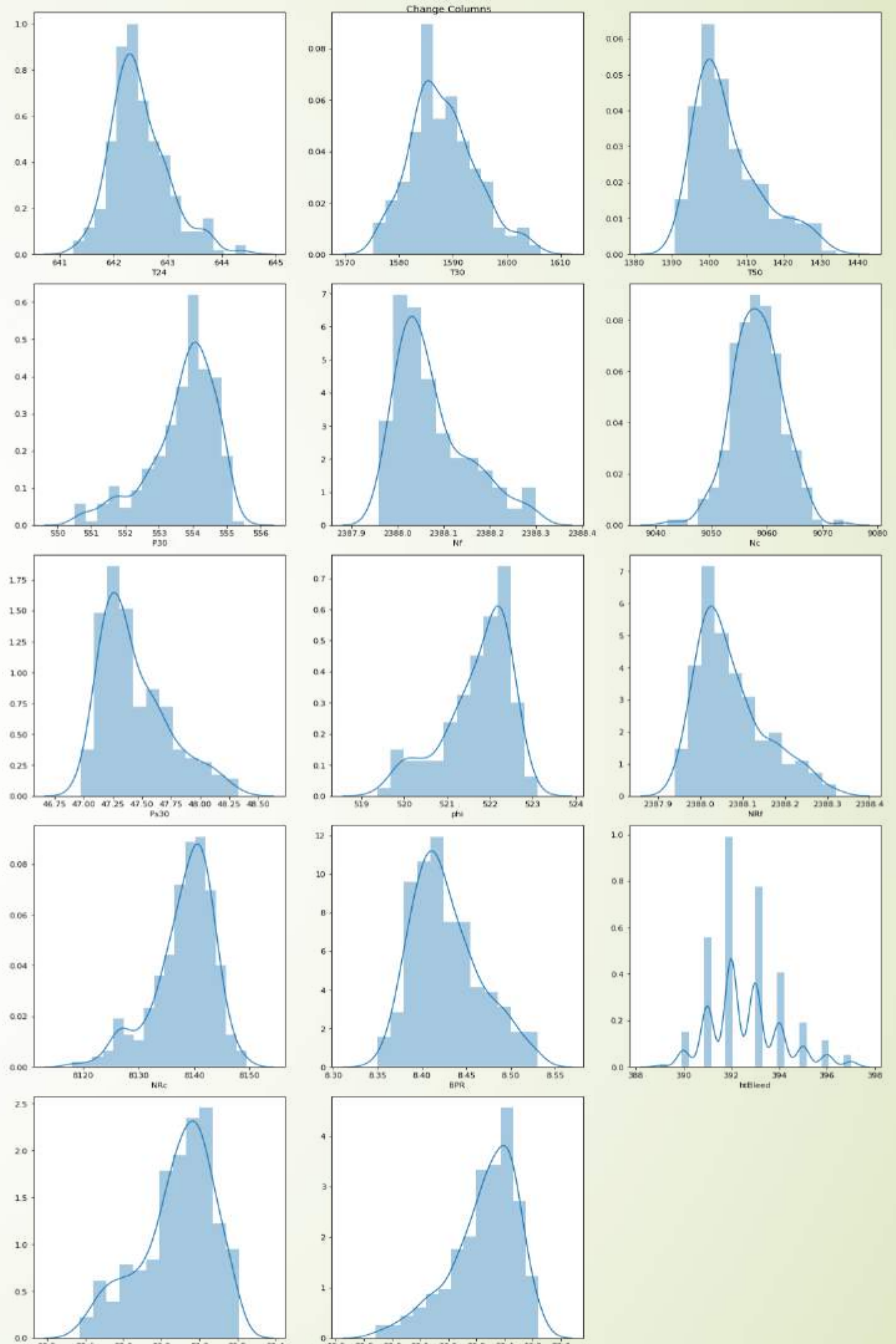
Understanding the Data



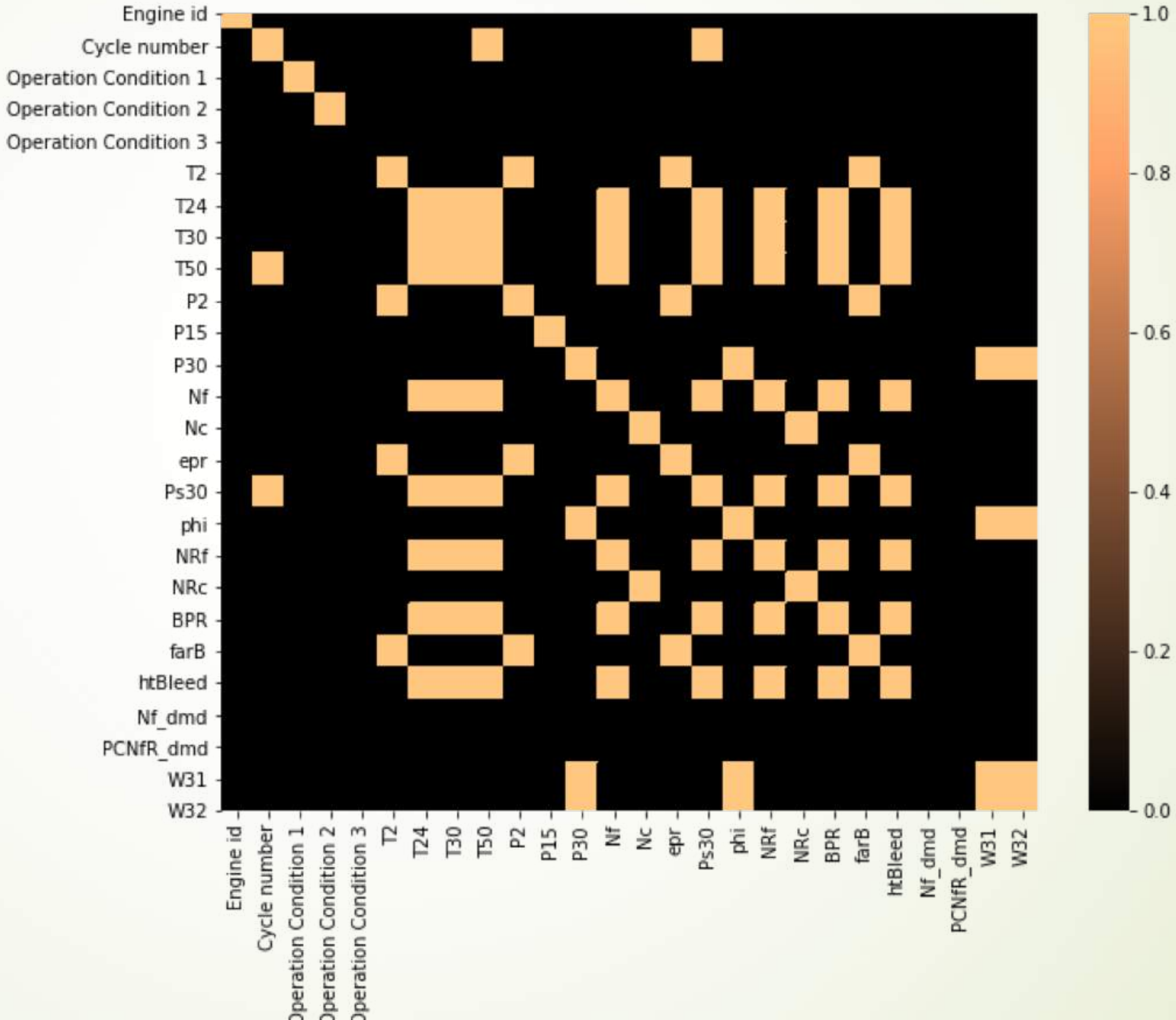


Understanding the Data



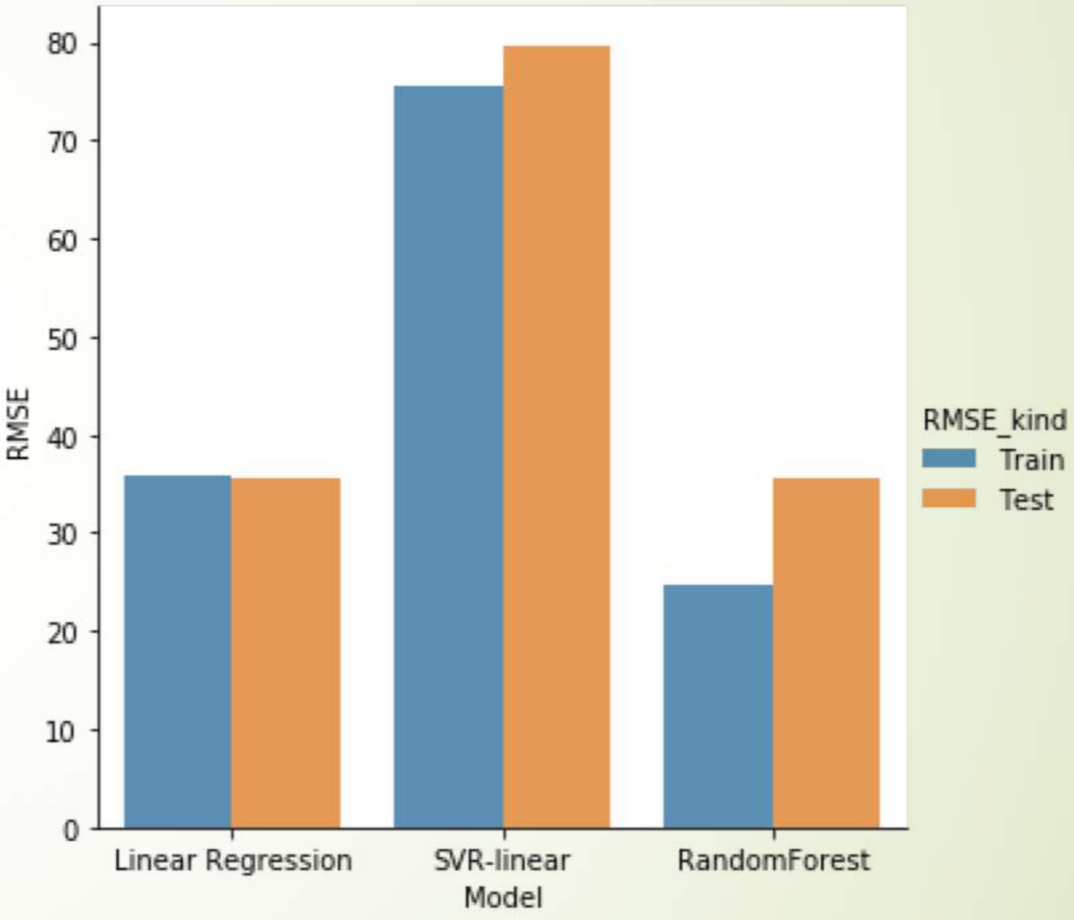


Linear correlations

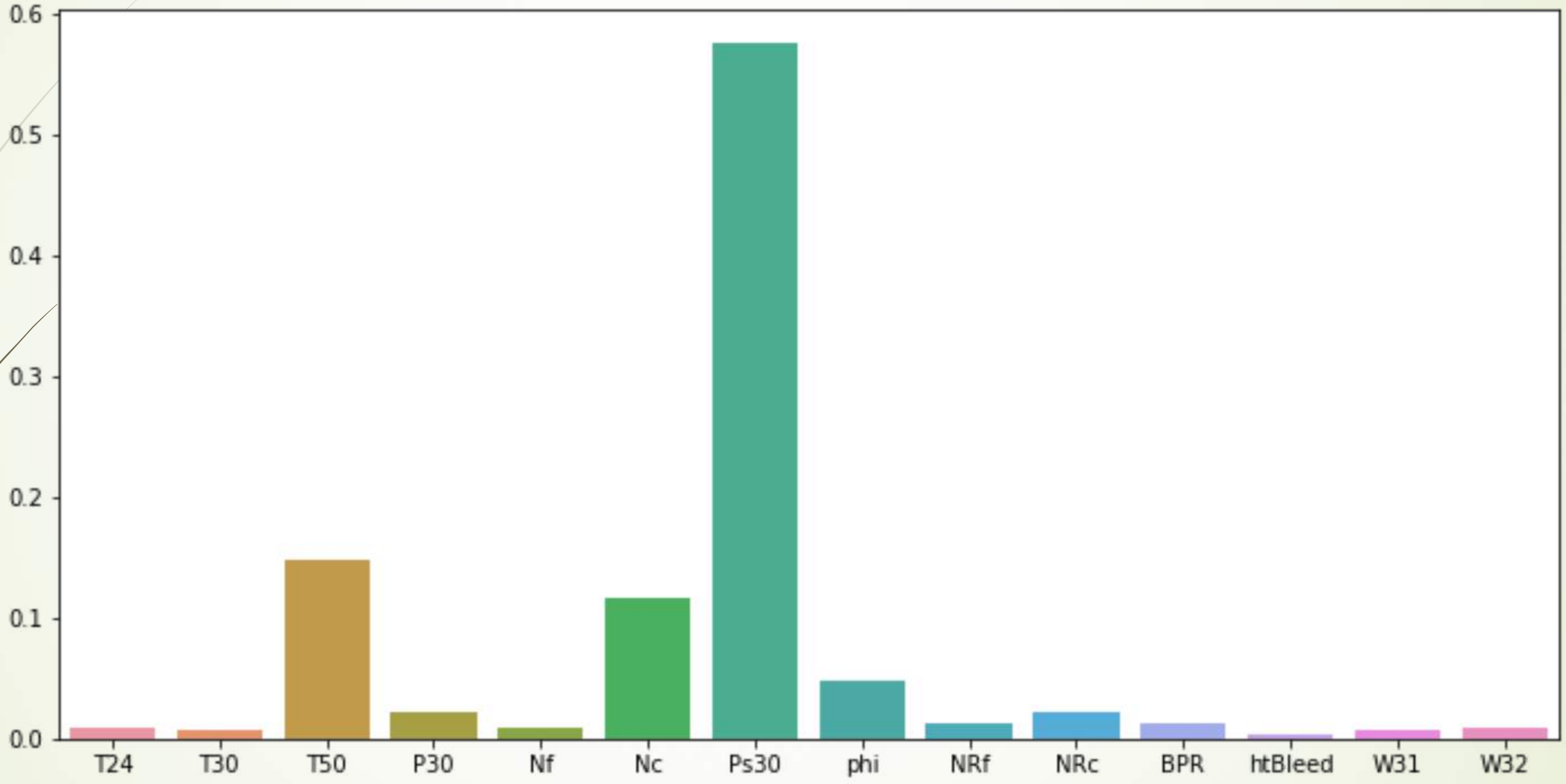


Model Development

- Preliminary models
- No preprocessing
- Some tuning on validation
- LinearRegression was best at around 35RMSE for both test and train

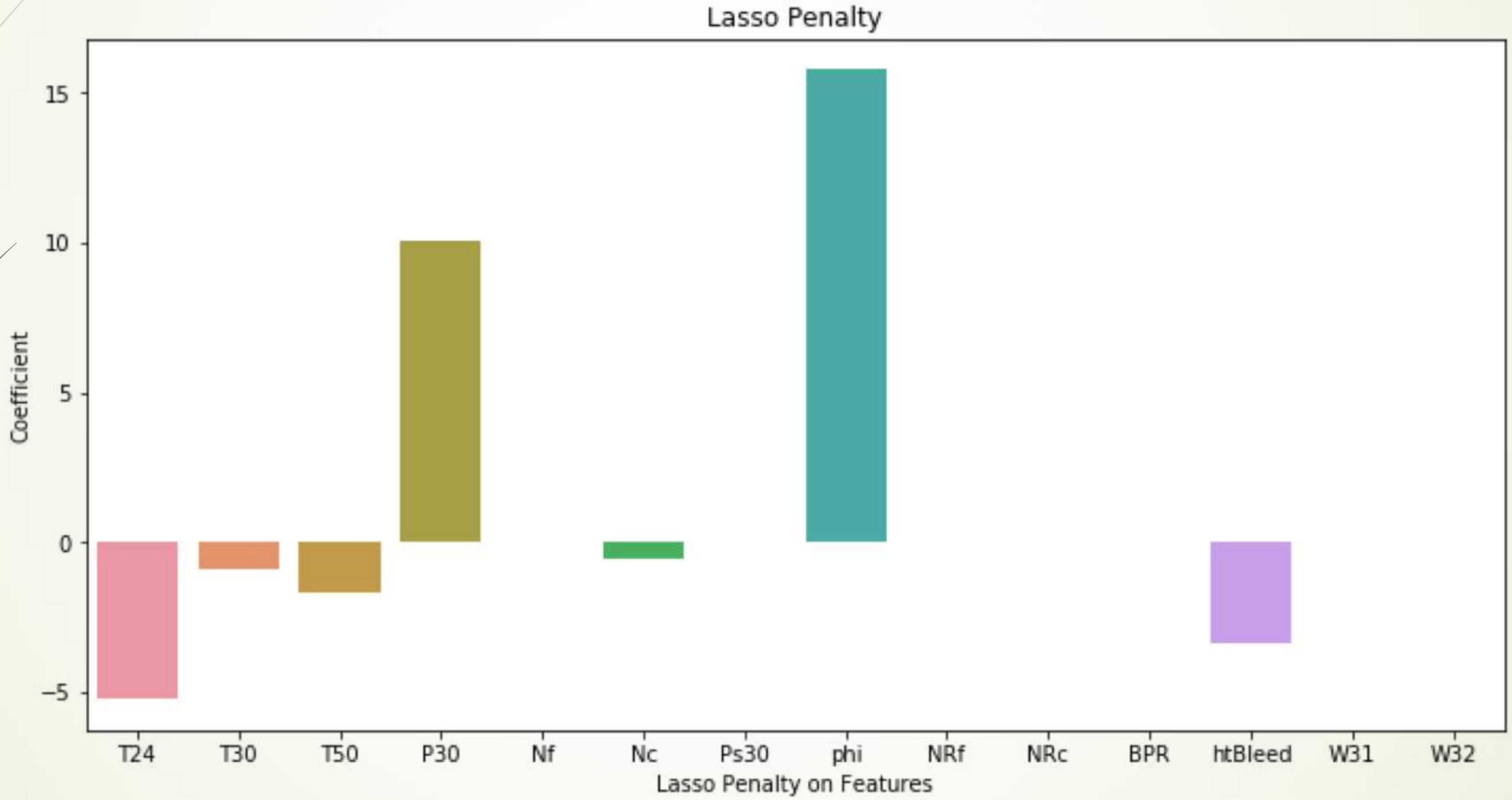


Model Development – Feature Selection



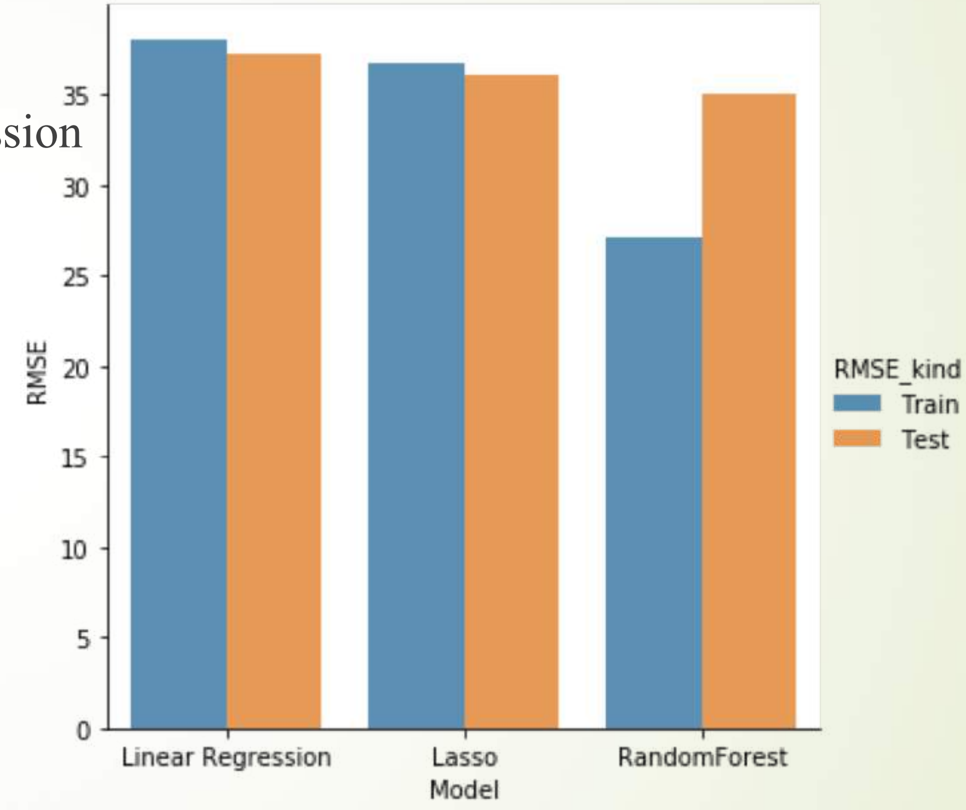
RandomForest Feature Importances

Model Development – Feature Selection



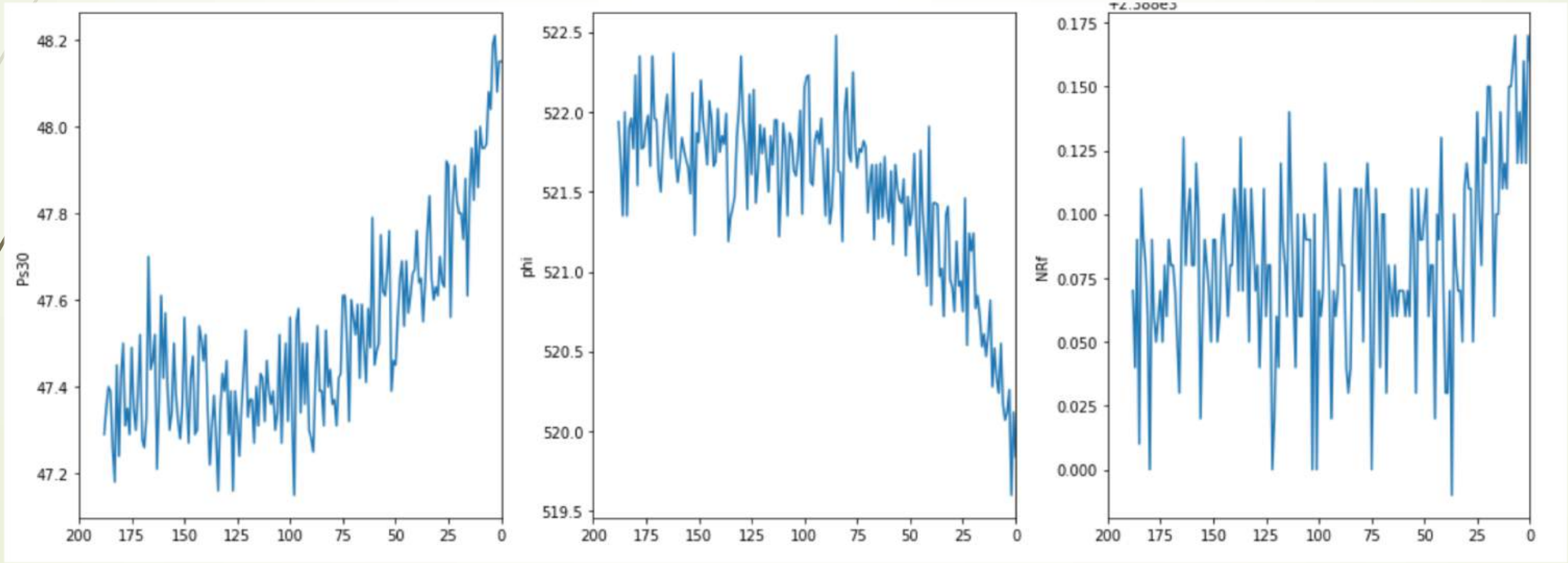
Model Development – Feature Selection

- Non-overlapping
- Backward elimination on linear regression
- Not a great result



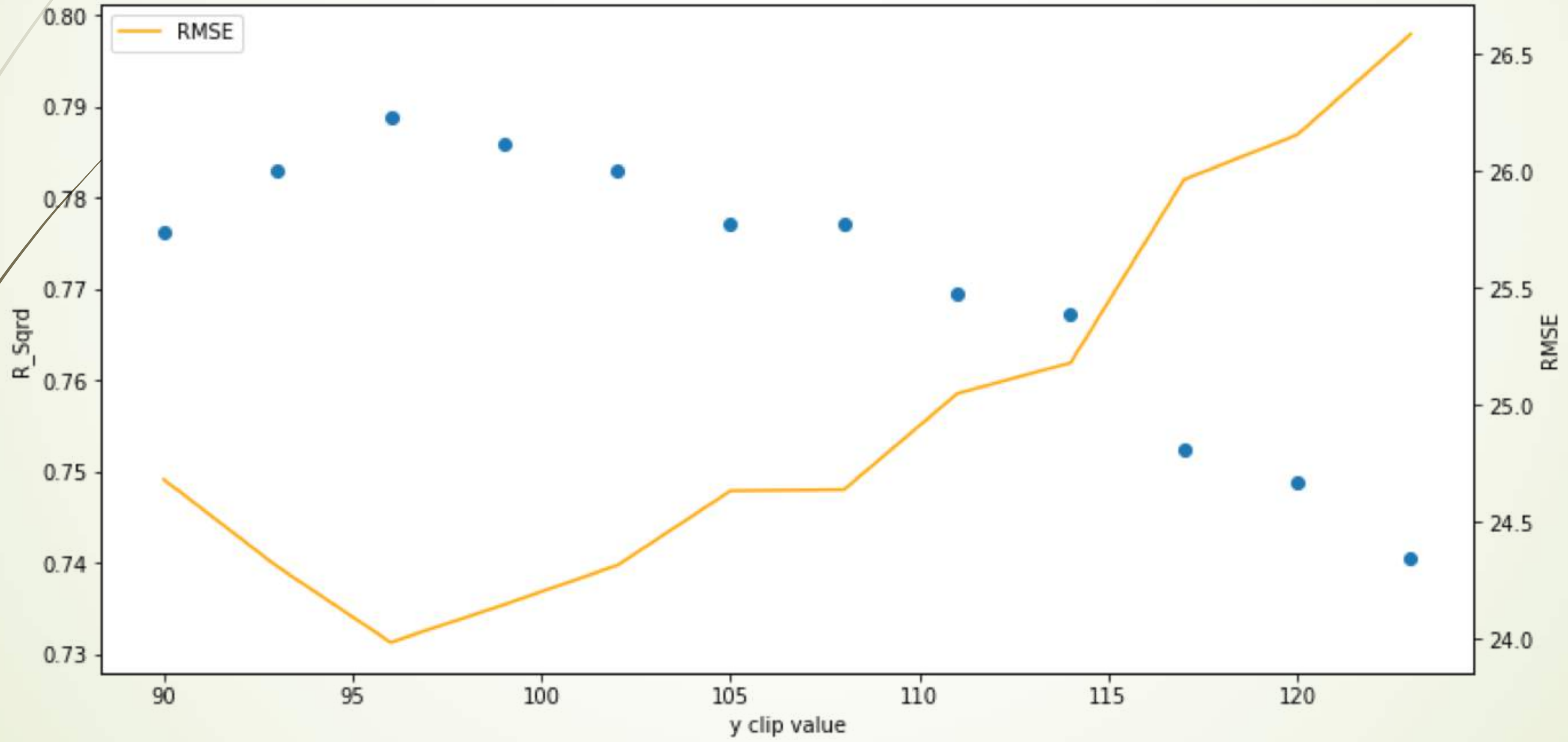
Alternate Feature Processing

- Revisit sensor vs cycle
- Pattern is valid for multiple engines
- Clipping might work



Clipping

- Set a threshold on cycles_to_failure
- A set of thresholds are tried to find the best one
- Random Forest had the best performance



Clipping - summary

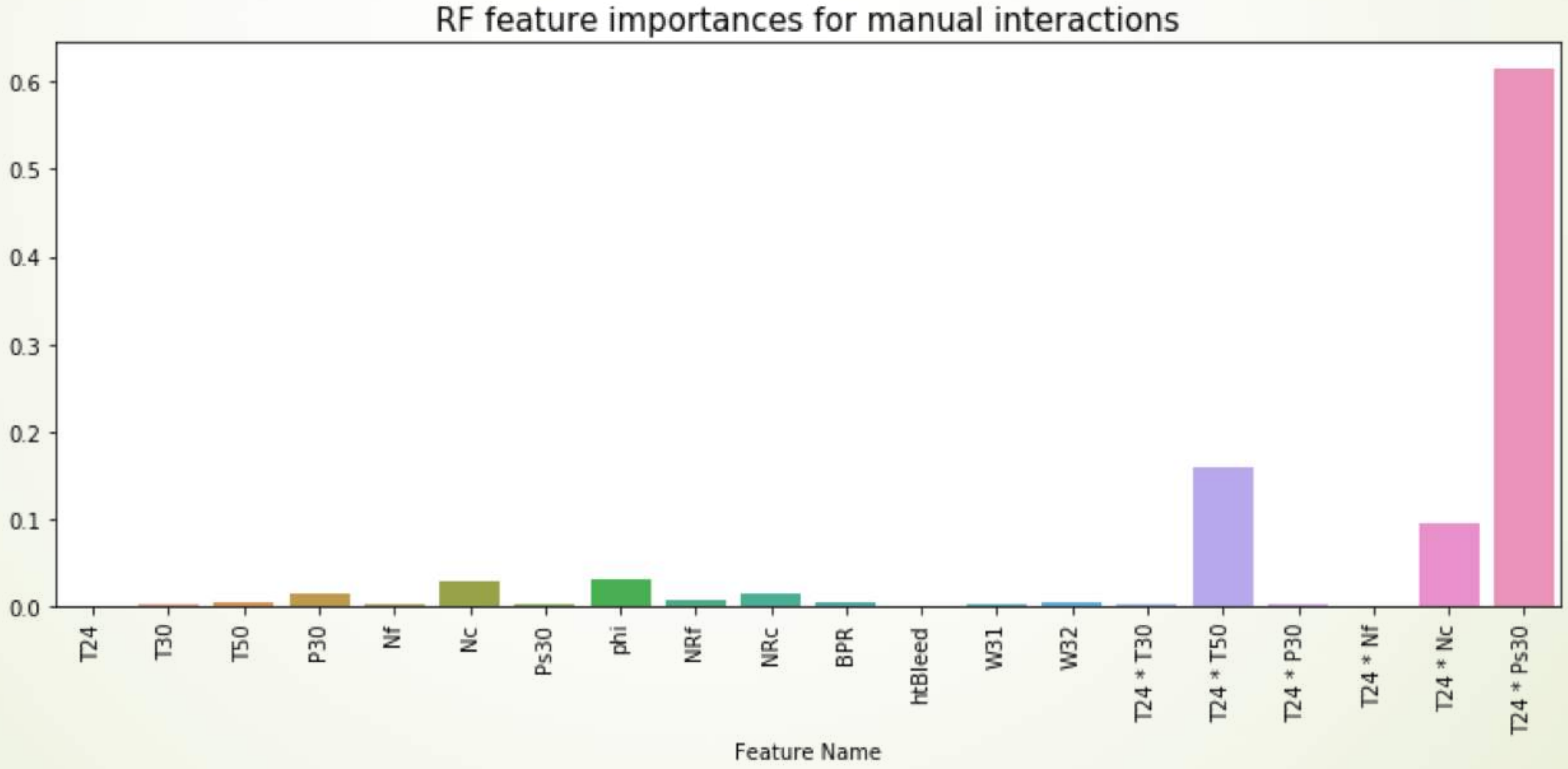
- Applied on svm and Random Forest
- Thresholds applied on validation
- Random Forest had best RMSE = 24.21 , R-squared = 78.47 balance on test while train was RMSE 23.9 and R-squared = 89.03

Interactions

- Dataset is built explicitly based on the formula

  '''cycles_to_failure ~ 1 + T24 + T30 + T50 + P30 + Nf + Nc + Ps30 + phi + NRf + NRc+ BPR + htBleed + W31 + W32 + T24 * T30 + T24 * T50 + T24 * P30 + T24 * Nf + T24 * Nc + T24 * Ps30'''

- Verified some combinations by hand



Interactions

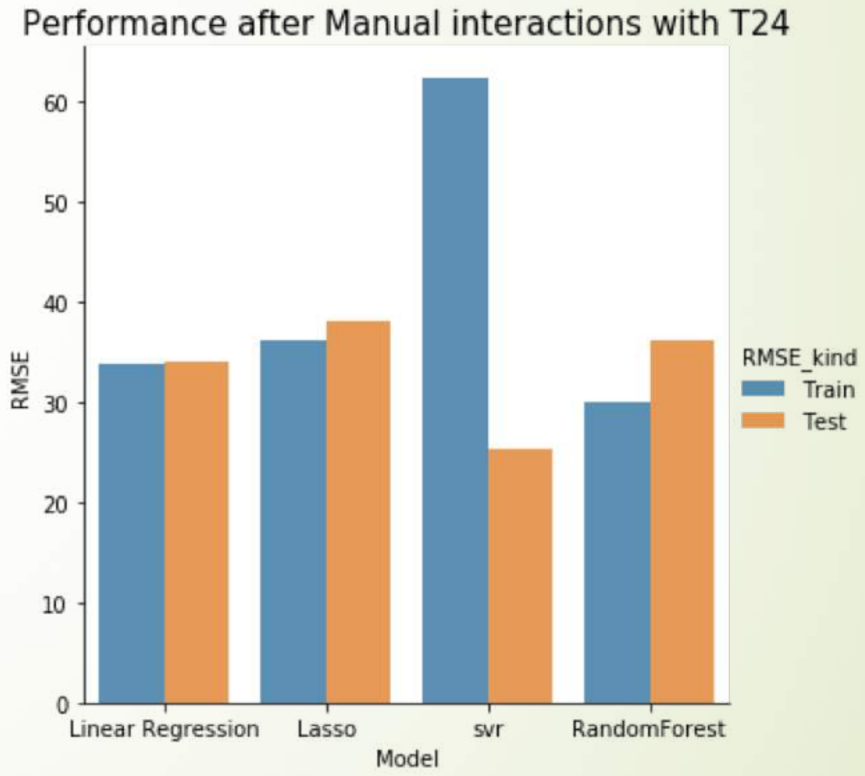
- Results for below formula

  "'cycles_to_failure ~ 1 + T24 + T30 + T50 + P30 + Nf + Nc + Ps30 + phi + NRf + NRc+ BPR + htBleed + W31 + W32 + T24 * T30 + T24 * T50 + T24 * P30 + T24 * Nf + T24 * Nc + T24 * Ps30'''

- T24 is weighed less than it's

  Interactions

- They become too much too soon



Interactions – Polynomial Features

- A formula can only get you so far

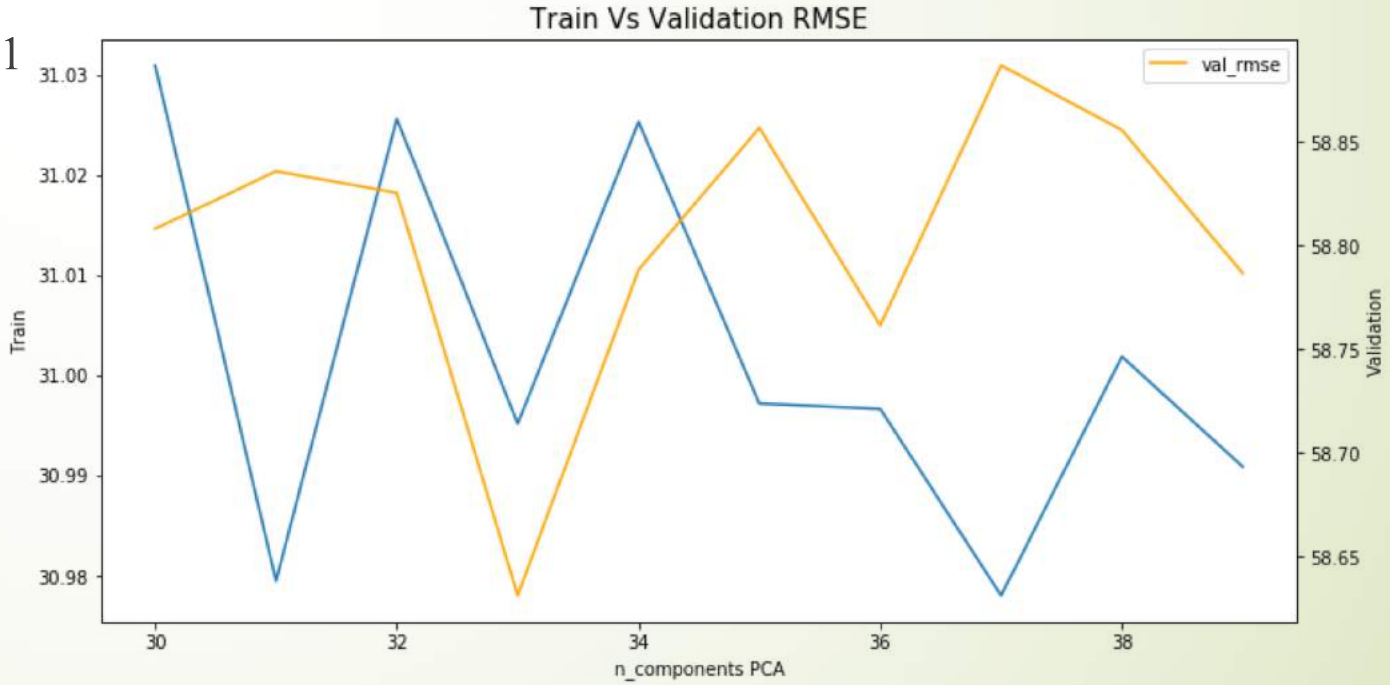
  '''cycles_to_failure ~ 1 + T24 + T30 + T50 + P30 + Nf + Nc + Ps30 + phi + NRf + NRc+ BPR + htBleed + W31 + W32 + T24 * T30 + T24 * T50 + T24 * P30 + T24 * Nf + T24 * Nc + T24 * Ps30’’’

- A degree 2 for the original features was tested
- To accommodate svr, features were scaled
- PCA to reduce dimensions to 31
- RF performed best at

  30.96, 35.89 on train, test resp.

- SVR was overfitting at

  12.08, 24.31 on train, test resp



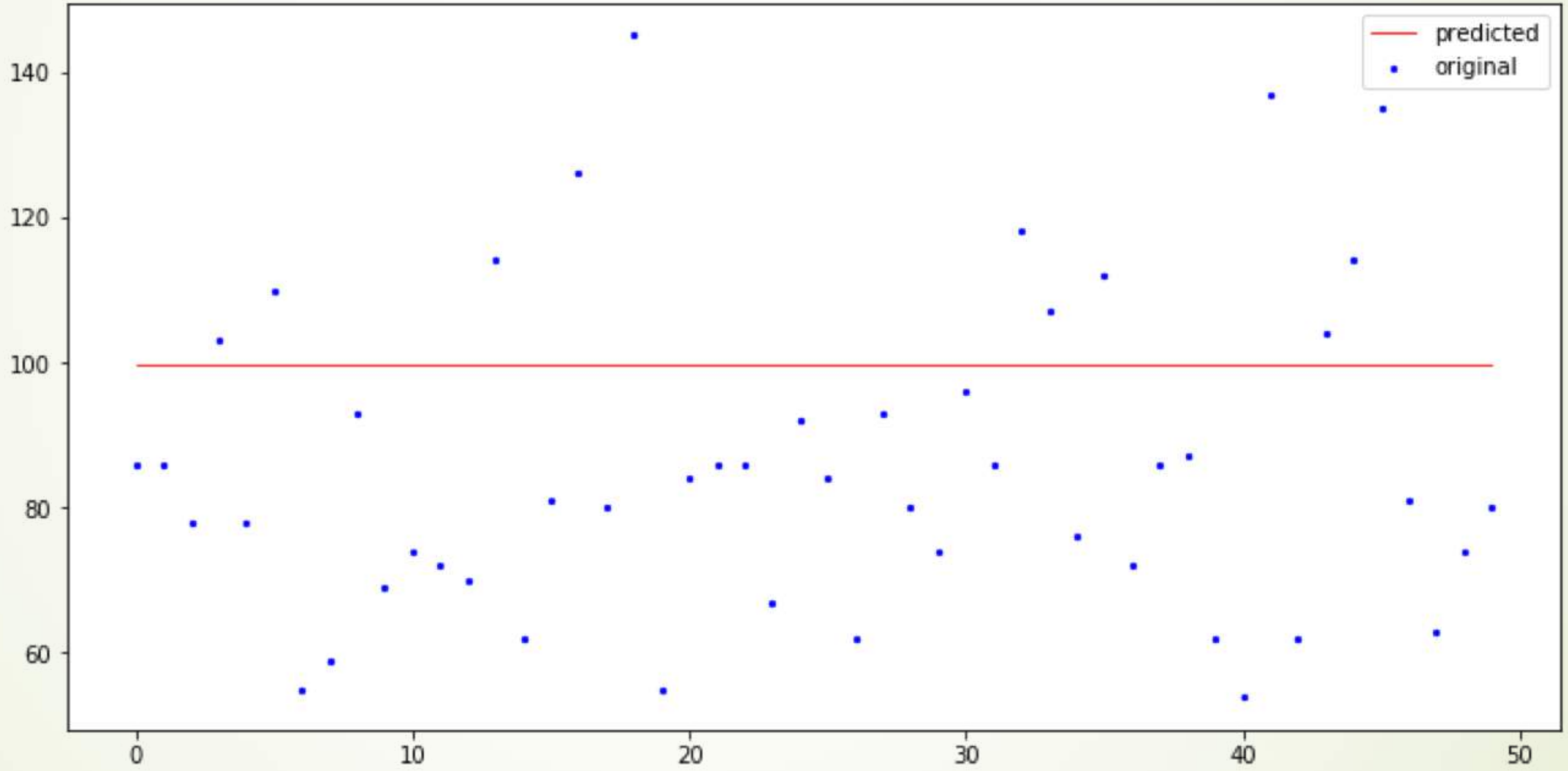
Convolutional Neural Networks

- Convolution filter
- Activation
- Pooling
- Dense
- Simple but not

  very effective
- RMSE: 25.73

```
Model: "sequential_15"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_20 (Conv1D)           (None, 117, 16)           80
_________________________________________________________________
flatten_15 (Flatten)         (None, 1872)              0
_________________________________________________________________
dense_20 (Dense)             (None, 32)                59936
_________________________________________________________________
dense_21 (Dense)             (None, 1)                 33
=================================================================
Total params: 60,049
Trainable params: 60,049
Non-trainable params: 0
```

Simple CNN 1 layer performance



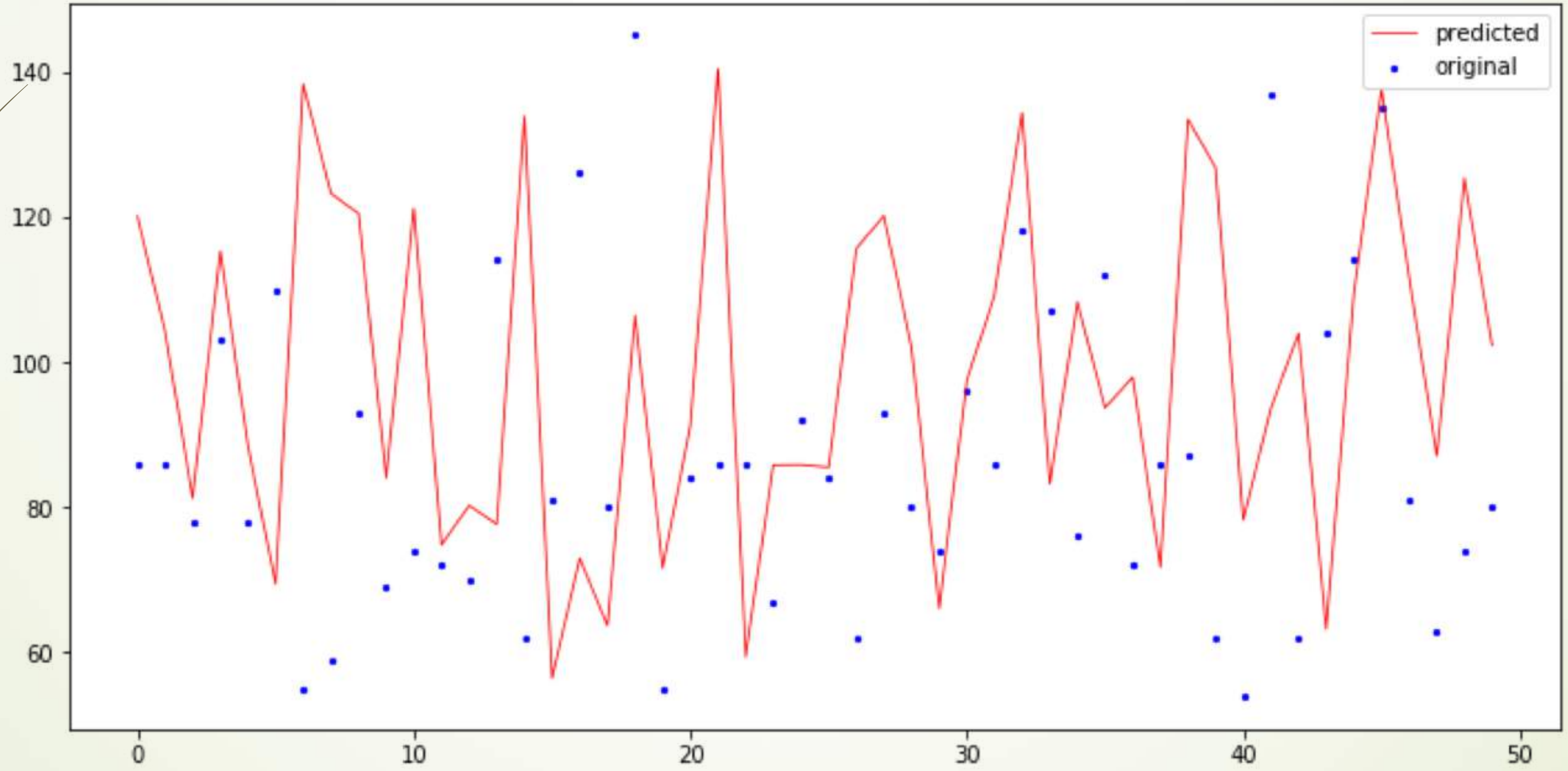
Deeper CNN

- Multiple convolutional layers
- Different activation
- RMSE: 34.217

```
Model: "sequential_13"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_16 (Conv1D)           (None, 117, 16)           80
_________________________________________________________________
average_pooling1d (AveragePo (None, 58, 16)            0
_________________________________________________________________
conv1d_17 (Conv1D)           (None, 58, 8)             136
_________________________________________________________________
leaky_re_lu (LeakyReLU)      (None, 58, 8)             0
_________________________________________________________________
flatten_13 (Flatten)         (None, 464)               0
_________________________________________________________________
dense_16 (Dense)             (None, 32)                14880
_________________________________________________________________
dense_17 (Dense)             (None, 1)                 33
=================================================================
Total params: 15,129
Trainable params: 15,129
Non-trainable params: 0
```

Deeper CNN - Performance

- More complex because of increased layers
- Still not good enough, takes too much time
- RMSE: 34.217



Conclusions

- Multiple models with variety of processing were tried
- Best, Random Forest at around 24 test RMSE and R-squared 79%
- Clipping worked
- Too many features with polynomial features, more training required
- Linear regression was consistent around 35-40% RMSE through out except after interactions
- Couldn’t get the best out of SVM
- Deeper Networks can result in better performance(CNN)

References

- Damage Propagation Modeling for Aircraft Engine Run-to-Failure Simulation, Abhinav Saxena, Kai Goebel, Don Simon, Neil Eklund
- Arias Chao, M.;
  Kulkarni, C.; Goebel, K.; Fink, O. Aircraft Engine Run-to-Failure Dataset Under Real Flight Conditions for Prognostics and Diagnostics. *Data* 2021,*6*,5. https://doi.org/10.3390/ data6010005
- Remaining Useful Life Estimation of Aircraft Engines Using a Modified Similarity and Supporting Vector Machine (SVM) Approach Zhongzhe Chen , Shuchen Cao and Zijian Mao
- Remaining Useful Life Estimation of Aircraft Engines Based on Deep Convolution Neural Network and LightGBM Combination Model. Lijun Liu, Lan Wang, Zhen Yu1
- Kang,Z.;Catal,C.; Tekinerdogan, B. Remaining Useful Life (RUL) Prediction of Equipment in Production Lines Using Artificial Neural Networks. *Sensors* 2021, *21*, 932. https://doi.org/10.3390/ s21030932
- https://towardsdatascience.com/the-importance-of-problem-framing-for-supervised-predictive-maintenance-solutions-cc8646826093
- https://machinelearningmastery.com/improve-deep-learning-performance/
- https://www.machinecurve.com/index.php/2019/11/12/using-leaky-relu-with-keras/
- https://stackoverflow.com/questions/5484922/secondary-axis-with-twinx-how-to-add-to-legend