Cranfield University

Statistical Learning for Predictive Maintenance

Student :
Jean Eudes KONAIN
Student No: 451122
Department : Aerospace
Word count : 2835

Lecturers:
Dr Ivan Petrunin
Dr Adolfo Perrusquia
Dr Zhengjia Xu

November 3, 2024

Contents

List of Figures

List of Tables

1 Introduction

Failure prediction is a major topic in predictive maintenance in many industries. Aircraft manufacturers, OEMs and end users are highly interested in prediction of component failures during the operation so that they can plan maintenance operations and reduce losses due to the time aircraft has spent at the ground.

Monitoring of the engine health and current condition is based on sensor data analysis and telemetry from the engine sub-systems. It is supposed to promote predictive maintenance by estimating either Time-To-Failure (TTF) or Remaining Useful Life (RUL) for aircraft components that are currently in-service and may be fully functional at the time of testing.

Based on the measurements from the sensors of the aircraft engine, the developed analysis framework should provide the following predictions, which are the objectives of this assignment:

- Time-To-Failure (TTF) prediction for the engine
- Classify which engine will fail in the analysed time period

The goal of the work is to propose, implement and discuss solutions aiming to enhance the maintenance operations and planning of time-based preventive maintenance of the aircraft engine by selecting and applying statistical learning methods concerning the prediction of the remaining life and the prediction of current status of the engines.

2 Regression Task: Predicting Remaining Cycles

2.1 Data visualization and choice of model parameters

2.1.1 Data visualization

Data visualization is an important step in making decisions about which machine learning model to use and choosing which variables have a significant impact on the model. According to Vitaly Friedman (2008) [1], data visualization allows "to communicate information clearly and effectively through graphical means" while "providing insights into a rather sparse and complex data set by communicating its key aspects in a more intuitive way."

So, to address the regression problem, I visualized the data to know:

- Whether there was a correlation between the variables and which ones met the requirements of the problem
- The type of regression model to use (linear or non-linear)

Figure 1 below shows the measurements of the sensors (s1, s2, s3, s4) as a function of the cycle. This is not representative of the problem posed because the cycle does not represent the time to failure of aircraft engines.

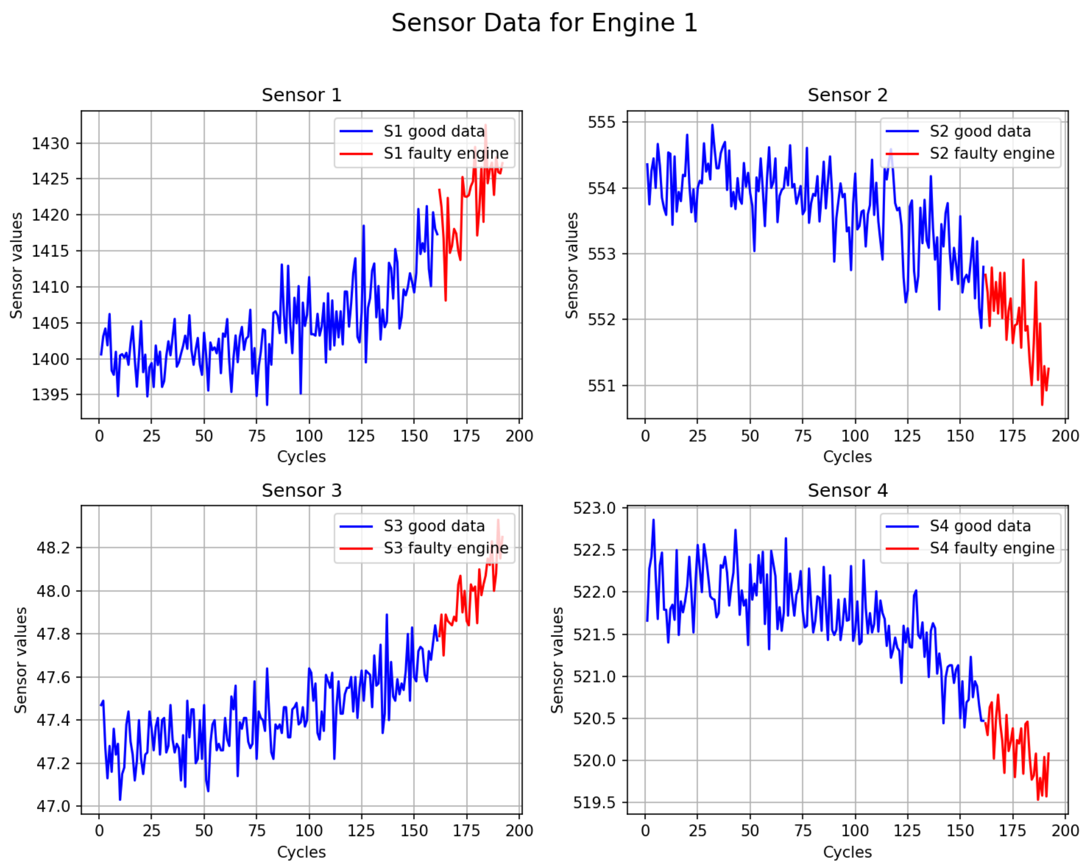

Figure 1: This graph shows the 4 simulated sensor measurements for aircraft engine 1 as a function of cycles.

Then, to understand the relationship between Time To Failure and the cycle, I drew the TTF = f(cycle) curve (Figure 2) and obtained a decreasing line that reflects a strong correlation between these two data. We can conclude that the more the number of cycles increases, the more the time to Failure decreases. This variable reflects the expected behavior.

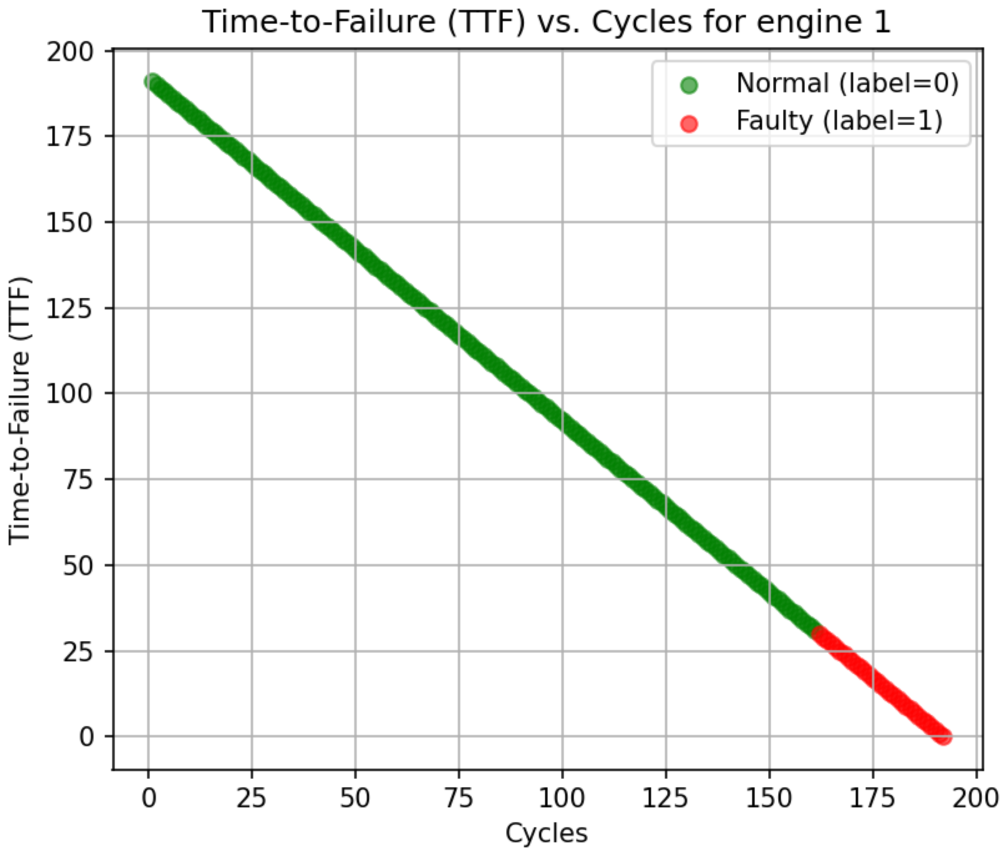

Figure 2: This graph depicts the Time to failure as a function of cycles.

Finally, I displayed the TTF according to the sensors (s1, s2, s3, s4) (Figure 3) and I retained the following variables: ttf, s1, s2, s3, s4. Subsequently, I made the following assumptions:

- Do I have to use the 4 sensors to train the model? Or 1 alone? Why?
- Will this impact the performance of the model?
- Do you have to average the 4 sensors? Is this representative of real data?

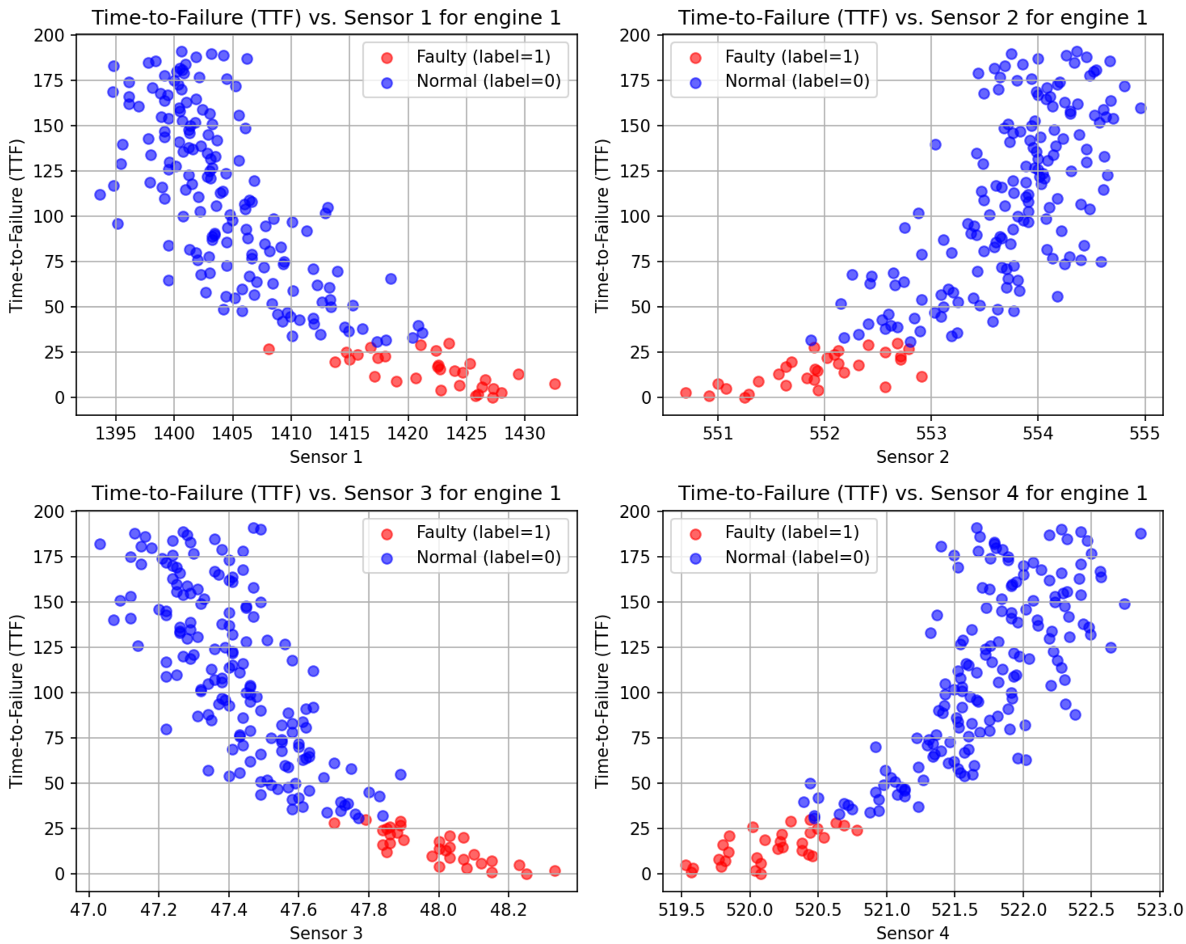

Figure 3: This graph shows the 4 simulated sensor measurements for aircraft engine 1 as a function of Time To Failure.

2.1.2 Choosing Model Setting

To address the questions stated above, I calculated the correlation between each variable (s1, s2, s3, s4 and ttf) and obtained the result below (Figure 4).

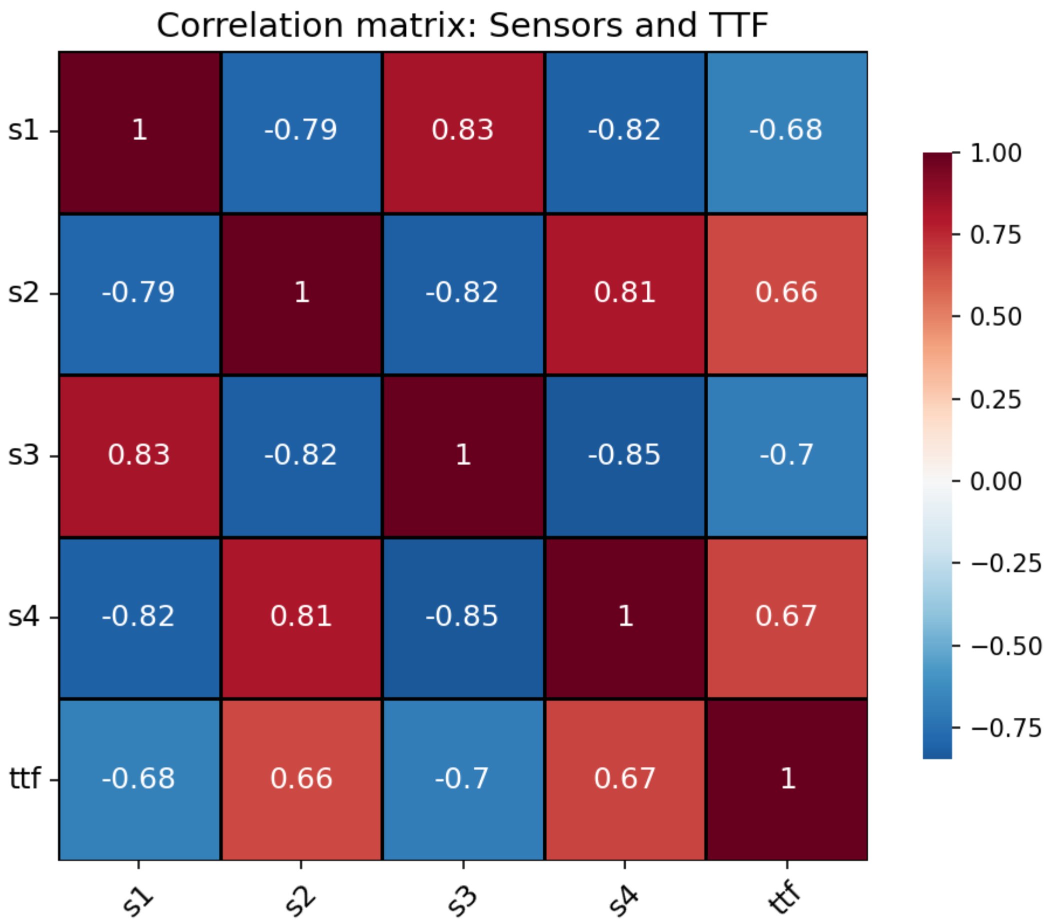

Figure 4: The matrix shows the pair-wise correlations between sensors and Time To Failure, with warmer colors indicating stronger positive correlations and cooler colors indicating stronger negative correlations.

We notice that:

- s1 and s3 have a negative correlation with the TTF. This means that when s1 and s3 increase, the Time To Failure decreases. So the s1 and s3 sensors are associated with engine degradation.
- s2 and s4 have a positive correlation with the TTF. This means that when s2 and s4 increase, the Time To Failure increases so the engine stays in good condition longer.

Using a single sensor means losing or neglecting a large part of the information available in the training dataset. The same is true for the average of the 4 sensors. In conclusion, it makes sense to use all 4 sensors (s1, s2, s3, s4) as the input variable for the regression model because each sensor measures slightly different aspects of an engine's degradation process.

2.2 Model Selection

2.2.1 Description of Method

Graphically, the relationship between sensor measurements and TTF is not linear (Figure 3). Therefore, the choice of the linear regression model is to be avoided. Instead, a non-linear regression model should be used. To justify my approach, I relied on work carried out by Adryan Fitra Azyus and Sastra Kusuma Wijaya (2022) [2] whose aim was "to look for the method and technique of ML, which is the best applied on PdM for aircraft in accuracy indicators". They compared 3 techniques: 2 Machine Learning methods (SVM and Random Forest) and a Deep Learning method (LSTM). In the rest of this project, I will compare the two Machine Learning methods (SVM and Random Forest).

Brief explanation of SVM

"Support vector machines are the kernel-based learning methods that implement the structure risk minimization principle and separate the classes of instances by building and maximizing margin hyper-plane" [3]. SVM is divided into two main categories: support vector classification (SVC) and support vector regression (SVR). According to Benkedjouh et al. (2013) [4], "the main objective of SVR is to estimate a functional relation between input and output random variables under the assumption that the joint distribution P of the input and output variables is completely unknown. The model created by SVR depends only on a subset of the training data, because the cost function for the model construction ignores all training data that are close within a threshold to the model prediction".

Brief explanation of Random Forest

Random Forest is a machine learning algorithm based on decision trees. According to Ho, Tin Kam (1995) [5], the concept is to train the same model several times or instances on the same dataset. Each training session will select a different subset of the data, and the outcome will be decided by a majority vote. For regression tasks, the output is the average of the predictions of the trees and for classification tasks, the output of the random forest is the class selected by most trees. More trees are associated with higher accuracy.

2.2.2 Data Splitting

In order to evaluate and improve the performance of the different models (SVR and Random Forest Regressor), I separated the data from the 'train_selected.csv' file into training data and validation data. This step is essential because you should never evaluate the performance of a model on the data used to train it. Figure 5 below shows the actual distribution. Of the 20631 samples, 80% of the data were randomly divided into a training set and 20% into a validation set.

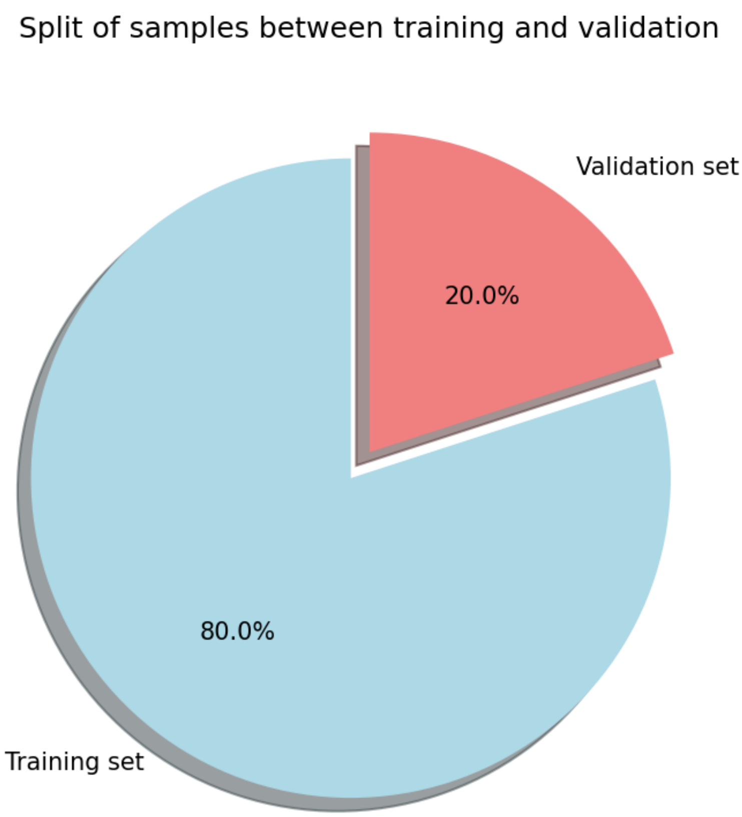

Figure 5: Split of sample between training and validation.

2.2.3 Data pre-processing

Normalization is essential for SVMs because it ensures that all features have the same scale, allowing the algorithm to calculate distances consistently and efficiently maximize the margin between classes. Without normalization, features with large values would further influence the model, skewing the results. In addition, SVM kernels are sensitive to scale differences, and normalization facilitates optimization convergence, improving model accuracy and performance [6][7]. To normalize the data, I used Scikit-Learn's StandardScaler method. This function normalizes the characteristics by centering the data on the mean of 0 and a standard deviation of 1 [7] (Table 1).

$$X_{normalized} = \frac{X - \mu}{\sigma} \tag{1}$$

μ: The average of the feature
σ: The Standard Deviation of the Feature

	s1	s2	s3	s4
Non-normalized data	1400.6	554.36	47.47	521.66
Normalized data	-0.932	1.125	-0.272	0.345

Table 1: The table showing the result of the normalization of the data from Engine 1 to Cycle 1

2.3 Model Fitting

2.3.1 Training the Model

First, I trained the two models (SVR and Random Forest Regressor) with the default hyperparameters in order to get a first estimate of their performance.

Then, to improve the performance of the models, I used GridSearchCV, a function of Scikit-Learn, to test different combinations of hyperparameters and select the one that offered the best results [8].

In the case of SVR, the hyperparameter grid tested included values for:

- **C** : A regularization parameter that controls the margin of error between the predictions and the actual values.
- **Epsilon** : A parameter for determining the required accuracy of the prediction.
- **Gamma** : A parameter influencing the distribution of drive points for nonlinear margins.

And in the case of the Random Forest Regressor, the hyperparameter grid tested included values for:

- **n_estimators** : number of trees in the forest.
- **Criterion** : Loss function to measure the quality of the division.
- **min_samples_split** : The minimum number of samples required to split a node.
- **min_impurity_decrease** : Minimal reduction in impurity for a node to be split.

For each combination of hyperparameters, I applied a 5-fold cross-validation, using the R^2 as an evaluation metric on the validation data to measure the ability of each model to explain the variance of the output data. This method allows for the comparison and selection of optimal parameters, minimizing the risk of overfitting while maximizing the accuracy of the predictions [8][9].

The best hyperparameters, provided the highest R^2 score, are:

- C = 10.0, Epsilon = 0.1, Gamma = 'scale' (SVR)
- min_impurity_decrease = 0.1, min_samples_split = 10, n_estimators = 300, criterion = 'squared_error' (Random Forest Regressor)

2.3.2 Quality of Fit

After optimization, the SVR and Random Forest Regressor models were fitted with the best hyperparameters identified. They were then evaluated on the validation data to measure the quality of their fit and get an overview of their performance. The main metrics used to assess this quality of fit are R^2, MAE and RMSE (Table 2).

	Mean Absolute Error	R^2	Root Mean Square Error
SVR	34.19	0.52	46.66
RFR	34.93	0.52	46.63

Table 2: Regression Evaluation Results

It can be seen that the two SVR and Random Forest Regressor (RFR) models have very similar performances. The SVR model achieves a slightly lower MAE (34.19) compared to that of the Random Forest Regressor (34.93), suggesting that the SVR is slightly better at minimizing the mean deviation between predictions and actual values. This result is confirmed through cross-validation. On average, the MAE of the SVR is lower than that of the Random Forest Regressor, confirming the slight advantage of the SVR in medium accuracy (Tables 3 and 4).

	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Average
RMSE	46.60	47.13	47.05	47.91	47.32	47.20
MAE	33.84	34.46	34.18	34.29	34.52	34.26

Table 3: Outcome of cross-validation in the case of the SVR model

	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Average
RMSE	46.45	47.19	47.03	47.55	47.08	47.06
MAE	34.51	35.34	34.99	34.87	35.02	34.95

Table 4: Outcome of cross-validation in the case of the Random Forest Regressor model

Overall, although the two models are comparable, the SVR seems to have a slight advantage in terms of average accuracy (MAE) but the difference is still minimal.

2.4 Prediction and Discussion

After getting the best model, I used this one to predict the Time To Failure on the test data. The graph below (Figure 6) illustrates a comparison between the model predictions (blue curve) and the actual Time to Failure (TTF) values (red curve) on the test data. Predictions broadly follow the actual trend, but discrepancies persist, especially in some areas.

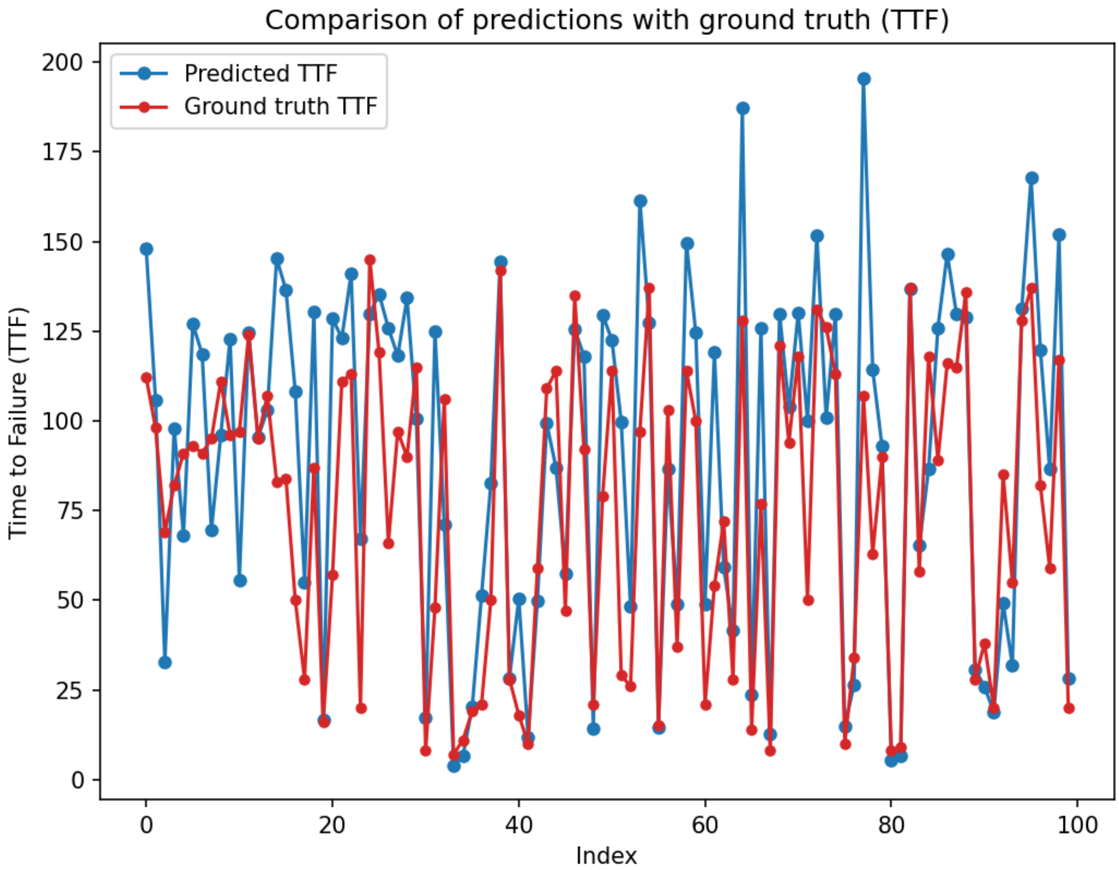

Figure 6: Comparison between predicted TTF and Ground Truth using SVR model.

In short, the predictions are close to the real values in some areas with a prediction difference of 24 cycles. However, the model also shows notable errors (RMSE: 31.5 cycles) (Table 5).

	Mean Absolute Error	R^2	Root Mean Square Error
SVR	24.02	0.42	31.49

Table 5: Outcome of the model after the prediction

3 Classification Task: Faulty Engine Detection

3.1 Data visualization and choice of model parameters

3.1.1 Data visualization

In this part, the goal is to implement a binary classification method that will classify whether or not the engine can be considered as faulty within the current operation cycle. To get a brief overview of the data, I have displayed the Time To Failure according to the label (Figure 7).

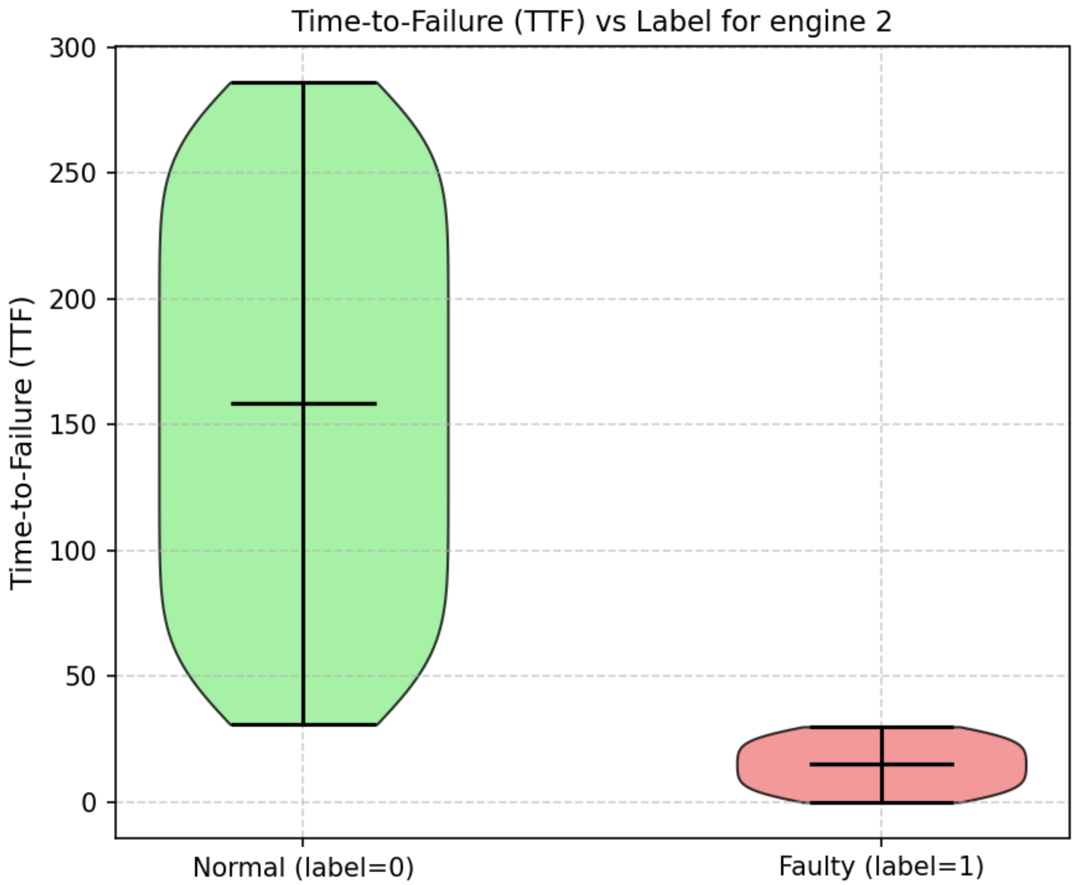

Figure 7: This graph depicts the Time to failure as a function of Labels.

This graph shows the distribution of Time-to-Failure for Engine 2 based on its operating status, either Normal (label=0) or Faulty (label=1). In normal condition, the TTF is significantly higher with a large dispersion, indicating a potentially long and variable service life. On the other hand, when the engine is in a faulty state, the TTF is much lower and concentrated around low values, meaning that failure is imminent.

Subsequently, I assumed : Is the use of Time To Failure a valid approach to classify the operating state of the motor, or is it necessary to include the data from the 4 sensors as well?

3.1.2 Choosing Model Settings

To address this question stated above, I calculated the correlation between each variable (s1, s2, s3, s4, ttf and label_bnc) and obtained the result below (Figure 8).

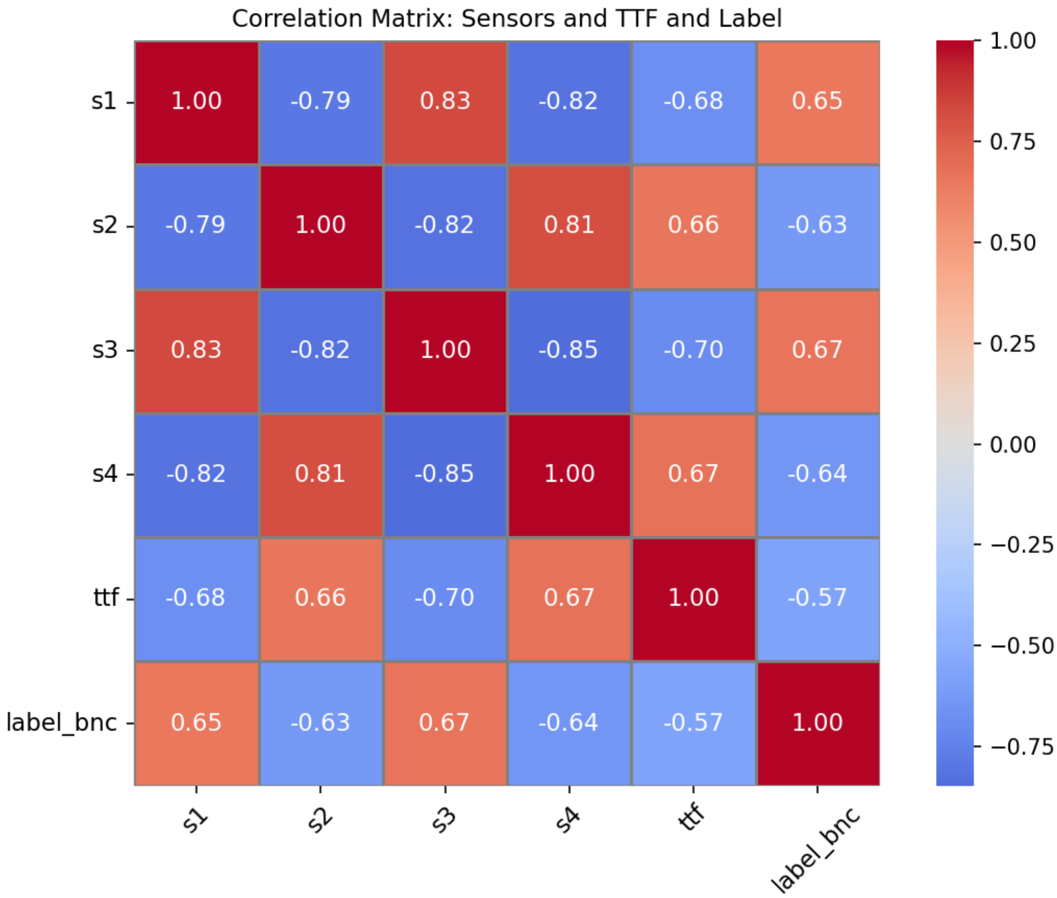

Figure 8: The matrix shows the pair-wise correlations between sensors, Time To Failure and labels, with warmer colors indicating stronger positive correlations and cooler colors indicating stronger negative correlations.

Note that the s1 (0.65) and s3 (0.67) sensors show a positive correlation with the fault label, while s2 (-0.63), s4 (-0.64), and TTF (-0.57) have a negative correlation. Thus, integrating the TTF with all four sensors could strengthen the classification by exploiting the additional information of each sensor on the risk of failure.

3.2 Model Selection

3.2.1 Description of Methods

As stated in section 2.2.1, my approach is based on work by Adryan Fitra Azyus and Sastra Kusuma Wijaya (2022)[2]. But instead of using Regression models, I'm going to compare two classification models : SVC and Random Forest Classifier (RFC)

3.2.2 Data Preparation

Once the two models (SVC and Random Forest Classifier) were selected, I separated the data into training data and validation data with the following respective proportions:

- 80% for training data
- 20% for validation data

Next, I normalized the data so that all features were on the same scale (Table 6)

	s1	s2	s3	s4	ttf
Non-normalized data	1400.6	554.36	47.47	521.66	191
normalized data	-0.932	1.125	-0.272	0.345	1.205

Table 6: The table showing the result of the normalization of data from Engine 1 to Cycle 1

3.3 Model Fitting

3.3.1 Training the Model

First, I trained the SVC and Random Forest Classifier models with their default hyperparameters to get an initial estimate of their performance. Then, I tweaked these models by adjusting the hyperparameters below to maximize the accuracy of the predictions.

- **C** : a regularization parameter controlling the margin of error between predictions and actual values (SVC)
- **n_estimators** : number of trees in the forest. (Random Forest Classifier)

The best hyperparameters selected are:

- C = 10.0 for the SVC model
- n_estimators = 200 for the Random Forest Classifier model

3.3.2 Evaluation on Training Set

After training, the SVC and Random Forest Classifier models were then evaluated on the validation data in order to measure the quality of their fit and to obtain an overview of their performance. The main metrics used to assess this quality of fit are Accuracy, Precision, and Recall (Table 7).

	Accuracy	Precision	Recall
SVC	0.996	0.986	0.988
RFC	1.0	1.0	1.0

Table 7: Classification Evaluation Results

We can see that the Random Forest Classifier obtains perfect scores in accuracy, precision and recall while the SVC obtains quite reasonable results. This means that the Random Forest Classifier model has perfectly classified all samples in the validation set. However, these high scores are often a sign of overfitting, indicating that the model might be over adjusted to the training data. This result is confirmed through cross-validation with an average accuracy of 1.0 for the RFC model, a result slightly higher than that of the SVC model (0.99) (Tables 8 and 9).

	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Average
Accuracy	0.996	0.998	0.997	0.997	0.998	0.997
Precision	0.988	0.996	0.990	0.990	0.994	0.991
Recall	0.986	0.994	0.992	0.994	0.998	0.992

Table 8: Outcome of cross-validation in the case of the SVC model

	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Average
Accuracy	1.0	1.0	1.0	1.0	1.0	1.0
Precision	1.0	1.0	1.0	1.0	1.0	1.0
Recall	1.0	1.0	1.0	1.0	1.0	1.0

Table 9: Outcome of cross-validation in the case of the Random Forest Classifier model

3.4 Classification Results and Discussion

Finally, I predicted the operating state of the engine. Since the test set does not contain the TTF data, I retrieved the results from the best regression model in the first part. To evaluate my models (Table 10), I assumed that all values above 30 in the ground truth TTF have a normal functioning (label = 0) if not a defective functioning (label = 1). This allowed me to obtain the ground truth label.

	Accuracy	Precision	Recall
SVC	0.90	0.89	0.68
RFC	0.89	0.88	0.64

Table 10: Outcome of the model after the prediction

In addition, the confusion matrix (Figure 9) shows that the model correctly classified 73 negative samples (0) and 17 positive samples (1), but made 10 errors (8 false negatives and 2 false positives). This indicates good overall accuracy, although errors remain, particularly in false negatives. Thus, the SVC model is a better predictor of engine health.

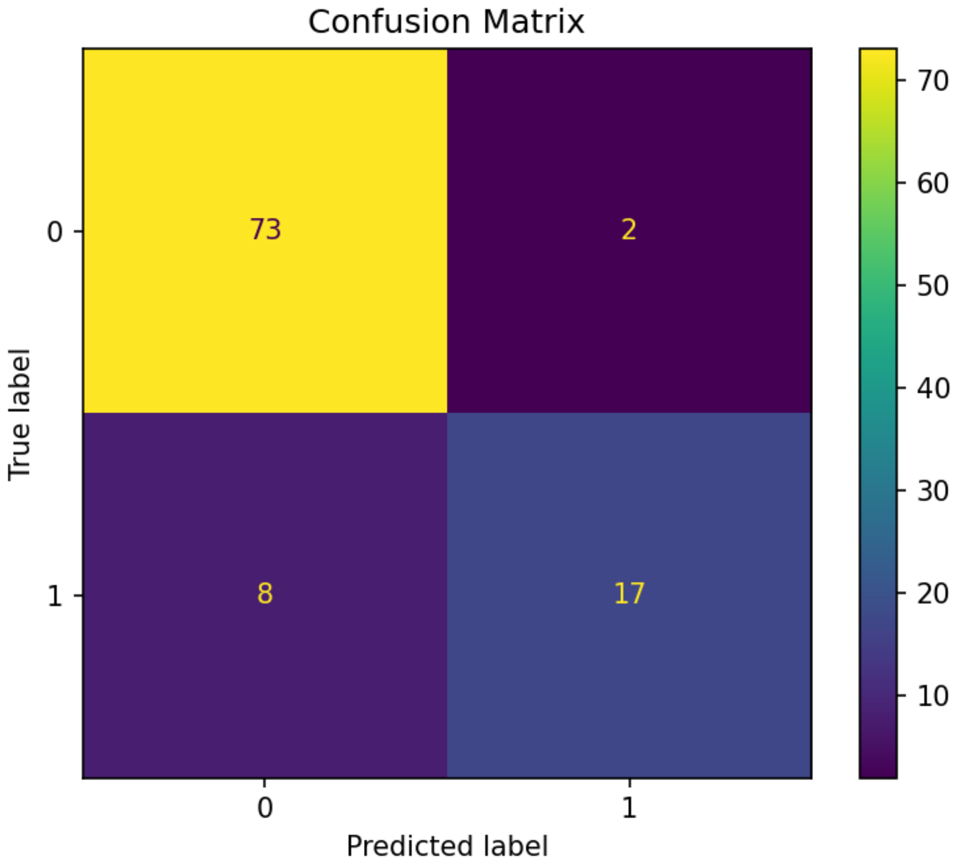

Figure 9: This graph shows the confusion matrix.

4 Conclusion

To sum up, the purpose of this study was to predict the Time To Failure for an engine and to determine whether an engine is normal or faulty. Among the two models chosen (SVM, Random Forest), the Support Vector Machine model proved to be the best model in predicting the remaining number of operational cycles before an engine failure occurs with a relatively small error (24 cycles) and the engine operating status with an accuracy of 90%. As a result, we can easily plan upstream maintenance operations, minimising the risk of in-flight incidents and optimising the efficiency of maintenance operations. This will improve passenger safety and reduce delays and cancellations [10].

5 References

[1] Vitaly Friedman (2008). Data Visualization and Infographics. Smashing Magazine[Internet]; 2008-01-14 [cited 2024-10-29]. Available from https://www.smashingmagazine.com/2008/01/monday-inspiration-data-visualization-and-infographics/

[2] Azyus, A. F., & Wijaya, S. K. Determining the Method of Predictive Maintenance for Aircraft Engine Using Machine Learning. Journal of Computer Science and Technology Studies[Internet], 2022[cited 2024-10-29], 4(1), 01–06. Available from https://doi.org/10.32996/jcsts.2022.4.1.1

[3] Yang, J., Zhang, Y. (2005). Application Research of Support Vector Machines in Condition Trend Prediction of Mechanical Equipment. In: Wang, J., Liao, XF., Yi, Z. (eds) Advances in Neural Networks – ISNN 2005. ISNN 2005. Lecture Notes in Computer Science, vol 3498. Springer, Berlin, Heidelberg. Available from https://doi.org/10.1007/11427469_136

[4] Benkedjouh, T., Medjaher, K., Zerhouni, N., and Rechak, S., Remaining useful life estimation based on nonlinear feature reduction and support vector regression, ScienceDirect[Internet], 2013[cited 2024-10-29] Engineering Applications of Artificial Intelligence, 26(7), pp. 1751–1760. Available from https://doi.org/10.1016/j.engappai.2013.02.006

[5] Ho, Tin Kam (1995). Random Decision Forests. Proceedings of the 3rd International Conference on Document Analysis and Recognition, Montreal, QC, 14–16 August 1995. pp. 278–282. Archived from the original (PDF) on 17 April 2016 [cited 2024-10-29]. Available from https://web.archive.org/web/20160417030218/http://ect.bell-labs.com/who/tkh/publications/papers/odt.pdf

[6] Scikit-Learn. Scikit-Learn User Guide; SVM [Internet]. [cited 2024 Oct 31]. Available from: https://scikit-learn.org/stable/modules/svm.html

[7] Scikit-Learn. Scikit-Learn User Guide; StandardScaler [Internet]. [cited 2024 Oct 31]. Available from https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html

[8] Scikit-Learn. Scikit-Learn User Guide; GridSearchCV [Internet]. [cited 2024 Oct 31]. Available from https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

[9] Scikit-Learn. Scikit-Learn User Guide; GridSearchCV [Internet]. [cited 2024 Oct 31]. Available from https://scikit-learn.org/stable/modules/grid_search.html

[10] Markus Gilges. The Impact of Predictive Maintenance on the Passenger Experience[Internet]. Connected Aviation Today; October 11, 2023 [cited 2024 Oct 31]. Available from
https://connectedaviationtoday.com/the-impact-of-predictive-maintenance-on-the-passenger-experience/