Assignment #4

Aircraft Engine Remaining Useful Lifetime Predictions and Maintenance Scheduling with Binary Integer Programming

Chris Papas	Eric Sharma	Nemania Borovits	Yosef Winatmoko
2032263	2032094	2032374	2032342

JM0100 Business Analytics — Group 21
13th May 2019

Notation

In this document, we are using the definition of summation symbol ($\sum$) according to Graham, Knuth, Patashnik and Liu (1989) which state that $\sum_{i=x}^{y} a_i = 0$ if $y < x$.

1 Prediction Task

1.1 Part A

The Remaining Useful Lifetime (RUL) prediction task belongs to a time-series problem. A common approach to model time-series data is construct aggregated features such as kurtosis and skewness. However, we decided to refrain from manually engineer aggregated features due to lack of domain knowledge in the engine maintenance domain. We adopted a time-window approach instead to exploit the local relation of the cycles. Concretely, every observation X_t includes settings and sensors value x from N_{tw} prior cycles

$$X_t = \{x_{t-N_{tw}}, x_{t-N_{tw}+1}, x_{t-N_{tw}+2}, ..., x_{t-1}, x_t\}$$

The task is to model a function $f(X_t) = \widehat{RUL}_t$, where $\widehat{RUL}_t$ is the predicted RUL at time t, such that the difference between $\widehat{RUL}_t$ and the actual RUL is minimal. The process to produce f is elaborated in the following paragraphs.

Dataset Split. The training dataset (*dataset 1*) includes 100 engines, each with multiple cycles. To make sure that our model can generalize well for unseen data, we have to split the dataset into train and hold-out subset. Furthermore, we require separate subsets for parameter tuning, thus we split the hold-out set into validation and test subset. The validation subset will be used to find the best parameter for the model while the test set represent the model performance for unseen data. Because we have enough data to have a separate hold-out set, thus cross-validation is unnecessary.

To construct the subsets, we pick 60 engine IDs randomly for the training subset and 20 for each validation and test subset. The goal of the test subset is to simulate the test dataset *dataset 2*, which consists of 100 unseen engines' sensors and settings value. By separating with engine IDs, we make sure that all engines in the test set does not exist in the training and validation set.

Pre-processing. Before we start modeling f, we conducted several steps of pre-processing. 7 sensors and 1 setting are removed from the features because they have the same value for all cycles of all engines. We ended up with 2 settings and 14 sensors: `setting1`,`setting2`, `s2`, `s3`, `s4`, `s7`, `s8`, `s9`, `s11`, `s12`, `s13`, `s14`, `s15`, `s17`, `s20` and `s21`. Due to different ranges of values among the features, the next step is to perform normalization. Following previous studies (Li, Ding & Sun, 2018; Zheng, Ristovski, Farahat & Gupta, 2017), min-max scaling is chosen to normalize the features value to range [-1, 1]. We make sure that the normalization used for validation and test set is based on the values seen in the training set to prevent overestimating the performance.

In this scenario, engines are labeled as *warning* and *danger* when they have $RUL \leq 40$ and $RUL \leq 25$, respectively. Throughout the optimization tasks, it is more important to correctly predict engines with $RUL \leq 40$, which will be the maintenance candidates. We decided to establish a ceiling for the RUL equal to 125, following previous studies by Li et al. (2018), Zheng et al. (2017). RUL ceiling introduces no harm in this scenario because the maintenance is scheduled only for engines labeled *warning* or *danger*.

Evaluation Criteria. The main metric for model selection is an asymmetric scoring function that penalize late predictions more than early prediction (Saxena, Goebel, Simon & Eklund, 2008). The score p is defined as

$$p = \begin{cases} \sum_{j=1}^{J} e^{-\frac{d}{10}} - 1, & \text{d} < 0 \\ \sum_{j=1}^{J} e^{\frac{d}{13}} - 1, & \text{d} \geq 0 \end{cases}$$

where $d = \widehat{RUL} - RUL$.

The asymmetric scoring function is chosen because it is preferable to maintain an airline engine as early as possible due to safety reason. Consequently, we will get a more conservative model; a model that has more early predictions than late. However, we notice that the scoring function is hard to scale the actual error in days. As an extra consideration, we also incorporate a symmetric measurement Root-Mean-Square-Error (RMSE) during modeling to measure the models' performance. RMSE is easier to interpret in terms of the number of days difference between the predicted and the actual RUL.

Models. We consider three models for the RUL prediction task. First, we use Random Forest (RF) as a baseline model. We picked RF because it is quick to implement and tune. Furthermore, a tree-based model is capable of

handling non-linear modeling, which is the case in this scenario. To tune the RF model, we use the grid search algorithm.

The other two approaches are based on the neural network algorithm. Deep Convolutional Neural Network (DCNN) can capture local relationship between features and learn the representation of a time-series data. By using a 1-dimensional kernels architecture, the network is constrained to consider sensors and settings that is the most recent as also the most relevant to predict the RUL. The architecture that we use for DCNN is similar with Li et al. (2018). It consists of 4 hidden convolutional layers, each with 10 $3x1$ kernel size and zero-padding to have the same dimensions. After the convolutional layers, we have a dropout layer with dropout rate 0.5 followed by fully-connected layer of 100 nodes. Another dropout layer with rate 0.25 moderates the connection from the fully-connected layer to the output layer which consists of 1 node. All layers have `tanh` activation function, except the output layer which does not include one.

The last model we adopted is a type of a sequence architecture called Long-Short Term Memory (LSTM). It is logical to use a sequence-based model for time-series modeling and the RUL decrease linearly from day-to-day. Knowing the RUL of the previous day can help predicting the RUL of the current day. The LSTM architecture that we use is proposed by Zheng et al. (2017) to solve an identical problem. The network consists of 2 hidden LSTM model. We adjusted the number of nodes of the first and the second hidden LSTM layer to 31 and 62 nodes to match the size of the time-window.

After performing a grid search for the RF, we found the following configuration as the best performance for the validation set: N_{tw} = 5, `max_depth` = `None`, `max_features` = 20, `n_estimators` = 600. For the DCNN and LSTM, we use N_{tw} = 30 because the minimum number of cycle in the test dataset is 31. We use learning rate of 0.001 and add decay rate of 0.001 for both DCNN and LSTM. As suggested by the original papers, the optimization algorithm used by DCNN and LSTM are Adam with batch size of 64 and RMSprop with batch size of 1, respectively. Due to the limited computational resource, we can only run 50 epochs for DCNN and 3 epochs for LSTM. Having said that, we noticed that the performance does not improve significantly when we train with more epochs.

Results. The test score of each model can be seen in Table 1. DCNN gives the best score for both the asymmetrics penalty p and $RMSE$. In general, LSTM also performs well while RF is relatively under-performing. Based on the test performance, we picked DCNN as our final model to get the RUL predictions of *dataset 2* engines.

Model	Penalty Scoring	RMSE
Random Forest	19.03	18.56
DCNN	**5.468**	**15.08**
LSTM	7.700	16.62

Table 1: Penalty score p and $RMSE$ for each model with test subset.

1.2 Part B

The RUL prediction for the test dataset use `DataSchedule.txt` as input and the predicted RUL using our final model can be seen in `PT/RUL_predictions.xlsx`. To continue with the Optimization Task, we duplicate the predicted file to `OT1/RUL_predictions.xlsx`.

2 Optimization Task 1

2.1 Part A

Variables. Based on the problem, we define the following variables:

(1) there are $I = \{1, 2, 3, ..., n\}$ teams and $J = \{1, 2, 3, ..., m\}$ engines

(2) the planning time horizon is T and we have $\Theta = \{1, 2, 3, ..., T-1\}$ days of possible maintenance

(3) for each team i, $q_i = \{q_1, q_2, ..., q_n\}$ is the type of team i

(4) for each team i and engine j, $\mu_j^{q_i}$ is the number of days required by team i (of type q_i) to complete maintenance of engine j

(5) for each engine j, R_j is the Remaining Useful Time (RUL) of engine j from current planning day and c_j is the daily penalty cost of engine j

Decision Variables. The decision variables define the start of a maintenance by team i on engine j at day t, given by x_{ijt} which is a binary variable:

$$x_{ijt} = \begin{cases} 1, & \text{if team } i \text{ start maintenance of engine } j \text{ at day } t \\ 0, & \text{otherwise} \end{cases}$$

Objective Function. The objective is to minimize the total penalty cost given by

$$\text{minimize} \sum_{j=1}^{m} \sum_{t=R_j+1}^{T} c_j - \sum_{i=1}^{n} \sum_{j=1}^{m} \sum_{t=1}^{T-1} x_{ijt} s_{ijt}$$

where s_{ijt} represents the possible cost saved if there is a maintenance by team i on engine j starting at day t. For a maintenance by team i on engine j that start at day t, the maintenance will be finished at day $t + \mu_j^{q_i} - 1$ and the total number days saved is $T - (t + \mu_j^{q_i} - 1)$. However, the maintenance can be completed before R_j, thus the number of days saved will be $T - R_j$ because there is no cost saved until day R_j. Consequently, we can calculate s_{ijt} as follows

$$s_{ijt} = (T - \max(t + \mu_j^{q_i} - 1, R_j))c_j \quad \forall i \in I, j \in J, t \in \Theta$$

The value of the total cost given no maintenance conducted $\sum_{j=1}^{m} \sum_{t=R_j+1}^{T} c_j$ will be the same regardless of the decision variables. Therefore, we can minimize the total penalty cost by maximizing the total cost saved by the maintenance, which gives us the following objective function:

$$\text{maximize} \sum_{i=1}^{n} \sum_{j=1}^{m} \sum_{t=1}^{T-1} x_{ijt} s_{ijt}$$

Constraints. At any given day t, each team i can only start maintenance at most 1 engine:

$$\sum_{j=1}^{m} x_{ijt} \leq 1, \quad \forall i \in I, t \in \Theta \tag{1}$$

Every team i can start a maintenance on another engine if they are not working at the last $\mu_j^{q_i}$ days for every engine j:

$$\sum_{j=1}^{m} \sum_{\hat{t}=d_{ijt}}^{t} x_{ij\hat{t}} \leq 1, \quad \forall i \in I, t \in \Theta \tag{2}$$

where d_{ijt} is the starting day of a maintenance by team i on engine j that finish at day t. d_{ijt} can be pre-computed as

$$d_{ijt} = \max(t - \mu_j^{q_i} + 1, 1) \quad \forall i \in I, j \in J, t \in \Theta$$

Every engine j can only be maintained at most by 1 team and 1 time only for the whole time horizon:

$$\sum_{i=1}^{n} \sum_{t=1}^{T-1} x_{ijt} \leq 1, \quad \forall j \in J \tag{3}$$

All maintenance should be finished before T:

$$\sum_{i=1}^{n} \sum_{j=1}^{m} \sum_{t=T-\mu_j^{q_i}+1}^{T-1} x_{ijt} = 0 \tag{4}$$

2.2 Part B

The decision variables, constraints and objective function used are defined in section 2.1. The parameters used in the model are:

(1) $I = \{1, 2, 3, 4\}$, $J = \{1, 2, 3, ..., 100\}$, $q_{i \in I} = \{A, A, B, B\}$ and $T = 25$

(2) The values for $c_{j \in J}$ are as follows: $c_j = 5$ for $j \in \{1, 2, ..., 20\}$, $c_j = 7$ for j $\in \{21, 22, ..., 40\}$, $c_j = 9$ for j $\in \{41, 42, ..., 60\}$, $c_j = 5$ for j $\in \{61, 62, ..., 80\}$, $c_j = 3$ for j $\in \{81, 82, ..., 100\}$

(3) The values of $\mu^A_{j\in J}$ are as follows: $\mu^A_j = 4$ for j $\in \{1, 2, ..., 25\}$, $\mu^A_j = 6$ for j $\in \{26, 27, ..., 50\}$, $\mu^A_j = 3$ for j $\in \{51, 52, ..., 75\}$, $\mu^A_j = 5$ for j $\in \{76, 77, ..., 100\}$

(4) The values of $\mu^B_{j\in J}$ are as follows: $\mu^B_j = \mu^A_j + 1$ for j $\in \{1, 2, ..., 33\}$, $\mu^B_j = \mu^A_j + 2$ for j $\in \{34, 35, ..., 67\}$, $\mu^B_j = \mu^A_j + 1$ for j $\in \{68, 69, ..., 100\}$

(5) The R_j for each engine $j \in J$ are defined in input file `RUL_predictions.xlsx`.

Results. An optimal solution is obtained. The results can be found in `optimization_tasks.ipynb`. Table 2 (Appendix A) also contains a summary of the results. In the results we observe that: total Penalty cost before maintenance is 984, total penalty cost after maintenance is 14 and total penalty cost saved is 970.
Discussion. There is a significant amount of cost saved, the percentage of cost saved is 98.58 %.

2.3 Part C

We solve the problem using the decision variables, constraints and objective function defined in section 2.1. The parameters remain the same as defined in part B of this section, however a different input file is used containing the R_j values for each engine $j \in J$. The file used in this case is `RUL_consultancy_predictions.xlsx`.

Results. An optimal solution is obtained. The results can be found in `optimization_tasks.ipynb`. Table 2 (Appendix A) also contains a summary of the results. In the results we observe that: total Penalty cost before maintenance is 1037, total Penalty cost after maintenance is 46 and total Penalty cost saved is 991.
Discussion. 16 out of 19 engines are maintained. This is because the optimizer prioritizes engine maintenance of highest possible saved cost by lower RUL, which in return has higher cost, and the three engines which are not maintained had RUL equal to 25. We also observe that one less engine, id $= 64$, is maintained in this case compared to part B.

2.4 Part D

Both the tasks in Part A and Part B (Optimization Task 1) are repeated but now with $T = 40$. The results for these tasks can be found in `optimization_tasks.ipynb`. Table 2 (Appendix A) also contains a summary of the results.

Comparing predicted RUL with T=25 to T=40. All engines are maintained in both cases (17 and 27 respectively). The percentage of saved cost in case of $T = 40$ is 98.13 % which means it is almost the same as when $T = 25$. Thus, the percentage of the saved cost depends more on the actual prediction of the RUL and not from time horizon. Another interesting finding is that when $T = 40$ all engines with RUL less than or equal to 25 are maintained before $t = 25$. This means that when enough resources are available the engine maintenance prioritization is done as similar to optimization task 1 (part B). We notice the same pattern for consultant's RUL as the predicted RUL regarding the maintained engines and the saved cost.

Comparing predicted RUL with consultancy RUL when T=40. The consultancy RUL includes an additional engine when compared to predicted RUL regarding to engine maintenance schedule. However, the difference in the saved costs between predicted RUL and consultancy RUL is almost the same in both cases regardless the time horizon.

3 Optimization Task 2

3.1 Part A

Additional Variables. In addition to the variables defined in Optimization Task 1, we have:

(6) for each team i of type q_i, the total number of engine they can work on during the planning horizon T is $k^{q_i}_T$

Additional Constraints. In addition to the constraints defined in Optimization Task 1, we have:

$$\sum_{j=1}^{m} \sum_{t=1}^{T-1} x_{ijt} \leq k^{q_i}_T, \quad \forall i \in I \tag{5}$$

3.2 Part B

We solve the problem using the decision variables, constraints and objective function defined in section 2.1. We also take into account the additional variables and constraints stated in section 3.1. The parameters used are the same as stated in Optimization Task 1, Part B.
Additional parameters, $k_T^A = 2$, where $T = 25$ and $k_T^B = 2$, where $T = 25$
Results. The results can be found in `optimization_tasks.ipynb`. Table 2 (Appendix A) also contains a summary of the results.

An optimal solution is obtained. In the results we observe that:

(1) 8 engines with $R_j \leq T$ are maintained.

(2) Team 1 maintains 2 engines, Team 2 maintains 2 engines, Team 2 maintains 3 engines and Team 4 maintains 2 engines.

(3) Total Penalty cost before maintenance: 984
Total Penalty cost after maintenance: 293
Total Penalty cost saved: 691

Discussion. Although the number of maintained engines dropped from 17 to 8, the percentage of the saved cost decreased only by 28%.

3.3 Part C

We solve the problem same as Optimization Task 2 (part B), however a different input file is used containing the R_j values for each engine $j \in J$. The file used in this case is `RUL_consultancy_predictions.xlsx`.
Results. The results can be found in `optimization_tasks.ipynb`. Table 2 (Appendix A) also contains a summary of the results.

An optimal solution is obtained. In the results we observe that:

(1) 8 engines with $R_j \leq T$ are maintained.

(2) Team 1 maintains 2 engines, Team 2 maintains 2 engines, Team 3 maintains 2 engines and Team 4 maintains 2 engines.

(3) Total Penalty cost before maintenance: 1037
Total Penalty cost after maintenance: 227
Total Penalty cost saved: 810

Discussion. The number of the maintained engines dropped from 16 to 8. The percentage of the saved cost decreased by 17%. This means that in contrast to OT1 where the percentage of the saved cost was higher from our prediction comparing to consultancy, in the case of Optimization Task 2 the percentage of saved cost from consultancy is higher. The latter phenomenon can be explained from the fact that the RUL from our prediction is higher which means that the total saved cost is lower. This can be explained from the fact that the actual examined engines are almost identical (7 out of 8). As a result, although we built our model conservatively the consultancy has probably built it even more conservatively.

4 Optimization Task 3

4.1 Part A

Additional Variables. In addition to the variables defined in Optimization Task 1 and 2, we have:

(7) for each team i of type q_i, team i has to wait for w^{q_i} days before starting maintenance on another engine

Additional Constraints. We keep constraints from Optimization Task 1 and 2, but we replace d_{ijt} in constraint (2) to take into account the waiting time:

$$d_{ijt} = \max(t - \mu_j^{q_i} - w^{q_i} + 1, 1) \quad \forall i \in I, j \in J, t \in \Theta$$

4.2 Part B

We extend Optimization Task 2 (Part B) with the constraints and variables stated in section 4.1. The parameter values remain the same as Optimization Task 2 (Part B). Additionally, $w^A = 1$ and $w^B = 2$.
Results. The results can be found in `optimization_tasks.ipynb`. Table 2 (Appendix A) also contains a summary of the results.

An optimal solution is obtained. In the results we observe that:

(1) 8 engines with $R_j \leq T$ are maintained.

(2) Team 1 maintains 2 engines, Team 2 maintains 2 engines, Team 2 maintains 2 engines and Team 4 maintains 2 engines.

(3) Total Penalty cost before maintenance: 984
Total Penalty cost after maintenance: 293
Total Penalty cost saved: 691

Discussion. Comparing to optimization task 2 (part B), we see that the results are the same. This is because there is enough time to incorporate the waiting constraints.

4.3 Part C

We solve the problem same as Optimization Task 3 (part B), however a different input file is used containing the R_j values for each engine $j \in J$. The file used in this case is `RUL_consultancy_predictions.xlsx`.
Results. The results can be found in `optimization_tasks.ipynb`. Table 2 (Appendix A) also contains a summary of the results.

An optimal solution is obtained. In the results we observe that:

(1) Out of 19 engines with $R_j \leq T$, a total of 8 engines are maintained.

(2) Team 1 maintains 2 engines, Team 2 maintains 2 engines, Team 3 maintains 2 engines and Team 4 maintains 2 engines.

(3) Total Penalty cost before maintenance: 1037
Total Penalty cost after maintenance: 248
Total Penalty cost saved: 789

Discussion. The results of optimization task 2 (part B) and optimization task 3 (part B) were identical, however in the case of comparing optimization task 2(part C)and optimization task 3(part C), we observe that the percentage of cost saved in task 3 is lower. Due to this difference, we can conclude that the consultancy RUL is lower than predicted RUL, as a result of which, in the case of predicted RUL there is enough time to incorporate the waiting constraints.

5 Optimization Task 4

Main findings. Based on the previous sections, we conclude that all of engines with RUL $\leq T$ are maintained when the max engine constraint is not included. This indicates that both time horizons (25,40) are sufficiently large. When the max engine constraint is introduced engines with higher cost are prioritized. There is an observed difference between scheduling using our prediction model and the consultant's. In particular, we concluded that the consultant's model is more conservative than ours, because it tends to predict lower RUL. Moreover, introducing waiting time does not affect our model. However, we observe a decrease in the percentage of the total saved cost from the consultant's.
Managerial insights: The findings of this assignment take into account only the saved cost from maintenance but not the labor cost. In the real world managers have to consider the trade-off between these two types of cost. As demonstrated in Optimization Task 2, even though teams maintained less than 50 % of the engines, more than 70% of the cost was saved. Additionally, managers can consider adding waiting time because it does not significantly affect the result.
Hard and easy aspects of the problem: From the beginning of this assignment, we understood that we must carefully think about every aspect during the problem formulation. The formulation of the objective and constraints had to be created with fine treatment. As a team we put a lot of effort in order to create each of them,

however the constraint that was the most difficult to think about was the working-time constraint. More specifically, we wanted to find how to fit in an equation the constraint that "every team i that starts working on engine j has to finish μ days before starting maintenance". Furthermore, we had to take into account that every team type needs different number of days to maintain each individual engine. For these parts, which we felt were the most difficult, team work really helped in formulating the solution.

On the other hand, when we realized that the difficult constraint could be modelled with decision variables of when a team start a maintenance, the rest of the formulation became trivial. Once we were finished with the formulation, implementation with `PuLP` could be done easily.

References

Graham, R. L., Knuth, D. E., Patashnik, O. & Liu, S. (1989). Concrete mathematics: A foundation for computer science. *Computers in Physics*, *3*(5), 106–107.

Li, X., Ding, Q. & Sun, J.-Q. (2018). Remaining useful life estimation in prognostics using deep convolution neural networks. *Reliability Engineering & System Safety*, *172*, 1–11.

Saxena, A., Goebel, K., Simon, D. & Eklund, N. (2008). Damage propagation modeling for aircraft engine run-to-failure simulation. In *2008 international conference on prognostics and health management* (pp. 1–9). IEEE.

Zheng, S., Ristovski, K., Farahat, A. & Gupta, C. (2017). Long short-term memory network for remaining useful life estimation. In *2017 ieee international conference on prognostics and health management (icphm)* (pp. 88–95). IEEE.

Appendix A

A summary of all the results can be found in Table 2.

	1(B)	1(C)	$1(D)_a$	$1(D)_b$	2(B)	2(C)	3(B)	3(C)
Total cost before maintenance	984	1037	3042	3166	984	1037	984	1037
Total Cost Saved	970	991	2985	3056	691	810	691	789
Remaining Cost	14	46	57	110	293	227	293	248
# projects by Team 1	6	5	9	10	2	2	2	2
# projects by Team 2	5	4	7	7	2	2	2	2
# projects by Team 3	3	3	5	5	2	2	2	2
# projects by Team 4	3	4	6	6	2	2	2	2
#Engines with RUL <= T	17	19	27	28	17	19	17	19
#Engines with RUL = T	0	3	0	0	0	3	0	3
#Engine maintained	17	16	27	28	8	8	8	8
% cost saved	98.58	95.56	98.13	96.53	70.22	78.11	70.22	76.08

Table 2: Summary of the results of the optimization tasks