DATA ANALYSIS TO REDUCE MAINTENANCE COSTS FOR LAND PLATFORMS.

Venkata Ramani Girish[1], Cheong Yek Meng[2], Eric Ang Suan Ping[2]
[1]St Joseph's Institution, 38 Malcolm Road, Singapore 308274
[2]Defence Science and Technology Agency, 1 Depot Road, Singapore 109679

ABSTRACT

In military operations, the readiness of land platforms is critical as their availability affects the operational readiness of any armed forces. Existing vehicle maintenance regime involves regular preventive maintenance to upkeep the vehicle, and corrective maintenance to replace defective components. This project employs advanced data analysis and machine learning to predict potential failures and propose just in time predictive maintenance strategies, aiming to enhance system availability through minimising time taken for regular maintenance regimes and reduce costs to replace defective components. Initial analysis indicated that variables such as Mileage, Engine Hours, and Age were key factors contributing to frequent breakdowns. Using QlikSense software for data filtering and Exploratory Data Analysis (EDA) for feature engineering, various models were trained, with the Random Forest model achieving the highest accuracy in predicting failures. Results show that it is potentially possible to use this predictive approach to pre-emptively detect failures for just in time repairs.

TABLE OF CONTENTS

INTRODUCTION

In the modern defence landscape, land platforms play a vital role in operational effectiveness, making their reliability and readiness crucial for mission success. However, the maintenance of these platforms—especially legacy systems—is challenging. Unpredictable nature of equipment failures not only disrupt operational availability but also contribute to rising maintenance costs. Current strategies of regular preventive maintenance could reduce the likelihood of breakdowns, at a cost of regular platform downtime and manpower cost. This underscores the need for innovative approaches to optimise maintenance processes.

As the saying goes, "A stitch in time saves nine". Addressing issues early prevents them from becoming larger problems. This concept is also known as 'Predictive Maintenance'. Unlike "corrective maintenance"[1] or "preventive maintenance", predictive maintenance forecasts when equipment is likely to break by using sensors, data processing, and sophisticated monitoring techniques, and these equipment could be repaired before further damages. To forecast probable future failures, this project aims to use advanced data analysis techniques on past telemetry data from ground platforms. This will potentially lower maintenance costs, increase platform availability, and improve planning of spare parts support.

This project studies the failures of the brake chamber of a selected platform that are causing operational disruptions and safety issues. These failures occur when the residual pressure of the brake is beyond an observed threshold of ≥ 0.05 bar. By analysing the platform's performance data and applying machine learning, we can identify the root causes and develop a predictive maintenance strategy to prevent future failures and enhance system reliability.

For this study, 129 records were given in total with distinct variables for each record. As there were many variables involved, this research also seeks to optimise the data analysis techniques used to better extract valuable features and conclusions from the dataset.

LITERATURE REVIEW

In a rapidly evolving world, there are many industries where machinery and equipment play a crucial role in operations. At many points in time, failures and breakdowns in the equipment occur, and substantial time costs were involved to conduct corrective maintenance. For Small and Medium Enterprises (SMEs), the cost of maintaining equipment can be the key factor that determines whether they make a profit or incur a loss. In the military context, the availability of a military platform is of utmost importance and such maintenance methods impact their operations and training effectiveness. Today, maintenance is recognised as a valuable contributor, playing a crucial role in driving performance enhancement. As such, maintenance should no longer be seen merely as a narrow operational and technical expense, but rather as a strategic long-term investment that considers the organisation's objectives and anticipated technological changes [1][2].

That is where Predictive Maintenance comes into play. Predictive maintenance leverages data analysis to detect operational issues and possible equipment faults, allowing for proactive

[1] Maintenance activities done to repair or restore equipment that has already broken down or malfunctioned.

repairs before failures happen [3]. The emergence of Industry 4.0, along with unlimited data storage, computing power, and advanced analytics, has made it possible to predict equipment failures, lower maintenance costs, and extend asset lifespan [4]. As such, predictive maintenance has really become an essential part of many companies.

A key area of research focuses on creating predictive maintenance models using advanced analytics techniques, such as machine learning, to forecast potential failures [5]. Research by Sezer et al. (2018), talks about low-cost Industry 4.0 architecture for predictive maintenance in small manufacturing enterprises, using machine learning to predict part rejection based on temperature and vibration data from a CNC turning centre. Research by Olaf Peter Schleichert et al. (2017) suggests that predictive maintenance using advanced analytics can increase equipment uptime by up to 20% in Industry 4.0. Lastly, research by Issam Mallouk et al. (2021) proposes a machine learning approach to predict the remaining useful mileage of truck tires using a supervised learning algorithm, Random Forest, which can help transportation companies reduce downtime and costs.

In a nutshell, equipment failures can be costly and disruptive. Predictive maintenance, enabled by advanced analytics, can help prevent failures, reduce costs, and improve equipment uptime.

METHODOLOGY

Main Parameters

This project focusses on five main parameters from the 129 records which affects the failure rate. They are “Variant” of the vehicle, “Phase” (of production)[2] and “Age till Failure”, “Mileage”, and “Engine hours”. In the data, “Phase” and “Variant” are ‘Discrete Dimensions’, or Qualitative, categorical discrete data. “Age till failure” is ‘Discrete Measures’, or Quantitative, numerical discrete data. “Mileage” and “Engine hours” are ‘Continuous Measures’, or Quantitative, numerical continuous data.

At first, variant was thought to be an important parameter to differentiate between breakdown rates as different variants (of the same platform) have slight modifications to cater to their functions. These modifications might lead to differences in the number of breakdowns and failures per variant. Phase was also thought to be important as different phases have different processes occurring. For example, between phase 1 and 2, production was done overseas in phase 1 and production was done locally in phase 2 even though spares were sourced overseas in both phases. This might lead to change in quality or even methods of assembly of the vehicle during production phases and cause variable failure rates.

Mileage, Engine hours and Age till Failure was then evaluated as parameters in determining breakdown as these variables indicate the usage of the Land platform vehicle, hence, the higher the usage, the higher the chance of a failure.

[2]There are three phases. In Phase 1, spares and production were done overseas. In Phase 2, spares were sourced overseas, and production was done locally. In Phase 3, both spares and production were done locally.

For the analysis in this project, preliminary data analysis will be done using QlikSense[3] to get various trends on basic variables. Then, Exploratory Data Analytics will be used to create the models.

Data Visualisation using QlikSense:

In Fig 1.1, Variant MN has the most records at 21 records, but Variants C and SV have the least records at two records each, which shows inequality and fluctuations in the count of records per variant. Therefore, using separate models to train individual variants can be ruled out.

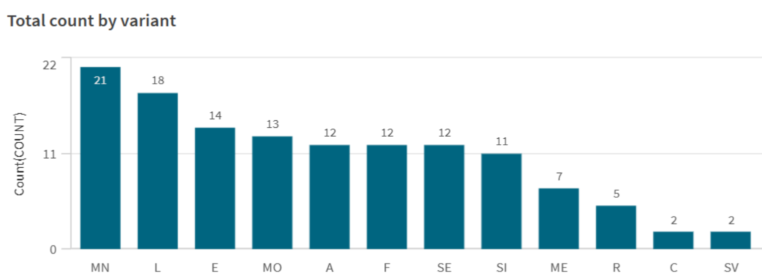

Fig 1.1: Total number of records by variant.

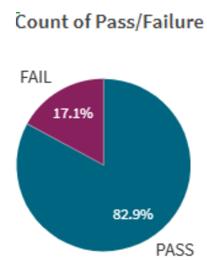

Fig 1.2: The Percentages of Passes and Failures in total

Among the records, there were a total of 107 passes and 22 failures (failure of brake chamber), translating it to 82.9% passes (see Fig 1.2). Therefore, from this, the baseline is to achieve an accuracy of 82.9% and an ideal final accuracy is an accuracy ≥ 90%.

Phase vs Percentage of Failures

Phase	Total Records	Failures	Percentage of Failures
1	36	6	**16.67%** *(to 2 d.p.)*
2	31	12	**38.71%** *(to 2 d.p.)*
3	62	4	**6.45%** *(to 2 d.p.)*
Totals	**129**	**22**	**17.05 %** *(to 2 d.p.)*

Table 1: Phase vs Percentage of Failures

[3] Qlik Sense is a user-friendly business intelligence and data analytics tool that enables real-time data exploration, visualisation, and interactive reporting.

Since each phase has a different number of records, the percentage function was used to generate insights. Based on this, there were the most percentage of failures at phase 2 (spares were sourced overseas & production was done locally) at 38.71%. Fig 1.5 *(in the Appendix)* has also shown this, where phase 2 had the most count of failures at 12 failures. From the table, Phase has enough records for every individual Phase and thus, training individual models per phase is a potential approach to consider.

Looking at the number of failures and failed variants for each variable (mileage, engine hours, phase and age), Figures 1.3 to 1.9 *(in the Appendix)*were generated. From all the figures, Phase (Fig 1.5) and Age till Failure (Fig 1.6 and 1.7) had the most correlation. In Fig 1.5, as stated above, it appears that individual phases have enough data and Phase 2 shows a significantly higher failure rate than Phases 1 and 3. In Fig 1.7, it was found that Age matters as well as 4 years till failure had the highest failure rate at 13 failures compared to other ages. Variant also affects age till failure, but this trend cannot be fully concluded as there were different numbers of records for each variant due to the divide. However, mean age till failure remains at 3.68 years *(to 2 d.p.)*.

As such, the data visualisation section can be concluded.

Main Insights generated:

(1) Variant somewhat impacts failure, although not yet clearly possible to tell as each variant has a different number of datapoints. Hence, although it is important, a separate model for every variant cannot really be used.

(2) There is a correlation between age and failure. 4 years after manufacturing, there were the greatest number of failures at 13 failures. Thus, using separate models for each Age till failure group is somewhat a possible method for consideration.

(3) Phase 2 is strongly correlated with failures as it had the most percentage of failures, 38.71%. Therefore, training separate models for each Phase is a potential approach that could be taken.

(4) Mileage and Engine hours, contrary to intuition, show no significant correspondence between both each other and failure rates.

MODEL SELECTION AND TRAINING

<u>Four models were chosen based on their suitability for the data set given:</u>

Decision tree model

Decision tree was chosen as it is a type of supervised learning algorithm in machine learning, applicable to both classification and regression tasks. This model works by recursively splitting the data into subsets based on input feature values, creating a tree structure [8].

Gradient Boosting

Gradient Boosting is a widely used machine learning boosting algorithm designed for both classification and regression problems. It is also commonly employed for structured data and predictive tasks. As a type of ensemble learning method, boosting builds models sequentially, with each subsequent model aiming to address the errors made by its predecessor. It integrates multiple weak learners to create a single, strong predictive model [9].

k-Nearest Neighbours

The *k*-nearest neighbours (KNN) algorithm is a supervised, non-parametric classifier that makes predictions or classifications based on the proximity of data points. It is widely used in machine learning for classification and regression tasks and anomaly detection due to its simplicity and effectiveness [10].

Random Forest

Random forest is a widely used machine learning algorithm that aggregates the results of several decision trees to produce a final outcome. Its popularity has grown due to its simplicity and versatility, as it can effectively solve both classification and regression tasks, as well as feature selection tasks (for feature engineering) [11].

PERFORMANCE OF MODELS

Feature Engineering

Fig 2.1: Feature Engineering [12]

Feature engineering, in an essence, is the process of taking raw data and turning it into something that machine learning models can understand. The quality of the features used to train machine learning models is critical to their success. Feature engineering is a set of techniques that allows us to create new features by combining or transforming existing ones. These techniques help to highlight the most important patterns and relationships in the data, which in turn helps the machine learning model to learn from the data more effectively.

Before doing feature engineering, due to lesser domain knowledge, data was mostly classified normally like using sum, mean and count. Examples during the thought process included adding Age, Mileage and Engine hours as part of the main features as they suggest wear-and-tear, which leads to higher probability of brake chamber failure. After gaining adequate domain knowledge, more complex features were included, which will be elaborated on in the following paragraphs.

In this case, the feature engineering done in this paper included Mileage per Engine Hour, total residual pressure, Pressure and age, Pressure Age Mileage, Pressure Age Mileage Engine and Pressure age Mileage hours *(Their formulas could be found in the Appendix)*. **Mileage per Engine Hour** feature measures the system's efficiency by calculating the distance covered per unit of engine runtime, emphasising its operational performance. By normalising mileage using engine hours, it ensures consistency across vehicles, allowing for more accurate comparisons regardless of differences in scale. However, this might be a little inaccurate because Engine Hours also counts idling time[4], which does not add to any distance in mileage. When the vehicle is not moving, there is nothing much applied on the wheels itself, so no wear and tear is expected. **Pressure Age Mileage** on the other hand, combines the Total Residual Pressure, Age till Date, and Mileage to represent multiple stress factors affecting system performance. It captures the interaction between age, usage (Mileage), and pressure, highlighting how these elements collectively impact the likelihood of failure. The feature **Pressure Age Mileage Hour** combines Total Residual Pressure, Age till Date, and Mileage per Engine Hour to highlight the interplay of system stress, aging, and operational efficiency. By focusing on efficiency alongside cumulative wear factors, it effectively identifies patterns linked to potential failures, aiding in more accurate predictions.

After choosing these features, the data was loaded for all four chosen models to see which features the models think are important, generating. Figures 2.2 to 2.5 *(in Appendix).* Based on the graphs, the following results were observed: In the **Decision Tree** Model, Total Residual Pressure, Pressure Age Mileage and Age till Date were rated the top three most important features. In the **Gradient Boosting** Model, Total Residual Pressure, Engine Hours and Age till Date were rated the top three most important features. In the ***k*-nearest neighbours** model, Total Residual Pressure, Pressure and Age, and Pressure Age Mileage Hour were rated the top three most important features. In the **Random Forest** Model, Total Residual Pressure, Pressure and Age, and Pressure Age Mileage Engine were rated the top three features respectively. From this data, the following can be implied:

(1) All models found Total Residual Pressure as their most important factor.
(2) In Decision Tree, Gradient Boosting and *k*-nearest neighbours, Age till Date was within top 3 of importance.
(3) In all Models, Pressure and Age, as well as other variations with other variables (including Pressure Age Mileage, Pressure Age Mileage Hour, et cetera) have been within top 3 of importance.

Prediction Models:

Based on the results of the feature analysis, some code was generated for the prediction model. Through that, the accuracy of the models are as shown in Table 2 and the confusion matrix in 3.

Model	**Accuracy / %** *(to 5 s.f.)*	**Reproducibility / %**
Decision Tree	97.436	100
Gradient Boosting	92.308	100
k-nearest neighbours	84.615	100
Random Forest	100.00	100

Table 2: Model vs Accuracy

[4] Idling occurs when the Engine is turned on but the vehicle is not moving.

Decision Tree Model: Firstly, the Decision Tree model had an accuracy of 97.436%. The Decision Tree performed very well, achieving near-perfect accuracy. Its simple, interpretable structure likely captured important patterns in the data effectively. However, close to perfect accuracy could lead to high risks of overfitting in the decision tree model compared to ensemble methods. A model is said to be overfit when it matches the training data too closely and is unable to generalise to new, unknown data [13].

Gradient Boosting Model: The Gradient Boosting model had an accuracy of 92.308%. By integrating weak learners (shallow decision trees), gradient boosting demonstrated its capacity to iteratively improve predictions with a high accuracy. Although its accuracy is lower than Random Forest and Decision Tree, Gradient Boosting often generalises well, especially on more complex datasets. It may require further tuning of hyperparameters to improve performance [14].

k*-Nearest Neighbours Model*: With an accuracy of just 84.615%, the k-Nearest Neighbours (k-NN) algorithm performed the worst among all the models. This is mainly because k-NN relies on distance-based calculations to identify the closest neighbours, and features with larger numerical ranges can dominate the distance measurement. As a result, predictions become inaccurate since some features contribute more than others, even if they aren't more important. This issue becomes more prominent when working with many features, a problem known as the "curse of dimensionality" [15]. As stated in the previous parts, an accuracy of $\geq 90\%$ was ideal. Nevertheless, to fix this, scaling the features using techniques like normalisation or standardisation ensures that each feature contributes fairly to the distance calculations [16].

Random Forest Model: The Random Forest achieved a perfect accuracy of 100.00%, showcasing its ability to handle the given dataset exceptionally well. This high performance can be attributed to its ensemble nature, where multiple decision trees are combined to reduce variance and overfitting, thereby improving generalisation. However, a perfect accuracy score raises concerns about potential overfitting, where the model memorises the training data instead of learning general patterns. Overfitting limits the model's ability to perform accurately on new, unseen data. To verify its robustness, additional techniques like cross-validation or testing on an independent dataset are essential to ensure that the Random Forest's performance is not just a result of overfitting but reflects true predictive power [17].

The reproducibility of the models also matters as it ensures that the results of a model can be consistently replicated under the same conditions. For this model to have a high reproducibility, a random seed of 42 was set. This reproduces training results when splitting datasets equally in each time, causing the same event to be repeated exactly [18]. Reproducibility builds trust in the model's results, as it shows that the findings are not due to random chance or specific conditions and allows data scientists to debug and improve models by understanding how changes in data, parameters, or algorithms affect outcomes [19]. The results show that all models have 100% reproducibility, which might suggest that the model might be overfitted to the training data, capturing noise and specific patterns that do not generalise well to new, unseen data. Moreover, since the number of records (or sample size) is smaller in this dataset, the model might perform better on it, and its performance might not be ideal on larger, more diverse dataset, impacting its effectiveness in real-world applications. For future experiments, training data can be increased so that there is a more generalised model [20].

Model	True Positives	True Negatives	False Positives	False Negatives
Decision Tree	33	5	0	1
Gradient Boosting	31	5	0	3
k-nearest neighbours	32	1	4	2
Random Forest	34	5	0	0
Mean	32.5	4	1	1.5

Table 3: Confusion Matrix[5]

Based on the data models' performance in Table 2, a confusion matrix was produced as seen in Table 3. From the table, it is seen that there are no False positives (the model predicted it as a pass but actually it is a failure) for all models except for *k*-Nearest Neighbours. This could have been due to the variance in magnitude of the data, which leads to inaccurate measure of distance and potentially false predictions. Nonetheless, since a large majority of the classifications are true positives and true negatives, it suggested that there is high accuracy in determining when repairs are necessary, so that platforms are only sent for repair when required and are not unnecessarily dispatched when repairs are not needed. However, there are a few false negatives for each model, which could degrade predictive model performance, leading to incorrect conclusions and flawed decisions [22]. In this context, false negatives may lead to maintenance being conducted as it is predicted to be a failure, but it should have passed, leading to unnecessary waste of time and money. Such false alarms would impact the availability of the land platform.

CONCLUSION AND EVALUATION

Machine learning models used in this paper were able to give a high accuracy and potentially raise the availability of the land platform. Random Forest had perfect accuracy, followed by Decision Tree, Gradient Boosting and *k*-Nearest Neighbours. These models had a relatively ideal accuracy as expected in the beginning and can benefit the military operations by optimising the resources.

Applications and Future Research

These models can be potentially used to predict failures for the platform and send them for repairs just in time, hence reducing costs and optimising resources. For future research, more data could have been used in the analysis phase of this research so that there could have been more generalisations and potentially, more trends can be observed. In addition, taking regular pressure readings in the platform would help on time classification of failures for the platform. Lastly, there

[5] A true positive and true negative suggests that the model correctly predicted that it is a pass and a failure respectively while a false positive and false negative suggests that the model wrongly predicted that it is a pass and a failure respectively. [21]

could be more sensors in the vehicle to get more parameters which were not explored in this paper to be checked so that insights on other parameters can lead to more insights on other factors contributing to brake failure.

ACKNOWLEDGEMENTS

I am deeply grateful to my mentors, Mr. Cheong Yek Meng and Mr. Eric Ang Suan Ping, for their invaluable guidance and support throughout this project. Without their expertise, this project would not have been feasible. I would also like to thank Ms. Esther Chew for her assistance with administrative and HR tasks, and the Defence Science and Technology Agency (DSTA) for the opportunity to participate in the 2024 Research@YDSP program.

REFERENCES

[1] Al-Turki, U. (2011). A framework for strategic planning in maintenance. *Journal of Quality in Maintenance Engineering*, *17*(2), 150–162. https://doi.org/10.1108/13552511111134583

[2] Sezer, E., Romero, D., Guedea, F., Macchi, M., & Emmanouilidis, C. (2018, June 1). *An Industry 4.0-Enabled Low Cost Predictive Maintenance Approach for SMEs*. IEEE Xplore. https://doi.org/10.1109/ICE.2018.8436307

[3] *What is Predictive Maintenance? [Benefits & Examples]*. (n.d.). Fiix. https://fiixsoftware.com/maintenance-strategies/predictive-maintenance/

[4] Siddharth Patil , Dhaval Thakkar, Suhas S, Nicki Yochim, Nikhila Gandikota, & Smrithi V. (2022). *Predictive maintenance Deloitte's approach*. Deloitte. https://www2.deloitte.com/content/dam/Deloitte/us/Documents/process-and-operations/us-predictive-maintenance.pdf

[5] Lee, C. K. M., Cao, Y., & Ng, K. H. (2017). Big Data Analytics for Predictive Maintenance Strategies. *Supply Chain Management in the Big Data Era*, 50–74. https://doi.org/10.4018/978-1-5225-0956-1.ch004

[6] Olaf Peter Schleichert, Björn Bringmann, Hardy Kremer, Sergey Zablotskiy, & David Köpfer. (2017). *Predictive maintenance: Taking pro-active measures based on advanced data analytics to predict and avoid machine failure*. Deloitte. https://www.beekeeper.io/wp-content/uploads/2024/10/Deloitte_Predictive-Maintenance_PositionPaper.pdf

[7] Issam Mallouk, Yves Sallez, & Badr. (2021). Machine learning approach for predictive maintenance of transport systems. *HAL (Le Centre Pour La Communication Scientifique Directe)*. https://doi.org/10.1109/tst52996.2021.00023

[8] Bagus Murdyantoro. (2023, November 19). *Building Decision Tree Algorithm from Scratch in Python*. Medium. https://medium.com/@bagusmurdyantoro1997/building-decision-tree-algorithm-from-scratch-in-python-4adc26ba1b57

[9] GeeksForGeeks. (2020, August 25). *ML - Gradient Boosting*. GeeksforGeeks. https://www.geeksforgeeks.org/ml-gradient-boosting/

[10] IBM. (2023a). *What Is the k-nearest neighbours algorithm? | IBM*. Www.ibm.com; IBM. https://www.ibm.com/topics/knn

[11] IBM. (2023b). *What Is Random Forest? | IBM*. Www.ibm.com; IBM. https://www.ibm.com/topics/random-forest

[12] GeeksforGeeks. (2023, March 20). *What is Feature Engineering?* GeeksforGeeks. https://www.geeksforgeeks.org/what-is-feature-engineering/
[13] DataHeadhunters. (2024, February 3). *Exploring the Limits of Decision Trees: Depth, Bias, and Variance*. Dataheadhunters.com. https://dataheadhunters.com/academy/exploring-the-limits-of-decision-trees-depth-bias-and-variance/
[14] GeeksforGeeks. (2024, March 6). *Gradient Boosting vs Random Forest*. GeeksforGeeks. https://www.geeksforgeeks.org/gradient-boosting-vs-random-forest/
[15] Shetty, B. (2022, August 19). *Curse of Dimensionality*. Built In. https://builtin.com/data-science/curse-dimensionality
[16] Miesle, P. (2024, September 6). *What is the K-Nearest neighbours (KNN) Algorithm?* DataStax; DataStax. https://www.datastax.com/guides/what-is-k-nearest-neighbours-knn-algorithm
[17] Luan, J., Zhang, C., Xu, B., Xue, Y., & Ren, Y. (2020). The predictive performances of random forest models with limited sample size and different species traits. *Fisheries Research*, *227*, 105534. https://doi.org/10.1016/j.fishres.2020.105534
[18] GeeksforGeeks. (2019, May 2). *random.seed( ) in Python*. GeeksforGeeks. https://www.geeksforgeeks.org/random-seed-in-python/
[19] Sundeep Teki. (2023, May 4). *The case for reproducible data science*. Domino.ai. https://domino.ai/blog/reproducible-data-science
[20] Nautiyal, D. (2017, November 23). *Underfitting and Overfitting in Machine Learning*. GeeksforGeeks. https://www.geeksforgeeks.org/underfitting-and-overfitting-in-machine-learning/
[21] Nisha Arya Ahmed. (2023, November 30). *What is A Confusion Matrix in Machine Learning? The Model Evaluation Tool Explained*. Datacamp.com; DataCamp. https://www.datacamp.com/tutorial/what-is-a-confusion-matrix-in-machine-learning
[22] Learn Statistics Easily. (2024). *What is: False Negative*. LEARN STATISTICS EASILY. https://statisticseasily.com/glossario/what-is-false-negative/

APPENDIX

Formulae:

1. Mileage Per Engine Hour = Mileage ÷ Engine Hours
2. Total Residual Pressure = RHS Residual Pressure + LHS Residual Pressure
3. Pressure and Age = Total Residual Pressure + Age Till Date
4. Pressure Age Mileage = Total Residual Pressure + Age Till Date + Mileage
5. Pressure Age Mileage Engine = Total Residual Pressure + Age Till Date + Mileage + Engine Hours
6. Pressure Age Mileage Hour = Total Residual Pressure + Age Till Date + Mileage Per Engine Hour.

Figures:

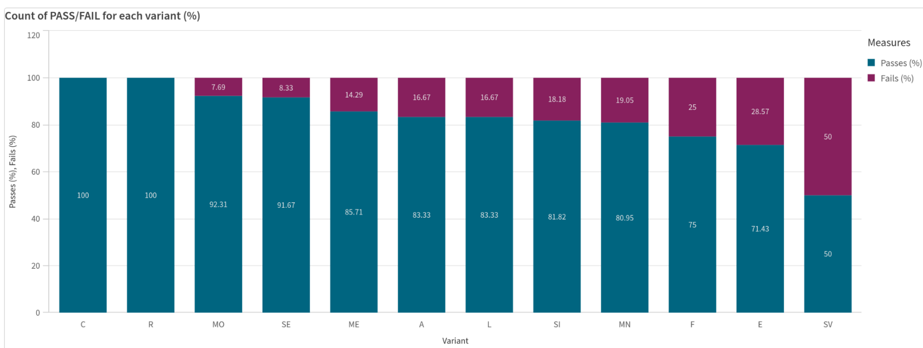

Fig A1.3: The Percentages of Passes and Failures by variant

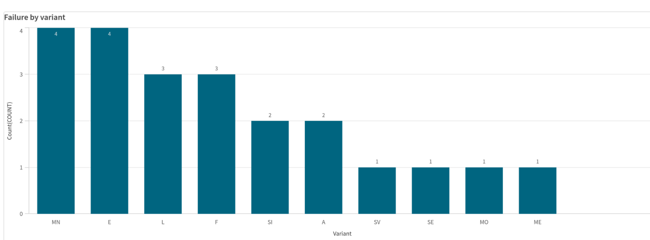

Fig A1.4: The Total Count of Failure by Variant

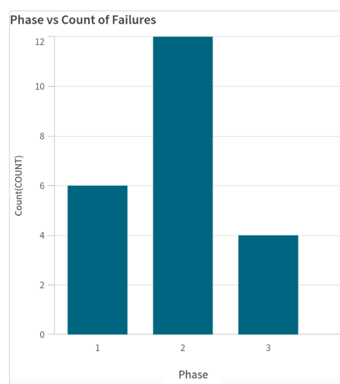

Fig A1.5: Count of Failures vs Phase

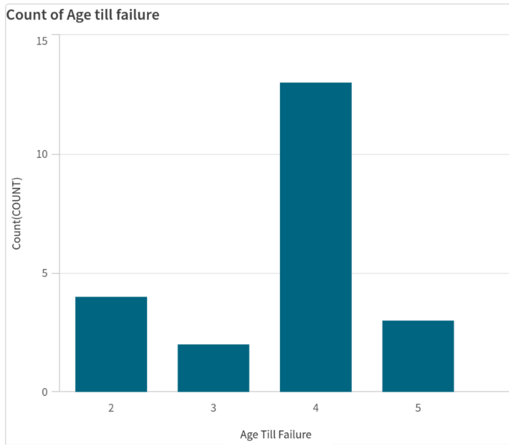

Fig A1.6: Count of failure vs Age till Failure

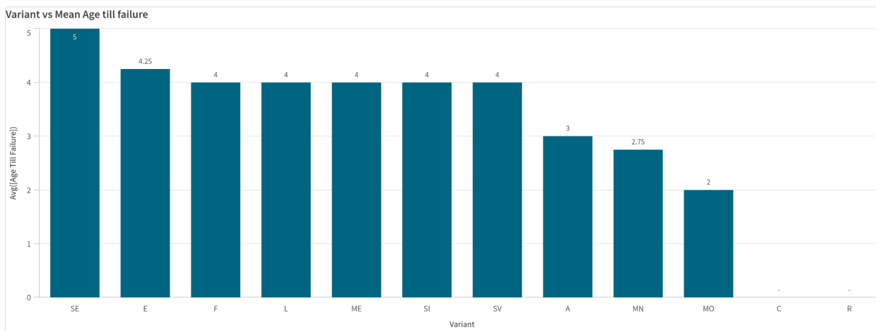

Fig A1.7: Variant vs Mean Age till failure

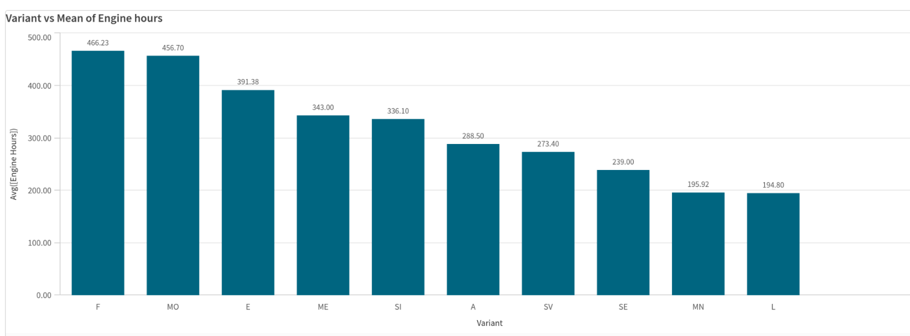

Fig A1.8: Variant vs Mean of Engine hours

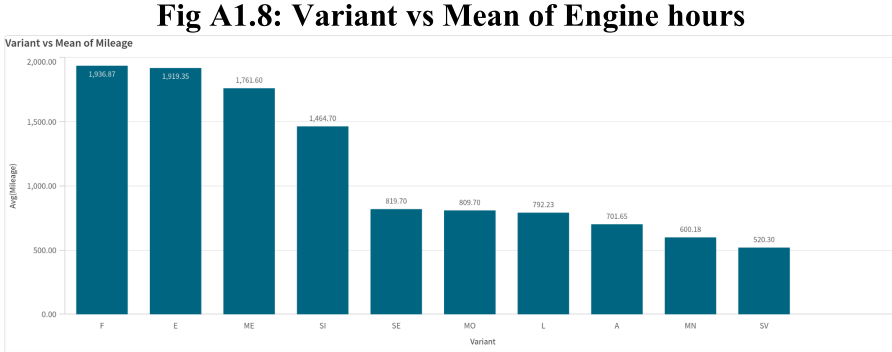

Fig A1.9: Variant vs Mean of Mileage

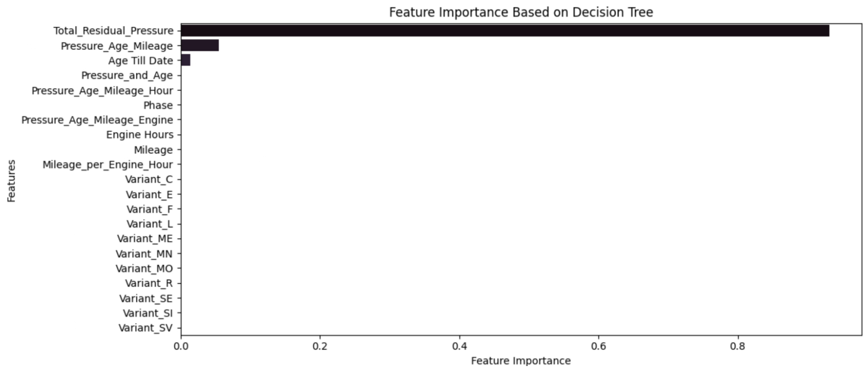

Fig A2.2: Decision trees feature engineering

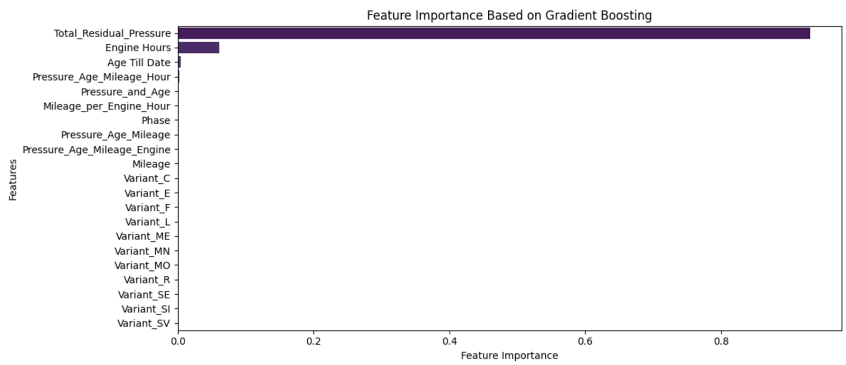

Fig A2.3: Gradient Boosting feature engineering

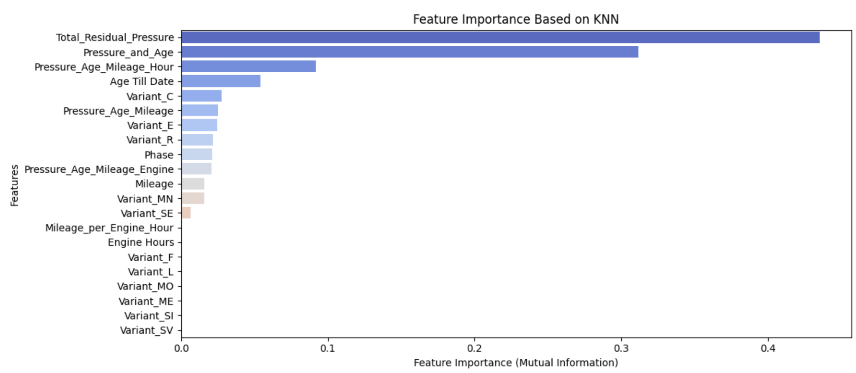

Fig A2.4: *k*-nearest neighbours feature engineering

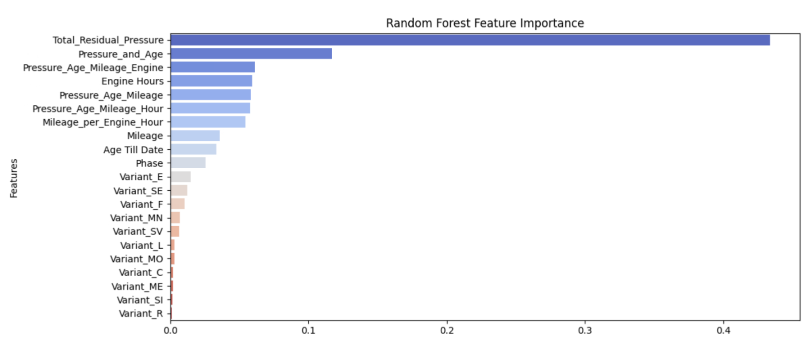

Fig A2.5: Random Forest feature engineering