Prescriptive Algorithms
Assignment 3: Group project

Chigozie Ifepe 2109365

Vishal Shegal 2109374

Pedro Villadangos Benavides 1930079

Borna Djavdan 2007904

Jheronimus Academy of Data Science

1.- Prediction Task

To train the model using the provided training dataset, begin by calculating the Remaining Useful Life (RUL) for each engine. This involves determining the maximum cycle number for each engine and subtracting the current cycle number from it.

Next, select features for the Random Forest model. As Figure 2 illustrates, some features have greater predictive importance than others. Utilize the feature importance scores from the Random Forest algorithm to guide the selection of training features.

It's important to note that the RUL varies across engines, influenced by differences in variable values across cycles.

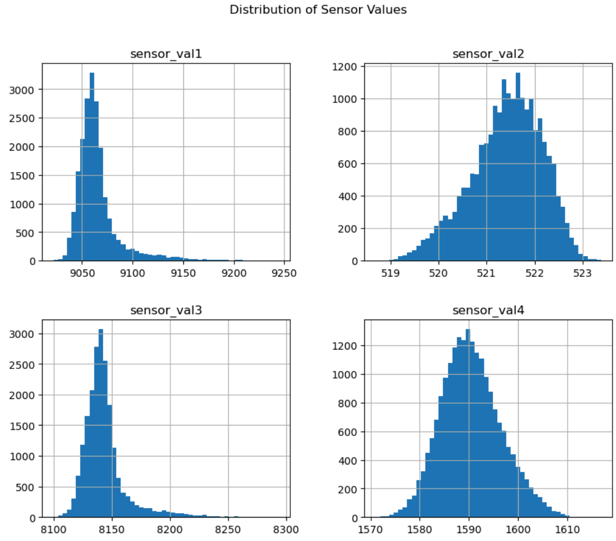

Fig 1. Histogram of sensor values

Feature importance is computed using the Random Forest method. Optimal hyperparameters for the model are identified through grid search, which evaluates various parameter sets and retains those that yield the best model performance.

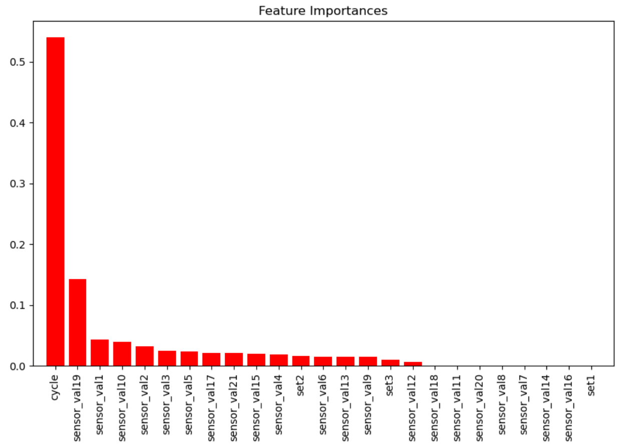

Fig 2. Feature Importance

Model evaluation is conducted using the mean absolute error (MAE), root mean squared error (RMSE), and mean squared error (MSE) metrics. These metrics quantify the difference between the model's predictions and the actual values, providing insight into the model's accuracy.

When comparing the predicted values with those provided by the consultancy company, we observe differences (see Fig 3) that are not statistically significant. It's also notable that the consultancy company's estimates are more conservative, typically assigning shorter RULs compared to our predictions. This conservative approach is understandable, given the critical importance of passenger safety.

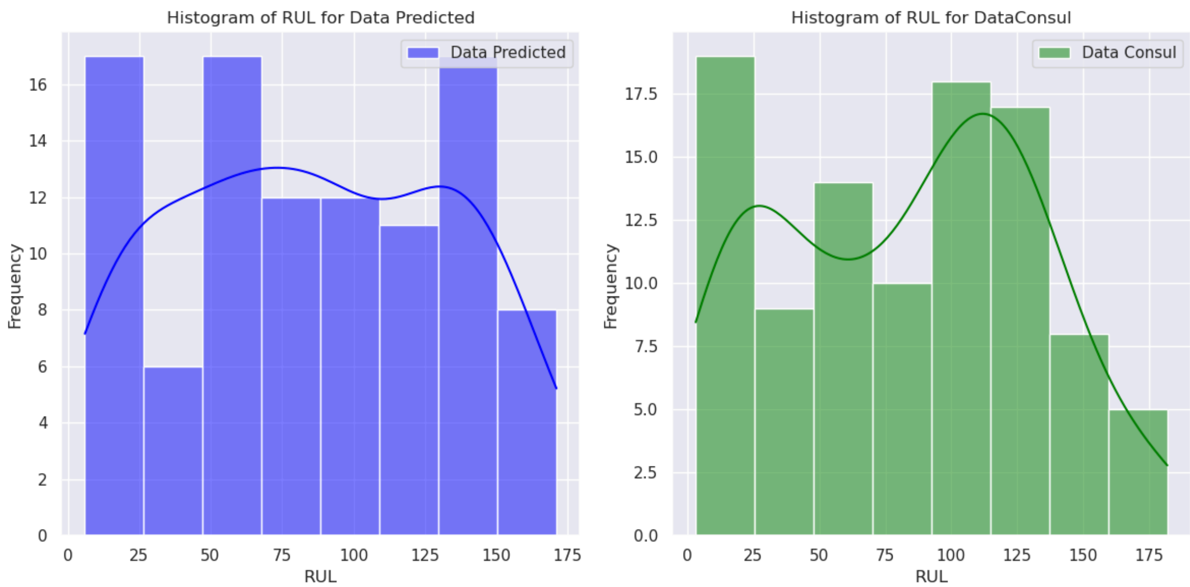

Fig. 3 RUL histogram for predicted values and RUL’s by the consultants

2.- Optimization Task

In our genetic algorithm (GA), each "individual" is essentially a maintenance plan, detailing when and how each engine will be maintained. This plan is divided into parts, each referred to as a "gene," which specifies the engine being serviced, the team (A or B) handling the task, and the exact start and finish days of maintenance. This structure ensures that all necessary details about the engine's maintenance are encapsulated within the schedule.

To refine and improve these schedules, we employ a method known as "tournament selection." Here, we randomly select a handful of schedules and evaluate them based on their effectiveness, a metric we refer to as "fitness." The schedule with the highest fitness progresses to the next round. This method balances retaining proven strategies and integrating new ideas.

To generate new schedules, we use a "custom crossover operation." This process involves cutting one schedule at a random point and filling in the remainder with segments from another schedule, ensuring no engine is double-booked. Additionally, we occasionally adjust the start dates of maintenance through a mutation process, which is carefully controlled to ensure all maintenance still fits within the mandated 30-day period.

However, should an initial schedule prove problematic—such as overlapping team assignments on the same engine or maintenance extending beyond the 30-day limit—we apply corrections. Our algorithm adjusts the schedules to ensure all constraints are met and the plan is viable.

Upon completion, the GA provides a comprehensive list of all engines, detailing which were maintained, by whom, and when. It also calculates the costs incurred from any maintenance delays, providing a total cost for any inefficiencies.

Despite these mechanisms, our current results indicate a persistent issue: the fitness values across different runs remain unchanged, and the calculated penalty costs consistently report as zero. This suggests that the algorithm may be overlooking certain constraints or not correctly applying penalty costs for all engines. Without accurate penalty cost integration, further analysis is not feasible, indicating a critical need to reevaluate the algorithm's penalty assessment components. We must ensure that the algorithm accurately captures and reflects the penalty costs for all engines to maintain both operational efficiency and cost-effectiveness.