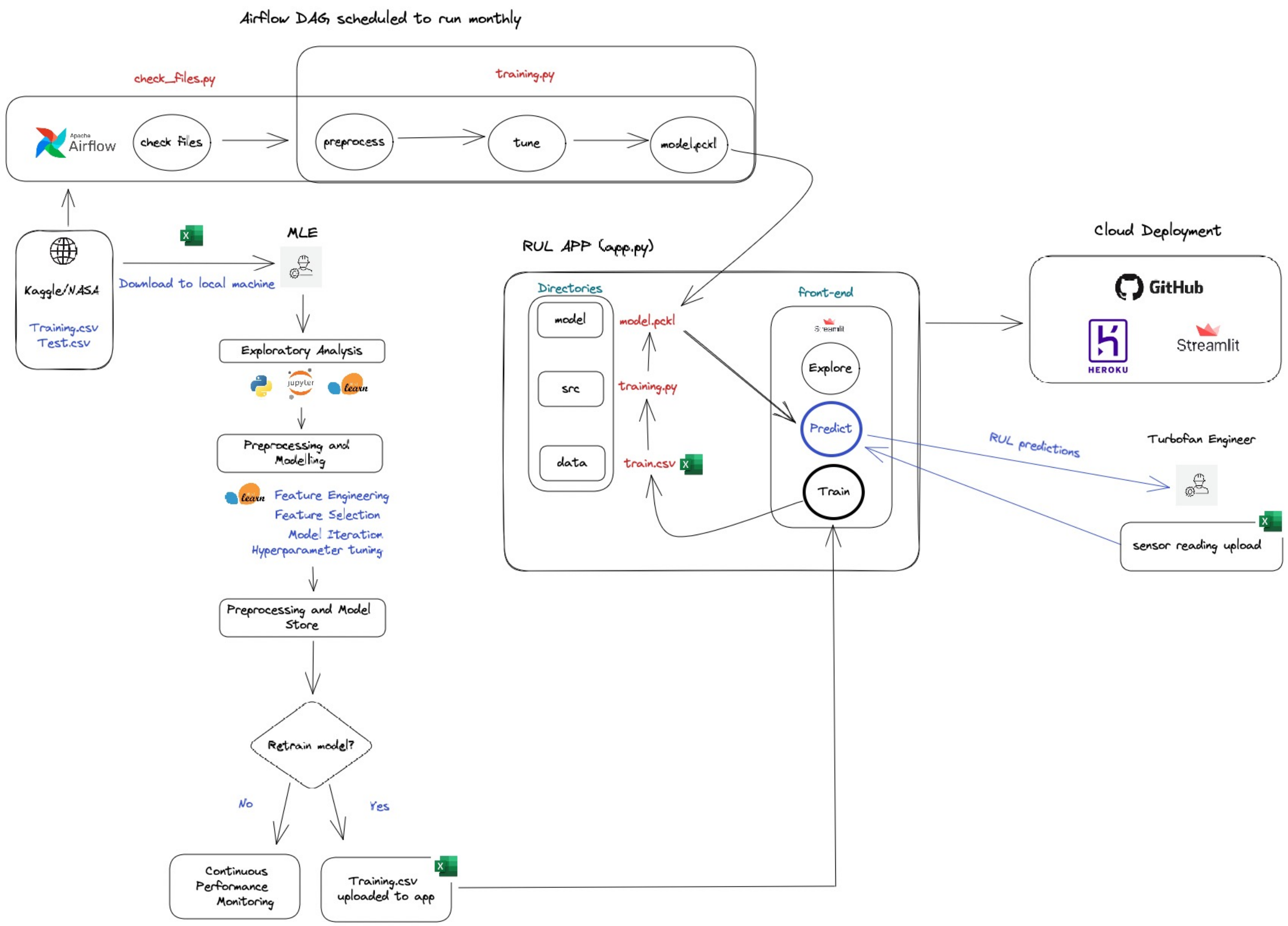
PREDICTING REMAINING USEFUL LIFE OF TURBOFAN ENGINES

Infrastructure Diagram and Model Explainability

INFRASTRUCTURE DIAGRAM



https://excalidraw.com/#json=g9RBQNyOCyXaJiW1QyLsY,kcjMP5_OhWzlYWk45xgQaQ

MODEL EXPLAINABILITY

One drawback of machine learning models is that they are inherently difficult to explain in terms of causality, i.e. what is the theoretical rationale for their parameter outputs in relation to the target response. Our best approach to help turbofan engineers understand predictions made by our model is to provide both a ‘global’ explanation and a ‘local’ one.

GLOBAL

- This helps in understanding how a model makes decisions for the overall structure
- Global interpretation helps in understanding the suitability of the model for deployment
- In our case, how does the SVR model work, how were the hyperparameters calculated and what assumptions did we make in training

LOCAL

- This helps in understanding how the model makes decisions for a single instance
- Using local interpretation we can explain the individual predictions
- In our case, for a single set of sensor readings, what sensors were the most influential in determining the prediction the SVR model made

MODEL EXPLAINABILITY LIBRARIES

SHAPLEY

- SHAP shows the impact of each feature by interpreting the impact of a certain value compared to a baseline value.
- The baseline used for prediction is the average of all the predictions. SHAP values allow us to determine any prediction as a sum of the effects of each feature value.
- SHAP values allow us to determine any prediction as a sum of the effects of each feature value.

ELI-5

- A *permutation importance* method, whereby the model's scoring changes with the feature in existence or not.
- High positive Eli-5 scores mean the feature is of importance relative to other features
- Interpretation of the Eli-5 score with respect to sensor reading impacts on RUL will be easier for maintenance and engineering teams.

MODEL EXPLAINABILITY LIBRARIES: ELI-5

```
perm = PermutationImportance(grid_search.best_estimator_, scoring = 'neg_root_mean_squared_error' ,
                                        random_state=101).fit(X_test_final, rul)
show_weights(perm, feature_names = list(X_test_final.columns))
```

Weight	Feature
1.0769 ± 0.4678	s_9
0.9339 ± 0.4084	s_12
0.8909 ± 0.6048	s_7
0.8679 ± 0.3072	s_2 s_4
0.8281 ± 0.7192	s_2 s_11
0.8232 ± 0.5218	s_11 s_17
0.7860 ± 0.3752	s_4 s_15
0.7320 ± 0.4025	s_11 s_15
0.7289 ± 0.3962	s_11
0.6712 ± 0.4483	s_3 s_11
0.6677 ± 0.2597	s_4
0.6228 ± 0.4584	s_14
0.5231 ± 0.2448	s_4 s_11
0.5041 ± 0.4417	s_21
0.4857 ± 0.4049	s_20
0.3831 ± 0.3075	s_4 s_13
0.3648 ± 0.1629	s_4 s_8
0.3406 ± 0.2097	s_13 s_15
0.3250 ± 0.2330	s_8 s_11
0.2917 ± 0.0916	s_11 s_13
… 26 more …	

SUMMARY

TBD