Predictive Maintenance of Construction Equipment using Log Data

A Data-centric Approach

Kotriwala, Bazil Muzaffar

Award date:
2021

Awarding institution:
Royal Institute of Technology

DEGREE PROJECT IN TECHNOLOGY,
SECOND CYCLE, 30 CREDITS
STOCKHOLM, SWEDEN 2021

Predictive Maintenance of Construction Equipment using Log Data

A Data-centric Approach

Bazil Muzaffar Kotriwala

KTH ROYAL INSTITUTE OF TECHNOLOGY
ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

Authors

Bazil Muzaffar Kotriwala <bmko@kth.se>
Information and Communication Technology (ICT) Innovation
KTH Royal Institute of Technology

Place for Project

Eskilstuna, Sweden
Volvo Construction Equipment

Examiner

Magnus Boman <mab@kth.se>
School of Electrical Engineering and Computer Science
KTH Royal Institute of Technology

Supervisor

Saikat Chatterjee <sach@kth.se>
School of Electrical Engineering and Computer Science
KTH Royal Institute of Technology

Sharif Khan Pattan <sharif.pattan@volvo.com>
Department of Electronics and Embedded Systems
Volvo Construction Equipment

Abstract

Construction equipment manufacturers want to reduce the downtime of their equipment by moving from the typical reactive maintenance to a predictive maintenance approach. They would like to define a method to predict the failure of the construction equipment ahead of time by leveraging the real-world data that is being logged by their vehicles. This data is logged as general event data and specific sensor data belonging to different components of the vehicle. For the scope of this study, the focus is on articulated hauler vehicles with engine as the specific component under observation. In the study, extensive time and resources are spent on preparing both the real-world data sources and coming up with methods such that both data sources are ready for predictive maintenance and can also be merged together. The prepared data is used to build respective remaining useful life machine learning models which classify whether there will be a failure in the next x days. These models are built using data from two different approaches namely, lead data shift and resampling approach respectively. Three different experiments are carried out for both of these approaches using three different combinations of data namely event log only, engine sensor log only, event and sensor log combined. All these experiments have an increasing look ahead window size of how far into the future we would like to predict the failure. The results of these experiments are evaluated in relation to which is the best approach, data combination, and window size to foresee engine failures. The model performance is primarily distinguished by the F-Score and Area under Precision-Recall Curve.

Keywords

Predictive Maintenance, Construction Equipment, Event Log, Sensor Log, Machine Learning, Imbalanced Data

Abstrakt

Tillverkare av anläggningsutrustning vill minska stilleståndstiden för sin utrustning genom att övergå från det typiska reaktiva underhållet till ett förebyggande underhåll. De vill definiera en metod för att förutse fel på byggutrustningen i förväg genom att utnyttja de verkliga data som loggas av fordonen. Dessa data loggas som allmänna händelsedata och specifika sensordata som tillhör olika komponenter i fordonet. I den här studien ligger fokus på ledade dragfordon med motorn som den specifika komponent som observeras. I studien läggs mycket tid och resurser på att förbereda båda datakällorna i den verkliga världen och att ta fram metoder så att båda datakällorna är redo för förebyggande underhåll och kan slås samman. De förberedda uppgifterna används för att bygga maskininlärnings modeller för återstående livslängd som klassificerar om det kommer att ske ett fel inom de närmaste x dagarna. Modellerna byggs upp med hjälp av data från två olika metoder, nämligen lead data shift och resampling approach. Tre olika experiment utförs för båda dessa metoder med tre olika kombinationer av data, nämligen endast händelselogg, endast motorsensorlogg och kombinerad händelselogg och sensorlogg. Alla dessa experiment har en ökande fönsterstorlek för hur långt in i framtiden vi vill förutsäga felet. Resultaten av dessa experiment utvärderas med avseende på vilket tillvägagångssätt, vilken datakombination och vilken fönsterstorlek som är bäst för att förutse motorhaverier. Modellens prestanda bedöms i första hand med hjälp av F-poäng och arean under Precision-Recall-kurvan.

Nyckelord

Prediktivt underhåll, byggutrustning, händelselogg, sensorlogg, maskininlärning, obalanserade data

Acknowledgements

I would like to thank KTH Royal Institute of Technology and Volvo Construction Equipment for giving me the opportunity to carry out this research work. It has been a great experience and I hope the findings of the work are of benefit to both organizations.

I would like to extend my gratitude to both my supervisors and the examiner at Volvo Construction Equipment and KTH Royal Institute of Technology respectively. All of you have been of great support throughout the project and have offered great feedback which has helped to shape this project in the right way. I am grateful for your continued assistance throughout this journey and wish you all the best with your future endeavors.

Acronyms

ARIMA	Autoregressive Integrated Moving Average
AU-PRC	Area under Precision-Recall curve
BCC	Binary-class Classification
CM	Condition Monitoring
CRISP-DM	CRoss Industry Standard Process for Data Mining
DL	Deep Learning
DT	Decision Trees
ECU	Electronic Control Unit
KTH	KTH Royal Institute of Technology
LDA	Logged Data Analysis
LSTM	Long Short Term Memory
LR	Logistic Regression
MCC	Multi-class Classification
ML	Machine Learning
PdM	Predictive Maintenance
RF	Random Forest
RSF	Random-Survival Forest
RNN	Recurrent Neural Networks
RUL	Remaining Useful Life
SVM	Support Vector Machines
VCE	Volvo Construction Equipment

Contents

Chapter 1

Introduction

The construction industry is central to the development of infrastructure of any country in the world [26]. Construction equipment manufacturers are a crucial component of this industry and the equipment they produce is essential to the success of construction projects. The key factors which lead construction equipment manufacturers to become market leaders is the quality and reliability of their equipment [44]. Reliable equipment has the least downtime, breakdowns, and is always readily available. Predictive Maintenance (PdM) is a technique that foresees the breakdown and sudden failure of equipment so that necessary preventable measures can be taken. This technique has shown to reduce the unexpected downtime of equipment by 70%, maintenance costs by 5% to 10%, and increased equipment uptime by 10% to 20% [47]. Log data is one of the main sources that records the functioning of such equipment via various software applications running on the equipment [44].

1.1 Background

Construction companies and contractors are the primary customers for construction equipment manufacturers such as Volvo Construction Equipment (VCE). The fleet of heavy earthmoving machinery is the main resource for these customers and they rely on them for the success of the projects they undertake [14, 26]. If these machines are not maintained well, they break down which causes delays in project completion times and effectively causes economic loss. Therefore, it is imperative to fix any imminent problems with the equipment ahead of time to avoid delays and in some cases even accelerate the project timeline with higher productivity [26]. Thus, the customers take

precautionary measures by identifying equipment that needs replacing ahead of time and similarly rely on the construction equipment manufacturers to notify them of any impending failures of their equipment which can cause a delay in their projects [14]. In the US, nearly 10% of equipment from the fleet needs to be replaced annually by these customers [24].

Over the years, PdM based on data-driven methods has become the most powerful and useful solution to tackle the issue of maintenance of industrial equipment. It makes use of the different types of big data generated by the equipment such as event-log data, component-based sensor data, duration logs, etc., to track its health and predict its next failure to conduct corrective maintenance before that happens [63]. Therefore, high-quality data is a key component for the success of any PdM system. Companies have been investing in infrastructure such as software, sensors, servers, cameras, etc., to collect such data so that they may equip themselves to perform PdM [47].

The log data being collected can be categorized into two different types of logs for construction equipment; event logs, sensor data logs (duration and distribution logs for specific components). These logs are used to retrieve runtime information of the equipment and find out patterns that indicate as to why a specific fault occurred [4, 22]. With the advancement in computational power and data mining techniques, analyzing this complex log data has shifted from a manual to an automated approach [22]. The software and sensors can be installed on different components of the equipment such as distance travelled, operating hours, temperature, fuel level, etc., to record this data [44]. However, installing such software or adding these sensors can be costly if you wish to monitor all the components of the equipment and typically due to a large fleet of equipment [44].

1.2 Problem

The problem can be divided in two parts, an overall general problem and a specific problem. Solving the specific problem inherently will contribute towards solving the overall problem. Therefore, in this study we aim to solve the specific problem.

General Problem

Downtime of equipment is primarily one of the most critical problems faced by the construction industry [26]. This problem has an impact on all parties involved in the industry; manufacturers, contractors, automotive engineers, customers, etc. With the rise of Industry 4.0 [47], the production processes have become completely interconnected and streamlined. The production processes and construction projects heavily rely on the equipment to be readily available and work correctly [2]. Therefore, the downtime of equipment has a ripple effect on interrupting and/or stopping production processes or construction projects prematurely. This has a large negative impact on the industry in terms of delays in production, high repair cost, depreciation of equipment, waste of economic and material resources [18]. However, if we can foresee the failures of the equipment, we can reduce the aforementioned problems by taking action ahead of time, thereby saving economic, time, and material resources [59].

Specific Problem

To predict the failure of the equipment ahead of time and reduce the downtime of equipment, we need to make use of equipment data that is being collected by equipment manufacturers on a regular basis for the numerous amount of different equipment they build. One such manufacturer is VCE who has a vast amount of vehicles in their fleet sold to customers around the world for construction projects. These vehicles are are collecting and logging the vehicle data in the VCE system. This gives rise to the problem of how can we actually leverage this data sitting idle in the databases and/or currently being used for reporting purposes to foresee vehicle failure. To solve this problem, first we define a precise research question.

1.3 Research Question

In this thesis, we aim to answer the following research question:

How can real-world event log and component based sensor log data be used to predict failure of construction equipment reliably ahead of time using a machine learning approach to enable appropriate prescriptive actions?

1.4 Purpose

In order to further increase the availability of customers' fleet of construction equipment whenever they require it, VCE [49] needs to develop smart solutions allowing anticipation of maintenance and failure of their equipment. In this way, the aftermarket chain can be ready and anticipate part availability while the customer can plan for equipment service at the right time. This results in a very low disturbance at the worksite, avoids project delays and ensures maximum productivity of each machine. Overall, there is higher availability of equipment for the customer to use and higher productivity which has a positive economic impact [44].

1.5 Goal

The goal of the thesis is to define a method to predict the failure of construction equipment (e.g. haulers or wheel-loaders) ahead of time by analyzing the specific machines' logs (generated by each Electronic Control Unit (ECU)) in order to identify patterns that can cause the machine to malfunction. The desired outcome for VCE is to have a model/solution which can do this effectively ahead of time for any specific component (e.g. engine, transmission, brakes, etc.) of any machine. The model will serve as proof of concept for the company that they can leverage the data they collect from their construction equipment to improve the uptime of their machines and reduce the number of failures. The model will act as a foundation that they can build upon and scale for other types of components and machines. To develop this model, we will use Machine Learning (ML) methods along with real-world log data to perform PdM.

1.6 Ethics and Sustainability

PdM is a significant contributor towards the sustainable future of the world. It plays a key role to further assist the manufacturing industry to reach the sustainability targets outlined by the United Nations [37]. It falls under the area of Sustainable Maintenance [1] which aims to reduce the environmental impact of the breakdown and depreciation of industrial equipment. It does this by maintaining the health of the equipment by replacing parts and reducing breakdowns ahead of time. This improves the durability of the equipment thereby, reducing the overall cost of maintaining the equipment. This

contributes positively to achieving sustainability targets such as reducing the energy usage (e.g. fixing electrical appliances prior to breakdown), increasing sustainable energy generations (e.g. lesser downtime for wind turbines), reducing maintenance waste, improving the safety of workers, etc [15, 42].

Nonetheless, its important to note that this PdM approach moves the industry towards more automation and reliable equipment manufacturing with an increase in optimal use of resources. However, this may result in fewer human resources being needed to maintain the equipment and be replaced by analysts who can apply these advanced predictive techniques. Even with these jobs being replaced, overall more jobs at the ground level (e.g. technicians) will be under threat as opposed to those created at an analytics level which may be a downside. Lastly, the project does not address any ethical questions or has any ethical implications.

1.7 Methodology

For this thesis, we will use the quantitative research method to answer the research question. This method is optimal to carry out experiments and test hypotheses using measurable tools such as big datasets, statistics, computations, mathematics, etc.

More specifically, we will make use of the applied research method which is suitable for our research since we are solving a practical problem whilst building on existing work and research. Our research will involve building a ML model using real-world log data directly retrieved from VCE's articulated haulers which will enable us to predict failures in advance.

There will not be a need for any data collection method as the focus of the thesis is to work with the existing historical data available at VCE. However, we will make use of data analysis methods mainly computational mathematics, coding, and statistics.

1.8 Delimitations

The log data provided by VCE for their construction equipment is not logged in real-time. The readings are recorded when a technician connects to the machine or when a failure occurs. Therefore, the data may not be at regular time intervals, so real-time scaling/forecasting is not considered in this project.

There is a wide range of construction equipment which VCE has in their product range, so we do not consider all machines for the purpose of our research. We only focus on one type of machine i.e. articulated haulers and build a model for that, which acts as a proof of concept for VCE to build upon and scale to the rest of their products. Similarly, each machine has different components e.g. engine, transmission, brakes, etc., with each of them having many different types of failures. In our research, we only consider one specific component i.e. engine and and focus on specific engine related failures.

Therefore, it is important to note that our solution is focused on a specific set of machines, components, and failures respectively. Moreover, we only consider machines that have a significant amount of historical data and not consider those which have been recently sold to customers thereby to avoid the cold-start problem for our model.

1.9 Outline

The remainder of the thesis is structured as follows:

- Chapter 2: Theoretical Background
- Chapter 3: Method
- Chapter 4: Data Preprocessing and Exploration
- Chapter 5: Applied Predictive Maintenance
- Chapter 6: Model, Results and Analysis
- Chapter 7: Discussion and Conclusion

Chapter 2

Theoretical Background

In this chapter, we look at the overall area of maintenance, its application and what role PdM has in this space. Moreover, we highlight the related work that has used PdM for different types of equipment using ML and non ML methods.

2.1 Maintenance

Maintenance plays a key role in nearly every industry since the industrial revolution. The shift to mechanical machinery, electrical equipment, hardware, software, automation, and in recent times Industry 4.0 [47], has enhanced the importance and dependence on maintenance more than ever. Simultaneously, modern industrial equipment has become highly complex; made up of many different mechanical and electrical components which are working together in harmony, and therefore, there is a higher chance of any one component failing. This by no means indicates that the failure is critical but it possibly can be. For instance, construction equipment has an engine, brakes, axle, electric components within which faults can occur [52]. Thus, customers purchasing such products require it to be maintained and similarly, manufacturers producing these products offer maintenance services and guidelines for their products. This has given rise to third party maintenance service providers as well.

On an industrial level, with the rise of automation, maintenance plays an increasingly important role in the smooth functioning of the modern production processes and ensures the uptime of equipment [47]. Therefore, there is a dependence on improving the maintenance techniques to guarantee higher productivity of modern machines and

processes.

2.1.1 Types of Maintenance

Maintenance can be carried out in different ways. Primarily, they are categorized as follows:

- **Reactive Maintenance**

  It is the most typical form of maintenance across all industries. You only perform maintenance on the equipment once it has already broken down. Mostly, it is applied to low-value products for instance - changing a light bulb once it has fused [19]. However, in some cases, high-value products such as cars owned by individuals are only taken to the mechanic to fix once they have broken down.

- **Preventive Maintenance**

  This form of maintenance is performed at scheduled regular intervals. It is often performed outside working hours and the cost of maintenance is typically not high. Equipment owners who perform this type of maintenance have internal maintenance plans scheduled at regular time intervals for e.g. bi-annual inspection of wind turbines [19, 20].

- **Condition-Based Maintenance**

  This form of maintenance is similar to preventive maintenance, however, instead of regular time intervals, it takes into account thresholds which the equipment reaches and then is inspected. For instance, automotive vehicle manufacturers would recommend that after driving every 10,000 kilometers, you should get your vehicle checked in for service [9]. Similarly, airplanes are checked after traveling a certain amount of distance as well [47]. It is often a form of recommended maintenance by equipment manufacturers.

- **Predictive Maintenance** (PdM)

  This is a modern type of maintenance that makes use of equipment data and advanced analytical techniques to predict the failure of the equipment ahead of time. It can predict the Remaining Useful Life (RUL) of the equipment and allows maintenance to be scheduled in advance [20]. Typically, this method is used in industries where the downtime of equipment has a high impact financially [47].

For instance, construction equipment manufacturers would like to know ahead of time whether their machine owned by a customer/contractor will break down anytime soon so that they can fix it before the customers' project is delayed due to a breakdown.

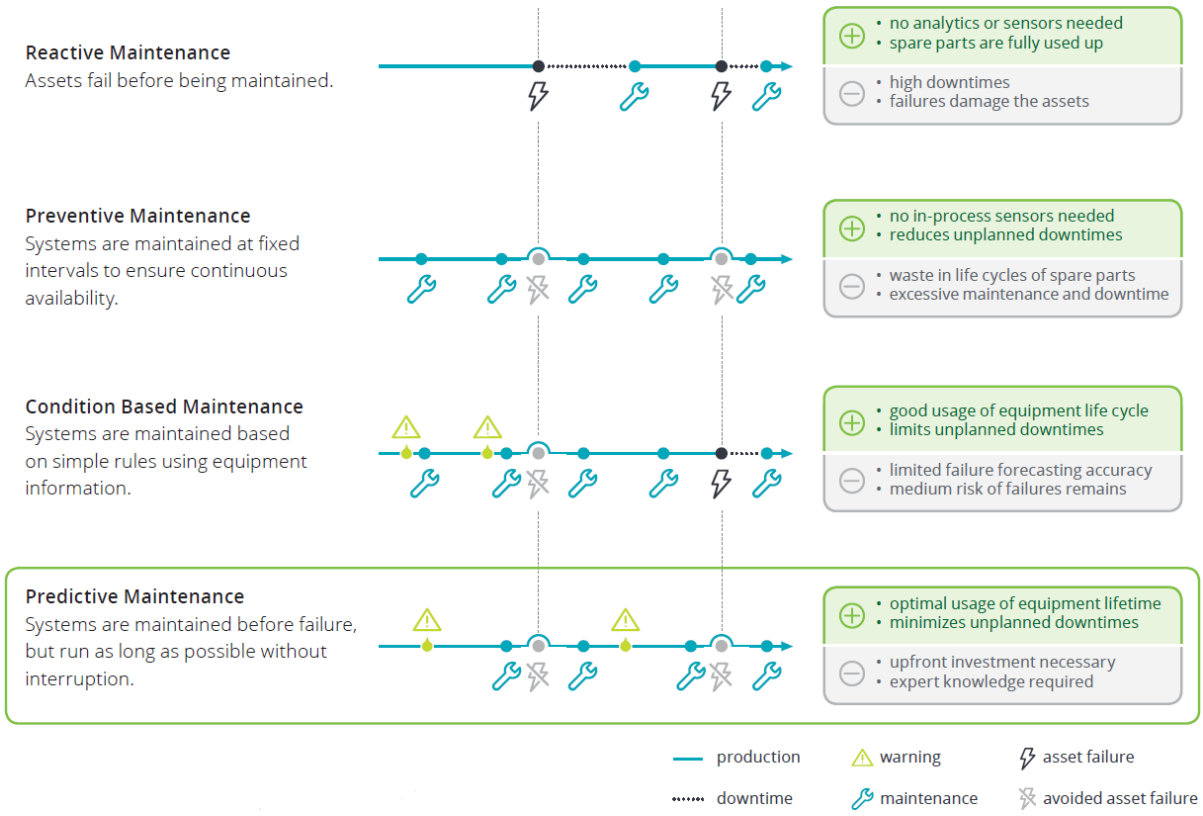

Figure 2.1.1: Overview - Types of Maintenance (Image retrieved from [47])

Industry 4.0 has given rise to the adaption of this PdM and more companies are shifting from a reactive approach towards this proactive approach [19]. However, it is important to note that the other forms of maintenance are not useless but in fact, in some cases still the best approach where PdM is not possible. For instance, refilling vehicle lubricants or tightening some hardware bolts [17]. Moreover, at times, for safety reasons or the type of action to be executed may require the equipment to be switched off. To minimize the impact on the business, in these cases the maintenance is planned and executed as quickly as possible [17]. This indicates another reason why companies tend to shift to predictive maintenance mainly for critical breakdowns and downtime which causes a significant financial impact.

2.1.2 Why Predictive Maintenance?

PdM increases overall productivity and has a positive impact on the business. The PdM lifecycle in Figure 2.1.2 shows the three core components of PdM in a spiral that is at the forefront of its success.

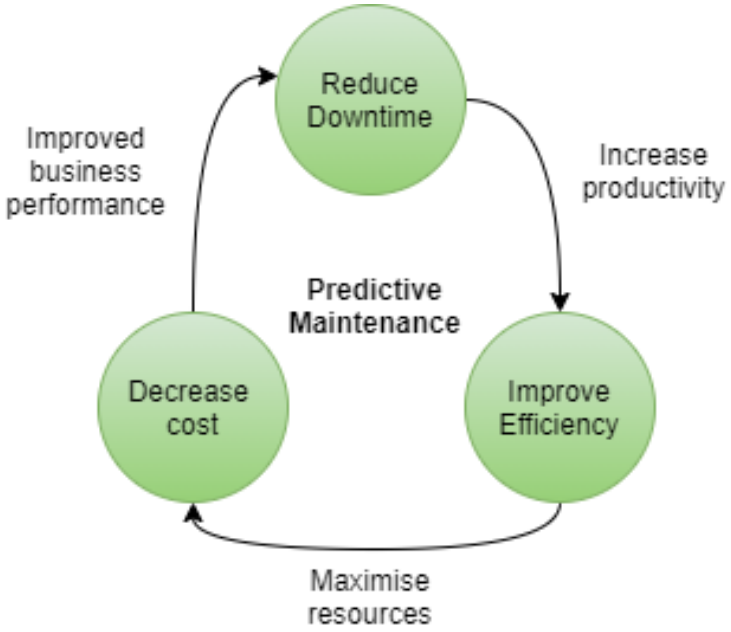

Figure 2.1.2: The Predictive Maintenance Cycle (Image adapted from [17])

Advantages of Predictive Maintenance

Advantages of PdM over the other aforementioned maintenance methods as follows [9]:

Predictive Maintenance	Other maintenance methods
• Lower maintenance costs as only the specific components which are forecasted to fail ahead of time are serviced/replaced. • Higher customer satisfaction as they can plan for any impending failures of their equipment ahead of time resulting in their projects to be executed on time as planned. • Real-time asset management with notifications to report any imminent failures.	• Higher maintenance costs as either the equipment has completely broken down or is inspected and serviced at regular intervals. • Equipment undergoes inspection/service/maintenance even when it is perfectly healthy as per pre-defined maintenance schedules. • There is no way of knowing ahead of time when a failure will occur unless the specific component undergoes inspection / service.

Challenges of Predictive Maintenance

Even with the great advantages of PdM, there exist some critical challenges in its application across industries. Mainly, there is a need for integration between the maintenance and analytics departments of any organization. They need to work together to make the transition towards PdM from their traditional reactive maintenance approaches. However, this is costly as possibly there is a need to hire analysts and maintenance technicians who are experts in this area of PdM [9]. Over the years, a lot of organizations have been collecting data but have not been leveraging it to enhance their products or have moved towards a data-driven approach. However, there seems to be a strong feeling amongst these organizations that they have the ability to easily transition and make use of this data. Unfortunately, often we see that they struggle to adapt as the data quality is not up to the standard to employ these advanced techniques [47]. Therefore, a big challenge the organizations are facing is to improve

their data quality essentially by modeling better the data they require and from which components of the equipment they need to collect it from [47]. They will need to install the right infrastructure to achieve this by employing the right people, installing the right software and a wide range of sensors across their different equipment which can relay data in real-time from the machines, identifying the critical failures, bottlenecks, and, areas of improvement [47]. Installing this infrastructure can result in the cost of equipment to increase for e.g. higher cost of sensors results in a higher cost of the vehicle for the customer. Some other concerns in recent times exist with government regulations on which data can be shared and the lack of defined standards in the industry makes this area more difficult to navigate [4, 9].

In conclusion, organizations will need to invest in transitioning their maintenance activities to the PdM approach which requires significant investment before they see the return on it. However, it seems to be a double-edged sword as organizations that are hesitant to invest in such innovative techniques will eventually get left behind by their rivals in the near future, while on the other hand, if they do decide to invest, they will need to invest a significant amount of capital to employ the expertise lay down the correct infrastructure to succeed.

2.2 Application of Predictive Maintenance

PdM in practice is dependent on the availability of data and its quality. Without the availability of data, PdM is not possible. It makes use of data along with predictive modeling approaches such as ML to predict the RUL of a component [52]. Once the model makes the prediction, typically a notification is sent to the maintenance team about the corresponding equipment who then contact the customer to schedule the maintenance. The following subsections provide a comprehensive outline of the requirements for creating such a system for construction equipment.

2.2.1 Log Data

Log data stores readings detailing the functioning of equipment via the software modules running on the equipment [44]. Each reading observation has a timestamp which is mapped to other features of the construction equipment for e.g. engine hours, distance travelled, operating hours, etc. (see figure 2.2.1). Each reading observation

is essentially an event and typically this kind of data is known as event-log data i.e. most common and widely available form of log data [44]. The frequency of these event notifications is determined by the data infrastructure set up by the company, often being reported in nearly real-time. The features which are included in these logs are the ones defined by the domain experts/developers in charge of the equipment [44]. In principle, these logs can be used to track how the equipment has been used over time and effectively make conclusions about its current state along with its overall health [44].

Similarly, sensor data is also another popular source to gather data for such equipment. Sensor data is often derived from each individual component installed on the equipment for e.g. the fuel level, voltage, brake oil temperature, etc [20]. Each reading observation in this data is derived via sensor readings corresponding to a specific date and time. It is also mapped to a timestamp, however, the features are time-based and/or distribution-based attributes.

Therefore, to enrich the data and make it more valuable, both event-log and component based sensor data can be used in a hybrid way by merging them together (time-based and distribution-based logs) [47].

Log data in practice

Typically, once we have loaded the log data, we identify which type of log data it is and whether we need to merge it with other log data sources to get access to for e.g. time-based and/or distribution logs and/or domain-specific logs. Once, we have all the log data at one source, then we perform some preliminary analysis and understand what the basic data looks like. We interpolate any missing values, identify key features and perform any feature engineering techniques if required. Then, we check whether the data is labeled or not in terms of failure/no failure. At this step, we already have an inclination on whether to use supervised or unsupervised learning methods for the prediction. Depending on the labels, we either make note of the outliers/anomalies or remove them. Finally, we prepare the dataset in a way that is ready to be used by ML methods.

Challenges of using Log Data

System logs are not always simple and easy to deal with it when it comes to predicting failure. Sometimes, the log data may not contain enough features which directly results in a failure or not enough failure observations popularly known as the class-imbalance problem [44]. Therefore, looking for patterns that can indicate when a failure may occur becomes more difficult. Moreover, there can be computational challenges during the pre-processing of the data due to its size if the data is logged in real-time [44]. Also, it can be difficult to identify outliers/anomalies at times due to the high-dimensionality of the data [12]. Lastly, for PdM, high data quality is of utmost importance without which it can be very challenging to achieve acceptable model performance [54].

2.2.2 Predictive Maintenance using Machine Learning

Predictive models are one of the key components on which predictive maintenance relies on [47]. These models are able to use the log data to predict the failure of a specific component, its health/condition, and its RUL [20]. Normally, to achieve this predictive maintenance systems make use of two popular machine learning techniques:

Classification

Classification algorithms are one of the most common and popular techniques used in ML across all domains. They are able to determine typical and atypical patterns from the historical data provided and based on that classify which group a certain observation belongs to [47]. It can be a Binary-class Classification (BCC) / Multi-class Classification (MCC) based on the data provided. In the domain of PdM, classification algorithms can be used in few different ways:

- BCC: Predict whether a machine is “likely to fail” or “not fail” [53].
- BCC: Predict whether a machine will fail or not in the next “x” days [20].
- BCC / MCC: Predict which failure/fault code will occur.

Typical algorithms used to perform such classification tasks are Random Forest (RF), Support Vector Machines (SVM), and K-Nearest Neighbours.

Anomaly detection

Another popular technique to identify any outliers/patterns which deviate from the data when the machine is working normally. This technique makes use of event and sensor log data generated by specific components at specific time intervals. These outliers can indicate specific types of failure and some recurring patterns can be identified to indicate whenever a failure is about to occur. This technique can be implemented using a single class SVM and autoencoder neural networks [47].

Infrastructure

To make use of these ML techniques in the domain of PdM, it is crucial to have the right infrastructure in place. In terms of data collection, having a streaming architecture can be really helpful. It ensures the availability of real-time log data of the equipment which can be used by the ML algorithms to predict failure [47]. The streaming architecture can be implemented using Apache Spark, Kafka, etc. However, the focus of our research will not involve this streaming architecture as the real equipment log data we have available from VCE is not collected in real-time.

2.3 Construction Equipment

Construction equipment is a vital product in the construction, mining or agriculture industry [49]. There is a broad variety of equipment that is required by projects in this industry for e.g. haulers, wheel loaders, excavators, etc. and is offered by VCE. However, for our research study, we will focus only on one specific type of construction equipment i.e. Articulated Haulers [50]. This is the type of equipment for which we have sufficient log data available and is one of the "star" [50] products of VCE with extremely high demand. It is one of the most sold VCE products around the world.

Figure 2.3.1: Articulated Hauler - A40G Series (Image retrieved from [51])

2.4 Related work

In recent times, with the rise of Industry 4.0, significant research has been done in the area of PdM [2]. These include using different data approaches (e.g. event-log data [4, 20, 44, 53], sensor data [61]), different domains (e.g. ATM machines [53], smart electric devices [4]), and different ML techniques (e.g. supervised learning classification models [11, 48], anomaly detection [41], unsupervised learning [13]).

2.4.1 Log Data and Machine Learning methods

Various different studies have been conducted using a different combination of log data and ML methods. A notable study is of NASA's aircraft engine [62]. The engine data has been collected using six sensors and has been used to predict the engine's

performance and its RUL [62]. Primarily, the study focused on using Long Short Term Memory (LSTM) to predict the failure and RUL of the engine. Moreover, it compared the performance of the LSTM with other ML models such as Support Vector Regression (SVR), Multilayer Sensor (MLP), and Deep Evolution Neural Network (DCNN) [2]. It concluded that the LSTM outperformed all the aforementioned models in predicting the RUL of the aircraft engine [2]. Other studies have also established that LSTM and Recurrent Neural Networks (RNN) models perform well on time series and sequential log data for PdM [2, 45, 56]. Even though these models perform well, there is a concern about their interpretability since they are black-box models. This can cause problems especially from the side of the technicians/maintenance team as they would want to know why they need to fix a component. Moreover, it is important to note according to the authors of this study [41] is that the use of data collected from sensors without any domain knowledge or expert input can lead to incorrect predictions. Therefore, it is crucial to ensure that the data quality produced by the sensors is high and there is input from experts [41].

White-box classification models have also been a popular way to predict RUL. Ensemble tree-based methods have been used such as RF [38] and Random-Survival Forest (RSF) [25] respectively. RF has been used to predict the RUL for vehicle compressors using log and service data which allows them to plan their visits to the maintenance workshop accordingly [39, 52]. Similarly, RSF is a technique that can be used to predict the probability of the component being functional at a particular date and time in the future. It does this by allocating components into groups based on the similarity of the patterns [52]. The similarity could be based on depreciation indicators of the equipment, the health of the equipment, or any other correlated patterns. In the case of construction equipment, the model could group equipment with similar patterns together. Furthermore, ensemble classification algorithms such as RF have been used with event-log data to predict the breakdown of ATMs [53], smart electrical devices [4], and discrete parts manufacturing [20] to reduce the unexpected downtime. These approaches have been successful with event-log data, however, do not make use of a hybrid data approach to merge event and sensor data together.

2.4.2 Non-machine learning methods

Our study is only going to be making use of ML methods for PdM, however, it is important to note that there exist other methods to perform PdM as well. One of the earliest methods in the field of PdM are non-ML rule-based expert systems which have been heavily adopted by the industry [6, 57]. The reason is that the research in these rule-based systems started as early as 1996 and precisely required expert knowledge which most organizations possess [6, 57]. The rules/patterns for failure are pre-defined in these systems and a notification is triggered once the specific pattern is encountered in the data. However, these rules are dependent on experts themselves and are not maintainable for modern large-scale applications due to the exponential increase of defining rules and hiring experts [53]. It is simply not viable or scalable, however, it can play a role in specific small-scale applications.

Moreover, some statistical methods have also been investigated in the area of PdM. The prominent ones are sequential pattern mining [10, 34, 60], survival model, and Cox model [53]. Sequential pattern mining relies on the strong association which exists in the log data of the equipment [10, 34]. It identifies the statistically significant patterns in this sequential data, however, is unable to make use of the time component [10, 34]. Therefore, it is unable to capture the effect of time series data and is not able to forecast time-based failure prediction. It is one of the major drawbacks of this method. Alternatively, both the survival and Cox model are able to handle the time component and are able to forecast how long before the failure occurs [30, 36, 40]. This method makes use of the failure data, however, all predictive features are still not captured as it does not make use of error logs [30, 36, 40]. So the different statistical methods seem to have their own significant drawbacks, however, they can be deemed quite useful depending on the type of data and computational resources you have at your disposal.

With the improvement in technology, big data, and IoT systems, more modern methods were developed and adapted [2]. An example would be a stand-alone Autoregressive Integrated Moving Average (ARIMA) model or a combination of a Deep Learning (DL) model with the ARIMA model [2, 16, 23]. Both of these techniques have been used for making PdM forecasts [16, 23]. Unfortunately, the studies have shown the ARIMA models have not proven to be as powerful and accurate as it's machine learning counterparts in terms of predictive performance [2]. Therefore, we only focus

on the cutting-edge data-driven ML models for the purpose of our study.

2.4.3 Conclusion

In conclusion, a lot of this research has been using open public datasets and not actually data that is originating directly from the machines themselves [63]. Moreover, these works mostly only make use of event-log data and do not use a hybrid data approach (event + sensor log data). Furthermore, a survey of 150 papers was conducted in the field of data mining in manufacturing and concluded that failure prediction in the automotive industry is more difficult to do than in other domains as it is difficult to do continuous monitoring in real-time [8, 39]. Therefore, we must also note that instead of predicting failure for a specific type of equipment, it is more viable to predict failure for the specific component within that equipment as the equipment is made up of numerous components and predicting all faults with high accuracy using a single model is extremely difficult [44]. Lastly, we see that rarely any PdM research has been done related to construction equipment.

Chapter 3

Method

3.1 Choice of research method

We made use of the quantitative research method to answer the defined research question for this project. This decision was made based on the notable previous research work done in this area which also makes use of this method as highlighted in Chapter 2. This is due to the fact that there is the availability of a vast amount of equipment data and well-grounded machine learning algorithms whose predictive performance can be measured using metrics that are widely recognized. This applies to our case as well, as we had real-world log data directly retrieved from VCE's articulated haulers which could readily be used towards carrying out our research. Moreover, we had the computational resources to build machine learning models on this data to predict failures ahead of time. Lastly, the expertise that was available at our disposal at VCE along with their years of experience working with the data sources further supported us to go with this approach. This went hand-in-hand with the expectation of VCE i.e. to build a predictive maintenance machine learning model whose quality can be measured quantifiably and can act as a proof of concept which can be scaled into their products and portfolio.

Having said that, a qualitative approach could also have been carried out by interviewing industrial experts and academics who have worked with such predictive maintenance machine learning models in theory and practice for instance at VCE and KTH Royal Institute of Technology (KTH) respectively. However, such a study requires considerable time and human resources to establish concrete results and conclusions.

Therefore, due to time constraints and lack of sufficient resources, we did not proceed with the qualitative approach.

3.2 Application of research method

When we talk about data science, there is always a big focus on ML techniques and algorithms. However, these ML algorithms are just a cog in the entire process that is required to carry out a quantitative data science project [19]. A popular methodology to carry out data science projects is known as the CRoss Industry Standard Process for Data Mining (CRISP-DM) [55] methodology which is widely used in both industry and academia.

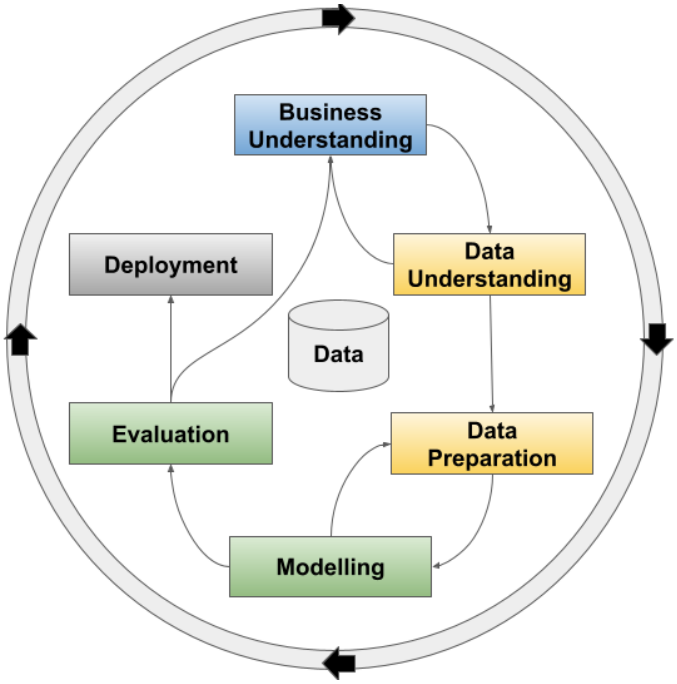

Figure 3.2.1: Phases of the CRISP-DM Model for Data Science (Image adapted from [55])

However, to carry out predictive maintenance projects in the realm of data science, SAP [19] has concluded that this methodology needs some amendments which need to be specified even though they may actually be applied in practice in CRISP-DM. Precisely, two additional steps:

1. Integrate domain expertise of the problem at hand into the iterative CRISP-DM lifecycle.
2. Monitor the ML model scoring and results during and after deployment.

Therefore, to apply the aforementioned quantitative research approach, we made use of an amended CRISP-DM methodology proposed by SAP [19].

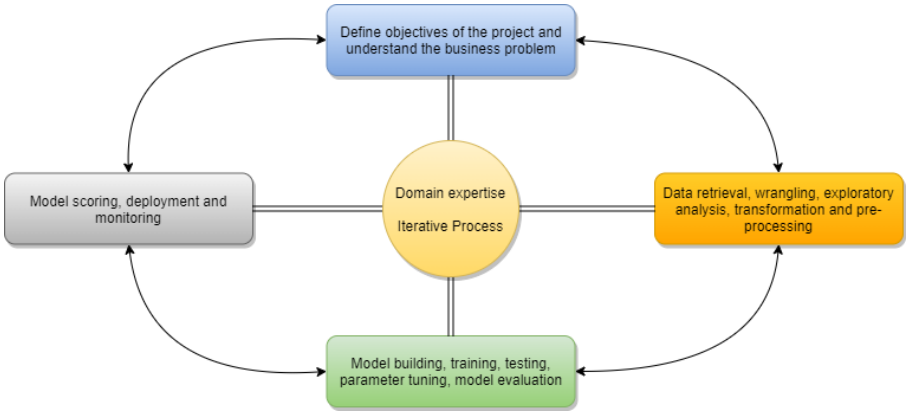

Figure 3.2.2: Amended CRISP-DM Data Science Process for Predictive Maintenance (Image adapted from [19])

In the last step of the methodology, we did not carry out the deployment part as it is not in the scope of this research project. Our research and the resulting model are aimed to act as a proof of concept which VCE can then build upon and deploy as part of their future work.

3.2.1 Domain expertise

Our research work has a broad domain relating to maintenance, construction equipment and log data. Therefore, to gain expert insight on existing and future work along with challenges VCE faces in these domains, we spoke to the people working across different departments in these areas at VCE. Table 3.2.1 outlines the experts role and the department they belonged to.

Table 3.2.1: List of departments and experts contacted

DEPARTMENT	EXPERTS
Advanced Engineering and Emerging Technologies	Research Engineers, Industrial PhDs
Analytics	Development Engineers, Technology Domain Leaders
Electronics and Embedded Systems	Senior Verification Engineer, Test Engineers
Uptime Center and Warranty	Head of Department, Thesis Students

After speaking to them, they pointed us in the right direction on how to make use of the

resources available at our disposal to carry out the research work. The following areas were discussed at length throughout the course of the project and the key takeaways are summarized along with the support the experts offered.

Construction Equipment and Maintenance

Most of the popular products sold in the market by VCE are wheel-loaders and articulated haulers. Therefore, the bulk of customer support and maintenance activities are related to these machines. Consequently, a high amount of data is available for these machines. These machines can have different types of failures on different types of components. Some failures can be critical which can result in the breakdown of the machine which means it comes to a halt. While other failures may not be of critical nature and may not need to be fixed immediately. Moreover, predicting a specific failure for the entire machine is quite a challenge. Therefore, it was recommended to go with a bottom-up approach:

1. Identify a specific component for which we can predict failures for e.g. engine, transmission, etc.
2. Identify specific types of failure to predict belonging to that component, which if were to occur may result in critical breakdown or high costs to fix.

The active care and uptime center provide warranty, repair services, and support to their customers. They perform a mixture of reactive and preventive maintenance of the machine depending on the components. With the help of PdM, they can manage the stock for replacement parts optimally and can plan to prevent failures and service the machine ahead of time.

Log Data

The data sources that are appropriate to our research work, how to access them along with the relevant tables, their structures and schema respectively. Quality of the log data available in these data sources and their limitations were discussed at length. The keys findings were that the data is not collected in real-time, only when the technician connects to the machine. Therefore, there is a logged timestamp and send out timestamp respectively. Moreover, the data is temporal, with timestamps at unequal intervals. Also, the log database was initially designed for reporting purposes, therefore, some of the data is pre-aggregated. Furthermore, the previous work done at

VCE using this log data and the challenges they faced from a practical standpoint were discussed and we were given access to detailed documentation of the log-data.

In conclusion, the domain experts' input helped us shape our research project and to avoid pitfalls.

3.2.2 Data sources

Data tables pertaining to event and sensor logs respectively will be retrieved from the VCE database. The details of all the data will be specific to Articulated Haulers particularly to all vehicles belonging to the A40G series. Table 3.2.2 and 3.2.3 below outlines the data tables belonging to each of these categories.

Table 3.2.2: List of data tables containing Event-Log Data

TABLE NAME	TOTAL OBSERVATIONS	TOTAL FEATURES
READING	49,660	11
FAULTCODES	887,999	17
VEHICLE	5,603	13

Table 3.2.3: List of data tables containing Sensor-Log Data

TABLE NAME	TOTAL OBSERVATIONS	TOTAL FEATURES
ECU PARAMETER	N/A	6
TEA2 READING	N/A	5

The data available in these tables is structured and labelled data.

3.2.3 Imbalanced data

When working with failure and maintenance datasets, class imbalance is extremely common. Formally, class imbalance is defined as a dataset where one or more classes have a much larger number of instances than the other classes. The most occurring class is known as the majority class, whereas the class with the least instances is known as the minority class [31].

Naturally, all equipment has more instances recorded of when it is functioning normally as opposed to when it has failed. Failure is considered to be a rare event i.e. an event which occurs less frequently than what occurs commonly [21, 35]. In the realm

of data mining and machine learning, this is considered to be a binary classification problem of predicting whether a specific event has occurred, in this case, whether a failure has occurred or not [21].

Traditional supervised classification algorithms do not perform as well on imbalanced data as they do on balanced data. For instance, Logistic Regression (LR), SVM, Decision Trees (DT) provide substandard results for imbalanced data as they classify the majority class correctly, however, the minority class is classified incorrectly most of the time [28]. Moreover, performance metrics such as accuracy are biased towards the majority class and can be quite misleading as the high accuracy might seem to you that the model is good but its only classifying the majority class correctly most of the time [33]. Furthermore, some minority class instances can be considered as anomalies or outliers by the models as they occur rarely [3].

Even though these problems exist, the aforementioned algorithms can still be used to build a model on such imbalanced data after adjusting the data and making tweaks to it using some recommended approaches. Some of these approaches include resampling techniques, feature selection, ensemble models, different performance metrics, etc. We incorporate these approaches into our method and discuss them in the sections ahead as the data we are dealing with is imbalanced.

3.2.4 Data Pre-processing

To prepare the data, a combination of different data processing techniques will need to be used. Here we highlight some of the key ones.

Missing Value Imputation

Real world data more often than not has missing data due to a variety of reasons. Some of them may involve issues with the collection process, incorrect data entry, the database system or network itself, etc [32]. This missing data can be interpolated using a variety of different techniques ranging from mean imputation to making a prediction for the imputation using predictive models such as regression. The most commonly used are mean, median, mode, min, max imputation techniques. Moreover, for time series data rolling average, neural networks, ratio and windowing techniques are also used [58]. The imputation technique largely depends on the data you have. Since our data involves time series of vehicles with continuous values increasing over time, we

will make use of the windowing and average techniques.

One-hot encoding

One-hot encoding is the go-to technique to prepare categorical data for machine learning models [43]. For each category, the encoding technique creates a binary vector, with labels one/zero indicating whether this category exists for a certain observation or not [43]. For x categories, it creates x binary vectors. These binary vectors act as features to the ML model and often may contribute significantly to the prediction. In our case, this technique will be crucial to prepare the vehicle ID's and historical failure features for the ML model to take as some of the inputs.

Resampling

The datetime interval frequency of time series data can be changed using the resampling technique. The resampling technique can make the interval between each consecutive observation equal by aggregating the data and assigning it to the next equal interval value. It can be divided into two categories:

1. Downsampling

   This approach decreases the frequency of the data by aggregating the values.

2. Upsampling

   This approach increases the frequency of the data by increasing the number of observations.

This approach is useful to handle class imbalance that can affect model performance. Therefore, it is one of the approaches which we will experiment with to see how its performs in comparison to other approaches and whether losing data has an impact on model performance while observing the effect of resampling on class imbalance as well. In our case, we will make use of a custom resampling technique using a windowing approach as we want to see the failure ahead of a certain period of time. The application of this technique is explained further in subsection 5.1.2.

Feature Selection

Feature selection is a widely recognized and effective technique to prepare data for ML tasks. Features which do not contribute to the model, are redundant or have anomalies

should be dropped [29]. In our case, the features pulled from the tables mentioned in Table 3.2.2 will be filtered out based on their relevance to our task and whether they contribute to the PdM model or not.

3.2.5 Models

The log data is tabular with a binary class label indicating whether there is a failure or not. Therefore, we used a supervised learning approach for our research work and selected the leading classification algorithms.

Logistic Regression

LR is a predictive modelling method which uses data to explain the relationship between a categorical dependent variable and one or more independent variables [27]. It is typically used for binary classification problems and predicts the class label with a certain probability. If the probability of prediction of a certain label is above the defined threshold which by default is 0.5, then that label is predicted [27]. It is one of the most common and popular techniques used in the area of classification.

Random Forest

RF is an ensemble supervised learning method which can be used for both classification and regression problems [38]. It is an ensemble method as it makes use of a vast number of decision trees to predict the class label and the class label with the highest number of predictions gets assigned as the final prediction [46]. Essentially, it takes the majority prediction made by the defined number of trees. This approach of choosing the highest voted predictions from the trees gives good results. Therefore, it outperforms an individual decision tree model.

XgBoost

XgBoost is a scalable distributed tree boosting algorithm which is in recent years has been used for state-of-the-art machine learning models and has also gained a lot of popularity in Kaggle competitions owing to its success [7]. Scalability and the algorithms ability to handle highly sparse and dimensional data is one of the key reasons for its success [7]. Most real world data sets are often sparse or highly dimensional for e.g. biological data, text classification, etc, this algorithm is of great

use to solve such problems. The algorithm is implemented using gradient boosting trees [7].

3.2.6 Model Evaluation

The performance of the models mentioned in section 3.2.5 can be evaluated using some well known metrics in the ML community [21]. The selection of the metrics was influenced by the fact that the data has a significant class imbalance. In view of the class imbalance, we chose multiple metrics which are not affected by it and are known to perform well to evaluate the performance of our models [5]. To understand these metrics, we first define the confusion matrix for a binary classification problem.

The confusion matrix helps us to understand deeply the predictive performance of a ML model by showing us which precisely how many times a class is being predicted correctly, incorrectly and what errors the model is typically making. It also introduces us to terms that are used to define performance metrics which we have used in our study. The terms are as follows:

Class Label	Predicted: Negative (0)	Predicted: Positive (1)
Actual: Negative (0)	True Negative (TN)	False Positive (FP)
Actual: Positive (1)	False Negative (FN)	True Positive (TP)

Figure 3.2.3: Confusion matrix for a Binary Classification Problem

- True Positive (TP): Instance label is positive and is classified correctly by the model as positive.
- True Negative (TN): Instance label is negative and is classified correctly by the model as negative.

- False Negative (FN): Instance label is positive and is classified incorrectly by the model as negative.
- False Positive (FP): Instance label is negative and is classified incorrectly by the model as positive.

The selected metrics to evaluate model performance are as follows:

1. **Accuracy**

   Accuracy is the most widely used metric for classification tasks [21], however, as mentioned previously in section 3.2.3, accuracy performs poorly for imbalanced data due to a bias towards the majority class and may not be the optimal metric in our case [5]. However, we will still keep it for comparison with other performance metrics and whether they conflict or validate the performance of a specific model in comparison to other performance metrics.

   For binary classification problems, accuracy is defined as the ratio of correct predictions to the total predictions the model makes. It can be denoted as follows:

$$Accuracy = \frac{CorrectPredictions}{TotalPredictions} = \frac{TP + TN}{TP + TN + FP + FN}$$

2. **Precision**

   Precision is the ratio of positive predictions that were correct. In other words, precision is the ratio of true positive labels to the total positive labels. It can be denoted as follows:

$$Precision = \frac{TP}{TP + FP}$$

3. **Recall**

   Recall tells us how well the positive classes were predicted. It is the ratio of positive cases that were correctly classified. In other words, recall is the ratio of true positive labels to the total positive predictions. It can be denoted as follows:

$$Recall = \frac{TP}{TP + FN}$$

4. **F1-score**

   F1-score is the harmonic average of the precision and recall. The F1-score value lies between zero and one, where one is when all instances are predicted correctly and zero is when all instances are predicted incorrectly. It can be denoted as follows:

$$F1score = \frac{2 * TP}{2 * TP + FP + FN}$$

3.2.7 Software Selection

To carry out this research work, we used the following software:

- **Programming Language:** Python 3.8+

  Python is the most preferred programming language for data science projects. It has numerous third-party libraries for data pre-processing and to build machine learning models. Moreover, it has a simple syntax, easy to learn and a very active community. Therefore, it is quite popular amongst data scientists and has an excellent reputation for ML projects. The aforementioned reasons fulfill the demands of our research work which made it the optimal choice for our project.

- **Notable Libraries:** pandas, numpy, scikit-learn

  These libraries are popular for data pre-processing and building ML models in python. They are quite extensive and were sufficient for the tasks required in our project.

- **Programming Environment:** Jupyter Notebook and Visual Studio Code.

  Jupyter notebook was used to perform quick dirty analysis of the data, build baseline models and to come up with a rough solution of our research question. Once, we were confident of our solution, we used Visual Studio Code to modularize the code in a way that is scalable and generalizable so that it can be deployed by VCE if they choose to do so.

- **Data Sources:** Logged Data Analysis (LDA) Data Warehouse and EDW Tool

  Both data sources were VCE's in-house data warehouse and analysis tools respectively. The LDA warehouse was accessed using DBeaver software and the

data was exported as .csv using SQL statements. The EDW tool is a click-based report generation tool and had it's own export functionality to generate data as .csv.

Chapter 4

Data Preprocessing and Exploration

In this section, we show how the research method was applied practically and what work was done to prepare the data in different ways which was then used to build the PdM ML models.

4.1 Data Retrieval

The historical data for the Articulated Haulers A40G series was retrieved from the data sources mentioned in subsection 3.2.2 and 3.2.7. The event-log and component-based sensor log data were retrieved from different databases respectively. To retrieve this data, different queries and merging of tables was required.

4.1.1 Event Log

The event log data was available in the in-house LDA warehouse. All this data was retrieved using SQL statements and was exported as comma-separated files.

- Reading table

  Contained the readings taken of vehicles at a given datetime distinguished by a unique reading ID. This table was joined with the vehicle table to map the vehicle information to the reading observations and then exported as comma-separated files.

Table 4.1.1: List of features belonging to the READING Table

FEATURE NAME	DESCRIPTION	DATA TYPE
READING_ID	Unique ID to identify reading	INTEGER
VEHICLE_ID	ID to identify vehicle for which the reading is taken	INTEGER
INCOMINGDATE	Timestamp of when the reading is recorded in the system	DATETIME
READOUTDATE	Timestamp of when the reading is recorded by the vehicle ECU	DATETIME
LAM_OPERATINGHOURS	Total hours the vehicle has been in operation	FLOAT
LAM_TOTALDISTANCE	Total distance travelled by the vehicle	FLOAT
LAM_NUMBEROFPARAMETERS	Number of logs associated with the vehicle	INTEGER
CUSTOMER	Customer who the vehicle belongs to	INTEGER
SENDER	VCE data source which records the data	STRING
ISPROTOTYPE	Indicates whether the reading is for a real or prototype vehicle	CATEGORICAL
SENDTIME	Timestamp of when the reading is sent to the system	DATETIME

Table 4.1.2: List of features belonging to the VEHICLE Table

FEATURE NAME	DESCRIPTION	VALUES
VEHICLE_ID	Unique ID to identify a vehicle	INTEGER
CHASSISSERIES	ID to identify which equipment series the vehicle belongs to	STRING
CHASSISNUMBER	Unique ID to identify the vehicle	INTEGER
COUNTRY	Country in which the vehicle is under operation	STRING
MODELID	ID to identify the model of the vehicle	INTEGER
DEALERID	ID to identify which dealer sold this vehicle	INTEGER
NUMBEROFREADINGS	Total number of readings taken for the vehicle	INTEGER
PRODUCTCLASS	ID to identify which Volvo company the vehicle belongs to	INTEGER
REGISTRATIONNUMBER	Registration number of the vehicle	INTEGER
PROTOTYPE	Indicates whether the reading is for a real or prototype vehicle	CATEGORICAL
SPECDATE	The date when the vehicle was ordered by the customer	DATE
PRODDELDATE	The date when the vehicle was delivered to the customer	DATE
ISACTIVE	Indicates whether the vehicle is still in operation	CATEGORICAL

- Faultcodes table

  Contained the failure data of vehicles at a given datetime distinguished by a unique faultcode ID. This table was also joined with the vehicle table to map the vehicle information to the faultcode observations and then exported as comma-separated files.

Table 4.1.3: List of features belonging to the FAULTCODES Table

FEATURE NAME	DESCRIPTION	VALUES
FAULTCODES_ID	Unique ID to identify the failure	INTEGER
VEHICLE_ID	ID to identify which vehicle has undergone a specific failure	INTEGER
SENDER	VCE data source which records the data	STRING
FAILURECOUNT	Total number of failures	INTEGER
FIRSTFAILURETIME	First time a failure occurred for a vehicle in a specific window	DATETIME
LASTFAILURETIME	Last time a failure occurred for a vehicle in a specific window	DATETIME
ENGINEHOURS	Total hours the engine has been under operation	INTEGER
COUNTRY	Country in which the vehicle is under operation	STRING
NUMBEROFREADINGS	Total number of readings taken for the vehicle	INTEGER
PROTOTYPE	Indicates whether the reading is for a real or prototype vehicle	CATEGORICAL
ISACTIVE	Indicates whether the vehicle is still in operation	CATEGORICAL
NODENAME	ID to identify which ECU of the vehicle has the failure occurred in	INTEGER
DIAOBJTYPE	Diagnostic object type to identify parameters and subsystems	STRING
FAILURECOMP	ID to identify which component of the vechile has failed	INTEGER
FAILURECOMPDESCRIPTION	Describes the failure component	STRING
FAILURETYPE	ID to identify type of failure	INTEGER
FAILURETYPEDESCRIPTION	Describes the failure type	STRING

Note that only failures related to engines were retrieved as our research work only focuses on PdM for the engine of the vehicles and also any observations where the failure component was null were ignored. Moreover, this table did not have a common

ID with the reading table and later we will be merging these tables together but not at the time of retrieval.

4.1.2 Sensor Log

The sensor log data was available in the EDW in-house click based report generation tool. The data was retrieved by constructing queries in the tool and exporting the output tables as comma-separated files. A total of three sensor logs were identified to be logging sensor data related to engines and were retrieved. Note, for data privacy purposes, the respective log output details are kept hidden.

- Engine coolant temperature

  Records the spread of the engine coolant temperature when the engine is running. This is a distribution log and the data is measured in an incremental way that is stored in a vector with eleven values.

Table 4.1.4: Coolant temperature log output format

Coolant temperature [°C]	<A	A - <B	B - <C	C - <D	D - <E	E - <F	F - <G	G - <H	H - <I	>I
Value No.	xxx	xxx	xxx	xxx	xxx	xxx	xxx	xxx	xxx	xxx

  - Value No. stores the duration of the log and the time for which the coolant temperature is within the defined interval.

- Engine oil pressure

  Records the spread of oil pressure within a small rpm and temperature area. Similar to the previous log, this is also a distribution log and the data is measured in an incremental way that is stored in a vector of eleven values.

Table 4.1.5: Oil pressure log output format

Oil Pressure [bar]	A - <B	B - <C	C - <D	D - <E	E - <F	F - <G	G - <H	H - <I	I - <J	>=J
Value No.	xxx	xxx	xxx	xxx	xxx	xxx	xxx	xxx	xxx	xxx

  - Value No. stores the duration of the log and the total time for which the oil pressure is within the defined interval.

- Fuel consumption

  Records the fuel consumption while the engine is on. This is a time-based log and the data is measured in a temporal way that is stored in a vector of four values.

Table 4.1.6: Fuel consumption log output format

Value No.	**A**	**B**	**C**	**D**
Logged Value	xxx	xxx	xxx	xxx

  - Logged value stores the time and fuel consumption component of the log.

If any of the aforementioned logs encounter erroneous data, as a defined rule, the system does not log that data. Therefore, we can assume that the sensor log data we have retrieved does not contain any incorrect data.

4.2 Data Quality, Cleaning and Preparation

Once the data had been retrieved, the comma-seperated files were loaded into Jupyter Notebook and were ready to be explored using python.

Before we started the exploration, we had to keep in mind that the data we were dealing with was real equipment data collected in the industry as oppossed to simulated data. Typically, the real world data is quite messy and requires significant exploration. With this in mind, considerable time and resources were spent on preparing the data and ensuring that the data prepared for the ML models is of high quality.

4.2.1 Reading Table

The table 4.1.1 was retrieved and some basic summary statistics were explored prior to pre-processing.

Table 4.2.1: Basic reading table statistics

MODEL	TOTAL VEHICLES	TOTAL READINGS
A40G	3,756	45,332

Table 4.2.2: Readings per vehicle statistics

MEAN	STD	MIN	25%	50%	75%	MAX
12.06	23.38	1.00	2.00	6.00	12.00	237.00

It is interesting to note that the median number of readings per vehicle is only six. The frequency of readings can be better understood by average age of the vehicles to-date.

Table 4.2.3: Vehicle age statistics in years

MEAN	STD	MIN	25%	50%	75%	MAX
1.93	1.59	0.03	0.55	1.60	2.93	6.99

The median age of the vehicle is 1.6 years. This shows that half of the vehicles are relatively new, however, the number of median readings still indicates that the reading is not taken frequently enough. Definitely not in real-time.

We proceeded to explore the data and performed some data cleaning and feature creation.

Data Cleaning

- Duplicate Data

  Duplicate readings were found for some vehicles at the same datetime where all the feature values were identical except the operating hours. For each of these duplicate readings, one reading had a real value for operating hours while the other logged it as zero. The duplicate readings with zero operating hours was removed as this was considered to be incorrect data.

  Moreover, there were other duplicate readings where for the same vehicle there was more than one reading at the same datetime. These duplicate readings for the same datetime were also removed and only the last occurrence of that reading was kept as it was the latest reading data for the vehicle at that specific datetime.

- Missing Data and Interpolation

  For some vehicles, the country the vehicle is being operated in was null, but these vehicles still had corresponding observations in the reading table. Therefore, instead of losing reading data by dropping these vehicles, a new category ”Unknown” was created for these vehicles and the missing countries were filled in accordingly.

- Features dropped

  In table 4.1.1, there are three different datetime features for a specific reading namely ‘INCOMINGDATE’, ‘READOUTDATE’, and ‘SENDTIME’. After exploring these dates, we found that ‘READOUTDATE’ is the one which records the reading at the earliest datetime while ‘INCOMINGDATE’ and ‘SENDTIME’ record the same reading at a later datetime. This behaviour is as expected, as the ‘READOUTDATE’ is the datetime when the vehicle ECU records the reading while the other two datetimes are for when the reading is transferred to and recorded by the database respectively. Therefore, we drop the ‘INCOMINGDATE’ and ‘SENDTIME’ features. We only make use of the ‘READOUTDATE’ in our analysis and moving on we refer to it as ‘DATE’.

Feature Creation

- Class Label

  Each reading for a specific vehicle at a specific datetime is considered as a ‘healthy’ vehicle reading. In other words, that vehicle has no failure at that datetime as long as there is not a failure observation at the same datetime for that vehicle. Using this approach, a class label column is created called ‘FAILURE’ with all healthy vehicle readings assigned the value of zero.

4.2.2 Faultcodes Table

Table 4.1.3 was retrieved and some basic summary statistics were explored prior to pre-processing.

Table 4.2.5 shows the total number of failures for 3,756 vehicles where the failure component is known. The failure number is high as the data is from 2015 to-date across all ECU components. Also, the total failures also consist of non-critical failures

Table 4.2.4: Dataset Excerpt: Pre-processed reading table features output format

DATE	READING ID	VEHICLE ID	FEATURES	FAILURE
2020-08-25	987xxx	123xxx	...	0
2020-12-30	988xxx	123xxx	...	0
2021-04-12	999xxx	123xxx	...	0

Table 4.2.5: Basic failure table statistics

MODEL	TOTAL VEHICLES	TOTAL FAILURES
A40G	3,756	327,840

which tend to occur more often than critical failures which cause breakdowns. If these non-critical failures are not fixed, they are logged by the system repeatedly on different dates until they are fixed. Therefore, the statistics shown in 4.2.6 are quite high and skewed.

Table 4.2.6: Failures per vehicle statistics

MEAN	STD	MIN	25%	50%	75%	MAX
89.76	71.44	1.00	39.00	75.00	120.00	600.00

We chose engine as the specific component for our study as any engine failure is typically considered to be critical, and we filtered the failure data only for those as shown in Table 4.2.6.

Table 4.2.7: Basic engine failure table statistics

MODEL	TOTAL VEHICLES	TOTAL READINGS	TOTAL ENGINE FAILURES
A40G	978	7759	3,231

For the A40G series, the engine failures approximately equate to 1% of total failures and about 25% of the vehicles undergo them at least once in their life time.

The maximum value of 25 engine failures might seem surprising at first, but at a closer look, it exists because repeated failures related to a specific type of engine failure were not fixed which were being logged repeatedly over the course of the

Table 4.2.8: Engine failures per vehicle statistics

MEAN	STD	MIN	25%	50%	75%	MAX
3.30	3.10	1.00	1.00	2.00	4.00	25.00

vehicle's operations. The engine failures can be categorised as follows along with their occurrence frequency.

Table 4.2.9: Engine failure categories with frequency

FAILURE COMPONENT	TOTAL FAILURES
Engine coolant level	2,260
Fuel delivery pressure	545
Water in fuel indicator	270
Engine coolant temperature	67
Engine oil level	36
Engine oil temperature	29
Engine oil pressure	24

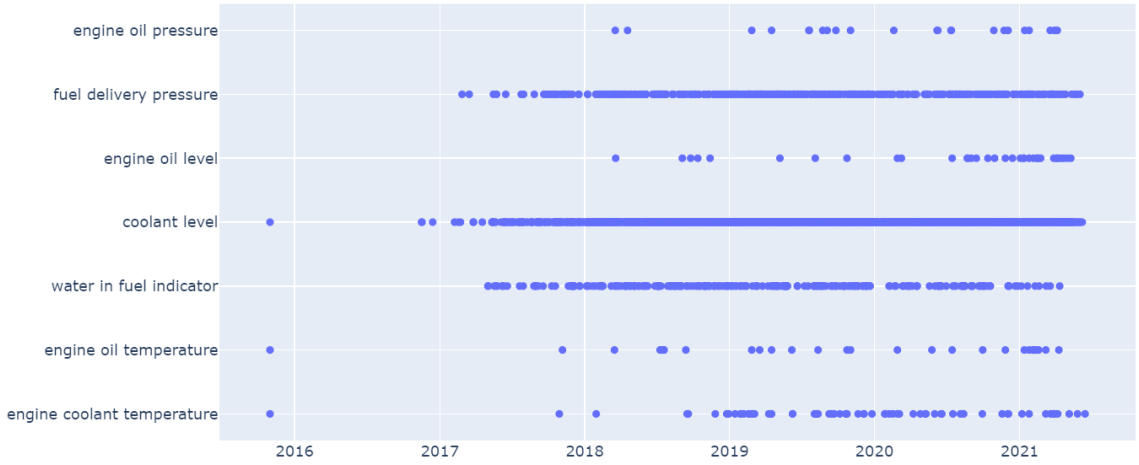

Figure 4.2.1: Engine failures by category over time

We proceeded to further explore the data and performed some data cleaning and feature creation.

Data Cleaning

- Vehicle Selection

  Before exploring the data further, we only select those vehicles with failure data

for which there exists at least one reading in the reading table. In other words, we only select vehicles such that they have at least one reading and one failure. This decision is made to reduce any chance of bias by the ML model towards a specific class by removing vehicles for which only one or the other class exists. After applying this filter, we find that there are 3,659 A40G vehicles for which there exists at least one reading and one failure.

- Duplicate Data

  Duplicate failure observations were found for some vehicles at the same datetime where the faultcode ID was the same. These duplicate observations were removed and only the last occurrence of the duplicate observation was kept as that was the latest failure data for that vehicle at that specific datetime.

  Moreover, any vehicles which had duplicate failure observations for the same component at the same datetime were also removed.

- Features dropped

  'SENDER', 'PROTOTYPE', and 'ISACTIVE' features were dropped as each of these columns had the same consistent value throughout all observations respectively. These features will not contribute anything to the ML model.

- Incorrect Data

  There were some failure observations where the datetime for the failure was from the future for e.g. 2023, 2048, 2081, etc. This data is obviously incorrect and was removed. Moreover, we only considered data from 2015 onwards as the data logging prior to that is not of the same standard or consistency.

Feature Creation

All features were created for every vehicle individually as each vehicle had a different history. In data terms, each vehicle has a unique time series with different types and number of engine failures.

- Class Label

  The failure data is being logged by the system in an incremental way. Each time a failure occurs for the specific vehicle, the 'FAILURECOUNT' feature is incremented by one. This is also why the 'FIRSTFAILURETIME' feature exists.

However, we re-assigned these labels and transformed the data as we want the data to be recorded as an isolated event instead of an incremental one. Therefore, the entire 'FAILURECOUNT' was assigned the value of one indicating that everytime a failure observation is recorded, it takes the class label one. This column was also renamed to 'FAILURE'. While the 'FIRSTFAILURETIME' column was renamed to 'DATE'. This re-assignment and re-labeling facilitated the concatenation of the reading (non-failure) and faultcodes (failure) data as shown ahead.

- Historic Failures

  For each failure observation in the faultcode table, the failure component description indicates which component and what fault it has undergone for e.g. the failure component description is 'engine coolant temperature'. This indicates that there is an issue with the engine coolant temperature for that vehicle at that specific datetime. To introduce lagged historic failure for each vehicles time series failure data, we created binary dummy variables for each failure component description and then lagged them by one. Therefore, for each vehicle, at any specific datetime, we have the last component failure it underwent. The features created in our case were for the six unique engine failures which exist.

  - H1: HISTORIC_WATER_IN_FUEL_INDICATOR
  - H2: HISTORIC_FUEL_DELIVERY_PRESSURE
  - H3: HISTORIC_ENGINE_COOLANT_TEMPERATURE
  - H4: HISTORIC_ENGINE_OIL_PRESSURE
  - H5: HISTORIC_ENGINE_OIL_TEMPERATURE
  - H6: HISTORIC_ENGINE_OIL_LEVEL

Table 4.2.10: Dataset Excerpt: Historic failures output format

DATE	VEHICLE ID	FEATURES	FAILURE	FAILUREDESC	H4	H5
2020-08-25	123xxx	...	1	Engine oil pressure	0	0
2020-12-30	123xxx	...	1	Engine oil level	1	0
2021-04-12	123xxx	...	0	No failure	0	1

- Total Historic Failures (THF)

  This feature keeps track of the total engine failures each vehicle has experienced throughout its life for each datetime observation. This feature was calculated by incrementing the count everytime an engine related failure was observed for a specific vehicle.

- Total Historic Unique Failures (THUF)

  This feature keeps track of the unique engine failures every vehicle has experienced throughout its life for each datetime observation. This feature was calculated by keeping track of the historic failures a vehicle had undergone in its life and if a failure occurred which was not found in the vehicle's historic data, the unique failure count was incremented by one.

- Days Since Failure (DSF)

  This feature keeps track of the number of days since the last failure occurred for that specific vehicle. This feature was calculated by keeping track of the last failure date for the specific vehicle and for every datetime observation, the datetime is subtracted from the last failure date. Once, a new failure occurs then the days since failure was set to zero and the count started again with the last failure date also being updated.

Table 4.2.11: Dataset Excerpt: More Historic failure features output format

DATE	VEHICLE ID	FEATURES	FAILURE	FAILUREDESC	THF	THUF	DSF
2020-08-25	123xxx	...	1	Engine oil pressure	0	0	0
2020-12-30	123xxx	...	1	Engine oil level	1	1	127
2021-04-12	123xxx	...	0	No failure	2	2	103

4.2.3 Prepare Final Datasets

To answer our research question, we decided to carry out three experiments by building ML models for PdM on three different datasets and then evaluate their performance. All three datasets had the non-failure and failure data respectively, however, differed in the type of data and features. Precisely, data for:

- Experiment one: Event log only
- Experiment two: Event log and component-based sensor log combined
- Experiment three: Component-based sensor log only

This dictated our approach on how we prepared our final datasets. We show the final dataset preparation as follows.

After pre-processing the aforementioned reading and faultcodes table, both of them were concatenated together to integrate the non-failure and failure data into one dataset. For each vehicle, the time series data was sorted by date in ascending order from 2015 to-date. This enabled us to have a clear picture of the life of every machine in order of the events that occurred over the course of its operations that were healthy readings and the respective engine failures.

At this point, we had all the event log data merged together including failures and non-failures for every A40G machine that had undergone an engine failure at least once.

Table 4.2.12: Dataset Excerpt: Merged dataset output format

DATE	VEHICLE ID	EVENT FEATURES	HISTORIC FEATURES	FAILURE
2020-08-25	123xxx	...	...	1
2021-04-12	123xxx	...	...	0

Then, we performed some further data cleaning and feature creation for each vehicles time series which was required after concatenating both tables.

Data Cleaning

- Missing Data and Interpolation

  After concatenating the datasets, some of the features that existed in the failure data as opposed to the non-failure data and vice versa were set to null by default. Therefore, these values needed to be filled using different interpolation techniques, some more straightforward than others.

The historic failure features for the non-failure data were null as these features were created in the faultcodes table. THF and THUF values were interpolated using straight forward backward and forward fill methods as they hold the last value of the latest failure seen for a specific vehicle. While, DSF was recalculated for the non-failure observations by subtracting the datetime of each observation from the datetime of the last failure for a specific vehicle.

The engine hours and number of parameters are discrete numerical features whose value was increasing over the life of the vehicle but were not updated by the system frequently and seemed to hold consistent values so they were also interpolated using the forward fill method.

Lastly, the number of operating hours is a continuous numerical feature whose value was also increasing over the life of the vehicle and was consistently updated by the system for every other reading. However, there were a few nulls every now and then in every other vehicle time series for this feature. Therefore, it was interpolated using a custom interpolation function which we discuss in the next subsection.

- Changing Precision

  The precision of the 'DATE' feature was upto the exact second. However, since this data was not being collected in real-time and we only wanted to predict the RUL of the equipment in terms of number of days, so we changed the precision of the 'DATE' feature to YYYY-MM-DD instead of YYYY-MM-DD HH:mm:ss. This was achieved by stripping the hour, minute and seconds component so this way we do not lose any data which you typically do if you aggregate.

- Dropping observations

  After concatenating the datasets, we found that there are some days where for a specific vehicle we find a failure and a non-failure observation. We drop the non-failure observation at this datetime and keep the failure observation to reduce the class imbalance as naturally the observations where there is no failure are more than the failure observations. In other words, we keep the minority class and drop the majority class when they exist on the same datetime for a specific vehicle.

Feature Creation

- Days Since Last Observation (DSLO)

  Since the frequency of logging is unequal for both non-failure and failure data, we introduced a feature to track the number of days passed since the last observation for each vehicles respective time series.

After these data cleaning and feature creation steps, we had completed the dataset for experiment one (event-log only). Before creating the dataset for experiment two and three, we discuss the custom interpolation function we used for the interpolation of continuous numeric data which proved to be useful for interpolating some missing sensor data as well.

4.2.4 Custom Interpolation Method

There was a need for a custom interpolation function as the traditional methods of using mean, median, mode, min, or max imputation were not sufficient for our case. Precisely, each vehicle has a time series recording the events over the life of the machine in which as time passes, continuous features such as operating hours, total fuel consumption, etc, tend to keep increasing but never decrease. Keeping this in mind, if we used any of the traditional approaches for instance mean imputation, often the mean value looked out of place for a time series observation as it might have been bigger than the next value in the time series or lower than the previous value which was not representative of the data for that specific feature in question.

In light of this, some inspiration was taken from the arithmetic and geometric progression theory (APGP) to define a custom way of determining the common difference between values in a specific window and using this to interpolate the nulls in that window. This ensured that the interpolated value was never bigger than the next value in time series and never smaller than the previous one. This helped to keep the inherent nature of the rising continuous values over time and did not add incorrect data to the feature. Note that APGP was not used itself as the data is not logged in any defined interval or consistency.

There were three cases which were encountered while interpolating and were handled by the custom function accordingly.

1. Start and end value exists

   This was the ideal case where we had a defined initial start and end for the window. We iterated over the feature values and once a null was encountered, the algorithm was aware that it needs to fill this null, so as soon as it reached the next valid value, that value was defined as the end of the window. Then, within this start and end pointer marking the window, the per day value was calculated and multiplied by the number of days passed since the start at the time the null value was encountered. This computation was then added to the start value to compute the null value. Once this was completed, the end of the window became the new start and the algorithm repeated until it found another null to fill, marked a new end pointer and calculated the interpolated value.

2. Start value is null

   Before we could interpolate the values as described in case one, we first needed to define a start value. To do this, we first defined a per day value for the entire feature values column by taking the last value and divided it by the total number of days in the time series. Then, we iterated over the feature values and the first non-null value we found, we counted the number of days passed since the start till that value and multiplied it by the per day value for the entire series and divided by ten to normalize it. Once that was done, we subtracted this from the first non-null value found. This gave us a start value for the series which was smaller than any other value encountered and was representative of the data.

3. End value is null

   The opposite of case two. Here, we found the days passed since the last valid value, multiplied it by the per day value and added the sum to the value at the last valid index.

The algorithm below shows the implementation of the custom function and how it handled these three cases.

Algorithm 1 Custom Interpolation

```
function Interpolate(df)                ▷ Where df - dataframe containing all the data
    feature_list = continuous features with null values
    for feature in feature_list do
        last_valid_index = feature.get_last_valid_index()
        total_days = ((date_at_last_valid_index − date_at_start_index)).days
        value_per_day = value_at_last_valid_index/total_days
        interpolation_map = HashMap()
        switch = False
        temp_list = []
        start_value = check_start_is_nan()
        if start_value is not False then
            feature_values[0] = start_value
            interpolation_map[0] = start_value
        end if
        for i, value in enumerate(feature_values) do
            if value_is_null is not False then
                if switch is False then
                    window_start = value
                    window_s_date = date_col[i]
                else
                    window_end = value
                    window_e_date = date_col[i]
                    window_total_value = window_end − window_start
                    window_total_days = (window_e_date − window_s_date).days
                    value_per_day = window_total_value/window_total_days
                    for i in temp_list do          ▷ DSLO - Days since last observation
                        inter_value = window_start + (DSLO[i] ∗ value_per_day)
                        interpolation_map[i] = inter_value
                        window_start = inter_value
                    end for
                    window_end = None
                    switch = False
                    temp_list = []
                    window_s_date = window_e_date
                end if
            else
                interpolation_map[i] = None
                temp_list.append(i)
                switch = True
            end if
        end for
        end_map = check_end_is_nan()
    end for
    return MERGE_HASHMAPS(interpolation_map, end_map)
end function
```

Algorithm 2 Check Start Value

```
function check_start_is_nan(df, value_per_day)
    if value_is_null(start_value) then
        for i, value in enumerate(feature_values) do
            if value_is_null is False then
                days_passed = (date_col[i] − date_col[0]).days
                return absolute(value − (days_passed ∗ value_per_day)/10)
            end if
        end for
    else
        return False
    end if
end function
```

Algorithm 3 Check End Value

```
function check_end_is_nan(df, value_per_day)
    interpolation_map = HashMap()
    if value_is_null(end_value) then
        for i = last_valid_index + 1 to length(feature_values) do
            inter_value = last_valid_value + (DSLO[i] ∗ value_per_day)
            interpolation_map[i] = inter_value
        end for
    end if
    return interpolation_map
end function
```

4.2.5 Sensor Data Transformation

Before we could create the final datasets for experiment two and three, the sensor data had to be preprocessed and transformed in a way for it to be ready to be integrated with the event log data and the failure data respectively.

As shown in subsection 4.1.2, engine-based sensor logs were retrieved. However, they needed to be transformed into the same structure as the experiment one data for them to be joined to it to create datasets for experiment two and three respectively.

Table 4.2.13: Dataset Excerpt: Sensor data before transformation output format

DATE	READING ID	VEHICLE ID	X	VALUE
2020-08-25	987xxx	123xxx	xxx	456000
2020-08-25	987xxx	123xxx	yyy	857123
2020-08-25	987xxx	123xxx	zzz	358456

In Table 4.2.13, we see that for the same 'READING ID' and 'DATE', the readings are repeating but are distinguished by the 'X' value indicating the corresponding 'Value No.' in the log. Therefore, for each reading for this log there are eleven observations and in our case that is a problem as this data cannot be joined with the rest of the prepared experiment data where a unique reading is represented as a single observation.

Therefore, we had to transform this data in a way that each 'READING ID' and the corresponding 'DATE' were represented as a single observation so that they could be joined with the rest of the data. To achieve this, we used the cross-table method which enabled us to reduce the number of observations significantly but increased the dimensionality of the data. The method achieved this by transposing the 'X' column values and created them as individual column values for each reading.

Table 4.2.14: Dataset Excerpt: Sensor data after transformation output format

DATE	READING ID	VEHICLE ID	xxx	yyy	zzz
2020-08-25	987xxx	123xxx	456000	857123	358456

All three logs were transformed using this same method as they were in the same structure at the time of retrieval. Once the structural transformation was complete, depending on the bit resolution of the log cells, the values needed to be transformed as well.

The following values were transformed depending on which log it was:

- Seconds

  All seconds were converted to hours for ease of interpretation and understanding. The cells where the resolution was one second was directly converted to hours

whereas the cells with resolution of ten seconds were first multiplied by a factor of ten and then converted to hours. Converting to hours also made it easier to understand the fuel consumption in the precision of liters per hour.

- Liter

  The original liter value in the cells was multiplied by a factor of one-tenth to convert it to per liter because of the bit resolution.

- Renamed labels

  The cell numbers were re-assigned labels by the cell description concatenated with the log number to identify which cell belongs to which log. For instance, '2' was re-labelled as 'Log_xxx_Feature_yyy'.

The data transformation was now complete for the sensor data and it was ready to be joined with the preprocessed data for experiment one which only included event log and failure data. To conclude, the final datasets were created for the following experiments:

1. Experiment one: Event only dataset

   Dataset created with failure and non-failure event log data and pre-processed as shown in subsection 4.2.3.

   Table 4.2.15: Dataset Excerpt: Experiment one - Data output format

DATE	VEHICLE ID	EVENT FEATURES	HISTORIC FEATURES	FAILURE
2020-08-25	123xxx	...	...	1
2021-04-12	123xxx	...	...	0

2. Experiment two: Event and Sensor combined dataset

   This dataset was created by joining the transformed sensor data with the experiment dataset on 'READING_ID'. The missing sensor values created after merging the two datasets were interpolated using the custom interpolation function as mentioned in subsection 4.2.4 as the sensor values for each vehicles' time series followed a continuous numeric distribution of increasing values over

time. The 'READING_ID' was dropped as the final step.

Table 4.2.16: Dataset Excerpt: Experiment two - Data output format

DATE	VEHICLE ID	EVENT FEATURES	SENSOR FEATURES	HISTORIC FEATURES	FAILURE
2020-08-25	123xxx	...	...	...	1
2021-04-12	123xxx	...	...	...	0

3. Experiment three: Sensor only dataset

   The dataset in experiment two had all the pre-processed data at our disposal merged together - event, sensor, failure and non-failure data. To create the dataset for this experiment, we dropped all the event data from the experiment two dataset to investigate the effect of only component-based sensor logs on the failure of the equipment.

Table 4.2.17: Dataset Excerpt: Experiment three - Data output format

DATE	VEHICLE ID	SENSOR FEATURES	HISTORIC FEATURES	FAILURE
2020-08-25	123xxx	...	...	1
2021-04-12	123xxx	...	...	0

Chapter 5

Applied Predictive Maintenance

The three different datasets were ready for the experiments to be carried out by building the ML models. However, these datasets could only be used to build Condition Monitoring (CM) ML models which is considered to be the most basic form of PdM. In CM, based on the input data, the ML model can inform you whether a specific vehicle has an engine failure at this current point in time instead of predicting whether it will in the future or not.

In our study, we wanted to build a PdM ML model which will predict whether a specific vehicle will undergo a failure in the future which is also known as the RUL of the vehicle. However, it was important to build a baseline ML model for CM to evaluate the performance and see how it differed from the RUL models performance. Simply because a CM model typically outperforms an RUL model as naturally it is easier to predict the current state of the machine instead of the future state. It would be interesting to see how the model performance differs for the same set of data.

The tables 4.2.15, 4.2.16, 4.2.17 were ready to be used for CM.

5.1 Remaining Useful Life

Now that the data was ready for CM, we needed to augment it slightly to prepare it for an RUL model. The RUL model built is a binary classification model which predicts:

- Will there be an engine failure in the next x days for a specific vehicle?

We created a new binary target class called FAILURE_WITHIN_NEXT_X_DAYS to achieve this. A value of one indicates that an engine failure will occur in the next x days while zero indicates otherwise. Typically, for critical failures or components, the desired period to predict ahead of time is 60-90 days. However, in our case, we decided to determine the next x days by looking at how often does an engine failure occur in our data. This was calculated by looking at the distribution of our created feature 'Days Since Failure' (DSF) for all the vehicles' time series.

Table 5.1.1: Days since failure per vehicle statistics

MEAN	**STD**	**MIN**	**25%**	**50%**	**75%**	**MAX**
140.56	168.06	0.00	18.00	78.00	207.00	1263.00

We see in Table 5.1.1 that 25% of engine failures occur within 18 days, 50% within 78 days and 75% within 207 days. Based on these quartile values for DSF, we created three target label columns.

1. 25%: FAILURE_WITHIN_NEXT_18_DAYS
2. 50%: FAILURE_WITHIN_NEXT_78_DAYS
3. 75%: FAILURE_WITHIN_NEXT_207_DAYS

Separate ML models were built with each of these individual class labels and the performance of the models was observed in relation to the different periods of time.

To create these new target classes, we make use of the two following techniques:

5.1.1 Lead Data Shift

The target variables were created using the lead variable technique. The lead variable essentially looks to the next observation and takes its value at the current observation. It works in the opposite way to a lag variable. However, in our case, we could not just not look at the next observation as the date interval for the time series is unequal. Therefore, instead, we made use of the window method to identify whether there were any failures in the next x days. This was achieved by creating a window with the start date of the current observation offset by one additional day and the end date was equivalent to the start date offset by an additional x days. If any failure observation existed within this window $W[start\_date + (1\_day) : start\_date + (x\_days)]$, then

the current observations' new target variable was set to one, otherwise zero. This process was applied to each vehicles' entire time series for every existing date to identify future failures ahead of time and was done for all three aforementioned values of x separately.

Algorithm 4 Get Lead Shifted Label

```
function get_shifted_label(df)        ▷ Where df - dataframe containing all the data
    fwn_x_days_list = []                                  ▷ FWN - FAILURE WITHIN NEXT
    vehicle_list = list of unique vehicle IDs in our data
    for vehicle_id in vehicle_list do
        vehicle_df = df[df[VEHICLE_ID] == vehicle_id]
        dates = vehicle_df[DATES]
        for date in dates do
            window = start_date + (1_day) : start_date + (x_days)
            curr_window_df = vehicle_df[window]
            if 1 in set(curr_window_df[FAILURE].values) then
                fwn_x_days_list.append(1)
            else
                fwn_x_days_list.append(0)
            end if
        end for
        vehicle_df[fwn_x_days] = fwn_x_days_list
    end for
end function
```

Let us observe the PdM dataset outputted by the algorithm that is ready to be used to build a RUL model.

Table 5.1.2: Dataset Excerpt: PdM data output format

DATE	VEHICLE ID	FEATURES	FAILURE	FWN 18 DAYS	FWN 78 DAYS	FWN 207 DAYS
2016-11-16	123xxx	...	1	1	1	1
2016-11-17	123xxx	...	1	0	0	0
2017-07-05	123xxx	...	0	0	1	1
2017-07-20	123xxx	...	0	1	1	1
2017-07-28	123xxx	...	1	0	1	1
2017-08-04	123xxx	...	0	0	1	1
2017-08-30	123xxx	...	0	1	1	1
2017-09-04	123xxx	...	1	0	0	1

The vehicle ID was one-hot encoded at this stage as the final step. The entire dataset including the target classes were now ready to be used to build individual RUL models using ML.

5.1.2 Lead Data Shift with Resampling

This is another technique to prepare the dataset for PdM that also uses the same approach as highlighted in the previous section to create the target classes. However, along with that, this technique fixes the unequal date interval. If you notice in Table 5.1.3, there are missing date observations, ideally you should have an observation logged every single day for daily data. However, that was not the case with our real-world data, so this is where we made use of resampling to introduce those observations to achieve an equal date interval for every vehicles' time series. Even though you lose data by using the resampling technique due to aggregation, it was acceptable to an extent in our case as there was a imbalance of observations between non-failure and failure. Therefore, losing some non-failure observations may not be a problem, however, losing too many failure observations could be.

This technique was applied by creating the lead data shift variable the same way as described in the previous subsection 5.1.1, but by making sure the date frequency differs by every x days. Observations which lie in the window between the start date and end date after every x days were used to extract the target variable and latest feature values to be assigned to the new equal interval observation and then the existing observations in the current window were dropped.

In other words, essentially we took the closest reading to the end-date for every window to resample and assigned the target to be one/zero depending on whether a failure existed in the window. This worked well as the data was a continuous time series for every vehicle with values increasing over time and as we ensured that we kept the latest observation to every resampled date.

Algorithm 5 Prepare resampled data

function prepare_resampled_df(df) ▷ Where df - dataframe containing the data
    $vehicle\_list$ = list of unique vehicle IDs in our data
    for $vehicle\_id$ in $vehicle\_list$ **do**
        $vehicle\_df = df[df[VEHICLE\_ID] == vehicle\_id]$
        $start\_date = df[DATE].values[0]$
        $end\_date = df[DATE].values[-1]$
        $dates = date\_range(start\_date, end\_date, freq = x\_Days)$
        $empty\_df = DataFrame(cols = df.columns, index = dates)$
        $resampled\_df = concat([df, empty\_df]).sort\_values(DATE)$
    end for
end function

Once we had applied Algorithm 5, we got the resampled dataframes for every vehicle, then we just applied the windowing method as shown in Algorithm 4 to get the target classes. For all three different values of x, we created three different resampled dataframes for every vehicle.

Let us observe the PdM dataset with resampling outputted by the algorithm that is ready to be used to build an RUL model.

Table 5.1.3: Dataset Excerpt: PdM data with resampling output format

DATE	VEHICLE ID	FEATURES	FAILURE	FWN 78 DAYS
2016-11-16	123xxx	...	1	1
2017-02-02	123xxx	...	1	0
2017-04-21	123xxx	...	0	0
2017-07-08	123xxx	...	0	1
2017-09-24	123xxx	...	1	1
2017-12-11	123xxx	...	1	1

Now we see that the date intervals have become equal. In this case, where $x = 78$, every observation differs by 78 days. The vehicle ID was also one-hot encoded at this stage as the final step. The entire dataset including the target classes were ready to be used to build individual RUL models using ML for resampled data.

Chapter 6

Model, Results, and Analysis

All the datasets have been prepared and the experiments have been defined by this stage as shown in Chapter 4 and 5. In this chapter, we describe the models built using these datasets and the experiments carried out on them along with the results of these different experiments and their analysis. The experiments are categorized as follows:

- Experiment one: Event data only
- Experiment two: Event and Sensor data combined
- Experiment three: Sensor data only

These experiments were carried out for both CM and RUL ML models. The main focus was on the RUL models but we established some baseline comparison results by carrying out the experiments for CM as well. For each experiment, the data was split in a stratified way into 70% training and 30% test sets respectively. This was executed three separate times where the split was done once on the original data, once on the up-sampled data where the number of instances of the minority class were made equivalent to the majority class and lastly one split for the down-sampled data where the number of instances for the majority class were made equivalent to the minority class. In all three cases, as it turned out, the majority class was non-failure and the minority class was failure.

Furthermore, the key metrics to measure the performance of both models were according to those mentioned in subsection 3.2.6, however, the main ones we looked at to distinguish model superiority were F-Score and Area under Precision-Recall curve

(AU-PRC). Both of these metrics were chosen as they have proven to be successful metrics which represent the true picture of a model when the data is imbalanced. Their scores indicate whether the model is predicting a high or low number of false positives and false negatives which helps to identify real threats and avoid false alarms. Moreover, their values can indicate how well the minority class (failure) is actually being predicted.

The highlighted rows in every results table indicates the top two models for that experiment.

6.1 Condition Monitoring

The 'FAILURE' column is the binary class label.

Table 6.1.1: Number of instances in the data

DATA	FAILURE = 0	FAILURE = 1
Original	7759	2850
Training	5431	1995
Test	2328	855
Upsampled Training	5431	5431
Downsampled Training	1995	1995

Table 6.1.2: Experiment one: Condition monitoring results

Algorithm	SAMPLING	ACCURACY	AUC	PRECISION	RECALL	F-SCORE	AU-PRC
LR	None	73.7	0.62	0.71	0.52	0.46	0.40
LR	Up	54.19	0.67	0.60	0.62	0.53	0.41
LR	Down	60.04	0.67	0.60	0.62	0.58	0.41
RF	None	79.61	0.84	0.75	0.69	0.71	0.66
RF	Up	79.89	0.84	0.75	0.72	0.73	0.65
RF	Down	75.05	0.83	0.71	0.76	0.72	0.64
XgBoost	None	80.21	0.86	0.75	0.75	0.75	0.69
XgBoost	Up	80.08	0.86	0.75	0.76	0.75	0.69
XgBoost	Down	73.8	0.85	0.71	0.76	0.71	0.66

Table 6.1.3: Experiment two: Condition monitoring results

Algorithm	SAMPLING	ACCURACY	AUC	PRECISION	RECALL	F-SCORE	AU-PRC
LR	None	73.17	0.53	0.67	0.5	0.43	0.28
LR	Up	69.62	0.52	0.48	0.49	0.45	0.27
LR	Down	70.25	0.54	0.50	0.50	0.46	0.28
RF	None	87.06	0.95	0.89	0.78	0.81	0.87
RF	Up	89.00	0.94	0.89	0.82	0.85	0.87
RF	Down	88.72	0.94	0.85	0.87	0.86	0.87
XgBoost	None	93.47	0.98	0.92	0.91	0.92	0.95
XgBoost	Up	93.43	0.98	0.92	0.91	0.92	0.95
XgBoost	Down	90.39	0.97	0.87	0.91	0.88	0.94

Table 6.1.4: Experiment three: Condition monitoring results

Algorithm	SAMPLING	ACCURACY	AUC	PRECISION	RECALL	F-SCORE	AU-PRC
LR	None	73.23	0.53	0.87	0.5	0.43	0.27
LR	Up	70.19	0.52	0.49	0.50	0.46	0.27
LR	Down	70.75	0.54	0.52	0.50	0.47	0.28
RF	None	86.93	0.95	0.89	0.77	0.81	0.87
RF	Up	88.75	0.94	0.89	0.82	0.84	0.87
RF	Down	88.63	0.94	0.85	0.87	0.86	0.87
XgBoost	None	93.53	0.97	0.93	0.91	0.92	0.95
XgBoost	Up	93.18	0.97	0.92	0.91	0.91	0.95
XgBoost	Down	89.04	0.96	0.85	0.89	0.87	0.93

After carrying out the three experiments for CM, we saw that the models in Table 6.1.2 built only using event data did not perform nearly well enough as those models in experiment two and three. The models for experiment two and three as shown in Table 6.1.3 seemed to perform extremely well and had very similar consistent results. For both of them, XgBoost was the highest performing model with an F-score of 0.91-0.92 and AU-PRC of 0.95 for either the original or upsampled data. The difference in performance was nearly negligible. After achieving these results, we became quite confident that we can perform condition monitoring using our data at a very high level with very few false positives and false negatives.

Moreover, all the three algorithms performed on a similar comparative level across experiments from which we derived that ensemble and boosting models are outperforming LR which as expected. However, due to the big difference in performance, we decided to drop the LR model for the RUL models as even after tuning it would not reach the level of performance of the ensemble or boosted algorithms. The CM results gave us a good grounding to proceed with building the RUL models with an inclination towards the experiment two and three datasets.

6.2 Remaining Useful Life

For both of the approaches mentioned in subsection 5.1.1 and 5.1.2, the three aforementioned experiments were carried out respectively on the three datasets pre-processed as shown in Chapter 4 and 5.

The class imbalance was further dependent upon the chosen approach and the value of x in the class label, where x can be 18, 78 or 207 days. The class imbalance decreases as the value of x increases as more failures are included within the x days window. In other words, as the window size increases, so does the likelihood of failure.

6.2.1 Lead Data Shift Approach

Table 6.2.1: Number of class instances in the data per label

DATA	**FWN_18 = 0**	**FWN_78 = 0**	**FWN_207 = 0**	**FWN_18 = 1**	**FWN_78 = 1**	**FWN_207 = 1**
Original	9433	7457	5389	1176	3152	5220
Training	6603	5220	3772	823	2206	3654
Test	2830	2237	1617	353	946	1566
Upsampled Training	6603	5220	3772	6603	5220	3772
Downsampled Training	823	2206	3654	823	2206	3654

In Table 6.2.1, we see that as the window size increases, the number of non-failures decrease and the number of failures increase across all sets of data.

Experiment one

Table 6.2.2: Experiment one: PdM for FWN 18 days using lead data shift label results

Algorithm	SAMPLING	ACCURACY	AUC	PRECISION	RECALL	F-SCORE	AU-PRC
RF	None	88.31	0.74	0.65	0.54	0.56	0.26
RF	Up	87.02	0.73	0.62	0.57	0.58	0.27
RF	Down	68.90	0.73	0.57	0.67	0.56	0.24
XgBoost	None	84.04	0.75	0.61	0.63	0.62	0.30
XgBoost	Up	79.36	0.75	0.60	0.66	0.61	0.29
XgBoost	Down	55.86	0.72	0.56	0.65	0.48	0.22

Table 6.2.3: Experiment one: PdM for FWN 78 days using lead data shift label results

Algorithm	SAMPLING	ACCURACY	AUC	PRECISION	RECALL	F-SCORE	AU-PRC
RF	None	78.23	0.83	0.76	0.68	0.70	0.69
RF	Up	78.83	0.83	0.76	0.71	0.72	0.69
RF	Down	74.27	0.82	0.71	0.74	0.71	0.65
XgBoost	None	76.03	0.83	0.72	0.73	0.72	0.65
XgBoost	Up	74.96	0.83	0.71	0.74	0.72	0.66
XgBoost	Down	68.77	0.81	0.69	0.73	0.67	0.61

Table 6.2.4: Experiment one: PdM for FWN 207 days using lead data shift label results

Algorithm	SAMPLING	ACCURACY	AUC	PRECISION	RECALL	F-SCORE	AU-PRC
RF	None	79.30	0.88	0.79	0.79	0.79	0.87
RF	Up	78.45	0.88	0.78	0.78	0.78	0.87
RF	Down	79.20	0.88	0.79	0.79	0.79	0.87
XgBoost	None	78.32	0.87	0.78	0.78	0.78	0.86
XgBoost	Up	78.04	0.87	0.78	0.78	0.78	0.86
XgBoost	Down	78.10	0.87	0.78	0.78	0.78	0.86

Experiment two

Table 6.2.5: Experiment two: PdM for FWN 18 days using lead data shift label results

Algorithm	SAMPLING	ACCURACY	AUC	PRECISION	RECALL	F-SCORE	AU-PRC
RF	None	88.82	0.71	0.68	0.53	0.53	0.26
RF	Up	85.67	0.66	0.57	0.54	0.55	0.20
RF	Down	66.26	0.70	0.56	0.65	0.54	0.22
XgBoost	None	87.56	0.74	0.65	0.59	0.60	0.30
XgBoost	Up	84.42	0.72	0.60	0.60	0.60	0.27
XgBoost	Down	54.45	0.71	0.56	0.65	0.47	0.23

Table 6.2.6: Experiment two: PdM for FWN 78 days using lead data shift label results

Algorithm	SAMPLING	ACCURACY	AUC	PRECISION	RECALL	F-SCORE	AU-PRC
RF	None	76.85	0.78	0.75	0.65	0.66	0.61
RF	Up	72.89	0.75	0.67	0.64	0.64	0.54
RF	Down	70.28	0.77	0.67	0.70	0.67	0.59
XgBoost	None	76.12	0.80	0.71	0.70	0.71	0.63
XgBoost	Up	74.49	0.80	0.70	0.70	0.70	0.62
XgBoost	Down	68.49	0.79	0.68	0.71	0.67	0.60

Table 6.2.7: Experiment two: PdM for FWN 207 days using lead data shift label results

Algorithm	SAMPLING	ACCURACY	AUC	PRECISION	RECALL	F-SCORE	AU-PRC
RF	None	73.92	0.82	0.74	0.74	0.74	0.82
RF	Up	74.11	0.83	0.74	0.74	0.74	0.82
RF	Down	73.74	0.82	0.74	0.74	0.74	0.82
XgBoost	None	76.97	0.86	0.77	0.77	0.77	0.85
XgBoost	Up	76.75	0.86	0.77	0.77	0.77	0.85
XgBoost	Down	76.37	0.86	0.76	0.76	0.76	0.86

Experiment three

Table 6.2.8: Experiment three: PdM for FWN 18 days using lead data shift label results

Algorithm	SAMPLING	ACCURACY	AUC	PRECISION	RECALL	F-SCORE	AU-PRC
RF	None	88.88	0.69	0.69	0.53	0.53	0.24
RF	Up	85.67	0.63	0.57	0.54	0.55	0.19
RF	Down	64.75	0.67	0.55	0.62	0.52	0.22
XgBoost	None	87.15	0.68	0.62	0.56	0.58	0.23
XgBoost	Up	82.53	0.67	0.58	0.59	0.59	0.22
XgBoost	Down	48.92	0.65	0.55	0.61	0.43	0.19

Table 6.2.9: Experiment three: PdM for FWN 78 days using lead data shift label results

Algorithm	SAMPLING	ACCURACY	AUC	PRECISION	RECALL	F-SCORE	AU-PRC
RF	None	75.62	0.75	0.73	0.63	0.64	0.58
RF	Up	72.35	0.72	0.66	0.62	0.63	0.51
RF	Down	69.15	0.74	0.66	0.68	0.66	0.55
XgBoost	None	72.98	0.75	0.67	0.66	0.67	0.58
XgBoost	Up	71.94	0.75	0.67	0.67	0.67	0.57
XgBoost	Down	62.80	0.74	0.64	0.67	0.62	0.54

Table 6.2.10: Experiment three: PdM for FWN 207 days using lead data shift label results

Algorithm	SAMPLING	ACCURACY	AUC	PRECISION	RECALL	F-SCORE	AU-PRC
RF	None	72.35	0.81	0.72	0.72	0.72	0.80
RF	Up	72.23	0.81	0.72	0.72	0.72	0.80
RF	Down	72.95	0.80	0.73	0.73	0.73	0.80
XgBoost	None	74.14	0.83	0.74	0.74	0.74	0.83
XgBoost	Up	73.64	0.83	0.74	0.74	0.74	0.82
XgBoost	Down	73.92	0.82	0.74	0.74	0.74	0.82

After performing all three experiments with three different failure ranges, we found that experiment one with only the event data outperformed experiment two and three with an F-score of 0.79 for failure within 207 days by the RF model. This is the highest score achieved across all the experiments.

The results for the 18 day window were not good enough with the F-score ranging between 0.56 - 0.62 across all experiments indicating a very high number of false positives and negatives. This shows that predicting failure in an approximately three

weeks window is extremely difficult, especially since our research is looking at only engine failures that are considered to be critical failures.

Nonetheless, 78 day models perform decent across all experiments with an F-score of 0.64 - 0.71. However, if this model could be improved to between a score between 0.75 - 0.80, then it can be considered as an extremely good model as predicting an engine failure approximately less than three months would be desired.

The results do show that a larger window improves the results significantly. One of the main reasons is that the class imbalance decreases naturally as the window size increases. Therefore, there are more failure observations for the model to build on as shown in Table 6.2.1. However, a 207 day window can be considered to be quite large. Therefore, possibly finding a trade-off between performance of model and failure window would be the key to settling on a well balanced model.

6.2.2 Resampled Data Approach

Table 6.2.11: Number of class instances in the data per label

DATA	**FWN_18 = 0**	**FWN_78 = 0**	**FWN_207 = 0**	**FWN_18 = 1**	**FWN_78 = 1**	**FWN_207 = 1**
Original	32,426	6606	2036	1997	1720	1412
Training	22,698	4624	1425	1398	1204	988
Test	9728	1982	611	599	516	424
Upsampled Training	22698	4624	1425	22698	4624	1425
Downsampled Training	1398	1204	988	1398	1204	988

In Table 6.2.11, we see that the non-failure class has increased dramatically for the 18 day period. This is because to ensure equal intervals, the data introduces missing samples for every 18 days which are non-failure observations. This increases the class imbalance significantly. However, the ratio seems to be much better for the 78 and 207 day periods respectively.

Experiment one

Table 6.2.12: Experiment one: PdM for FWN 18 days using resample approach results

Algorithm	SAMPLING	ACCURACY	AUC	PRECISION	RECALL	F-SCORE	AU-PRC
RF	None	93.47	0.69	0.50	0.50	0.49	0.10
RF	Up	92.47	0.69	0.50	0.50	0.50	0.10
RF	Down	65.87	0.67	0.53	0.62	0.47	0.10
XgBoost	None	81.72	0.67	0.54	0.61	0.54	0.12
XgBoost	Up	66.86	0.66	0.53	0.62	0.48	0.11
XgBoost	Down	39.96	0.67	0.53	0.61	0.34	0.11

Table 6.2.13: Experiment one: PdM for FWN 78 days using resample approach results

Algorithm	SAMPLING	ACCURACY	AUC	PRECISION	RECALL	F-SCORE	AU-PRC
RF	None	76.70	0.67	0.50	0.50	0.47	0.30
RF	Up	75.74	0.67	0.55	0.52	0.51	0.31
RF	Down	66.25	0.68	0.59	0.62	0.58	0.33
XgBoost	None	73.42	0.68	0.60	0.61	0.61	0.34
XgBoost	Up	68.98	0.68	0.59	0.62	0.60	0.35
XgBoost	Down	55.88	0.66	0.57	0.61	0.52	0.32

Table 6.2.14: Experiment one: PdM for FWN 207 days using resample approach results

Algorithm	SAMPLING	ACCURACY	AUC	PRECISION	RECALL	F-SCORE	AU-PRC
RF	None	62.22	0.67	0.61	0.57	0.55	0.56
RF	Up	63.29	0.66	0.62	0.59	0.58	0.55
RF	Down	64.06	0.68	0.63	0.63	0.63	0.56
XgBoost	None	61.84	0.67	0.60	0.60	0.60	0.56
XgBoost	Up	62.71	0.67	0.61	0.61	0.61	0.55
XgBoost	Down	59.71	0.65	0.59	0.60	0.59	0.54

Experiment two

Table 6.2.15: Experiment two: PdM for FWN 18 days using resample approach results

Algorithm	SAMPLING	ACCURACY	AUC	PRECISION	RECALL	F-SCORE	AU-PRC
RF	None	93.91	0.68	0.52	0.50	0.49	0.10
RF	Up	92.01	0.67	0.53	0.51	0.51	0.09
RF	Down	60.09	0.63	0.52	0.59	0.44	0.08
XgBoost	None	90.08	0.66	0.54	0.54	0.54	0.10
XgBoost	Up	80.31	0.65	0.53	0.59	0.53	0.10
XgBoost	Down	48.06	0.65	0.52	0.60	0.38	0.10

Table 6.2.16: Experiment two: PdM for FWN 78 days using resample approach results

Algorithm	SAMPLING	ACCURACY	AUC	PRECISION	RECALL	F-SCORE	AU-PRC
RF	None	78.26	0.67	0.55	0.51	0.47	0.31
RF	Up	75.54	0.64	0.57	0.54	0.54	0.29
RF	Down	62.73	0.64	0.58	0.62	0.56	0.29
XgBoost	None	75.58	0.66	0.59	0.56	0.57	0.33
XgBoost	Up	73.18	0.66	0.59	0.59	0.59	0.33
XgBoost	Down	58.05	0.66	0.58	0.62	0.54	0.32

Table 6.2.17: Experiment two: PdM for FWN 207 days using resample approach results

Algorithm	SAMPLING	ACCURACY	AUC	PRECISION	RECALL	F-SCORE	AU-PRC
RF	None	61.84	0.64	0.60	0.58	0.57	0.54
RF	Up	60.68	0.64	0.59	0.58	0.58	0.53
RF	Down	60.77	0.64	0.60	0.60	0.60	0.53
XgBoost	None	63.38	0.67	0.62	0.61	0.61	0.57
XgBoost	Up	61.06	0.66	0.59	0.59	0.59	0.56
XgBoost	Down	62.51	0.67	0.62	0.62	0.62	0.55

Experiment three

Table 6.2.18: Experiment three: PdM for FWN 18 days using resample approach results

Algorithm	SAMPLING	ACCURACY	AUC	PRECISION	RECALL	F-SCORE	AU-PRC
RF	None	93.89	0.66	0.50	0.50	0.49	0.09
RF	Up	91.73	0.64	0.52	0.51	0.51	0.09
RF	Down	59.46	0.61	0.52	0.59	0.44	0.08
XgBoost	None	90.06	0.60	0.53	0.52	0.52	0.08
XgBoost	Up	72.89	0.60	0.52	0.57	0.49	0.08
XgBoost	Down	42.6	0.60	0.52	0.57	0.35	0.08

Table 6.2.19: Experiment three: PdM for FWN 78 days using resample approach results

Algorithm	SAMPLING	ACCURACY	AUC	PRECISION	RECALL	F-SCORE	AU-PRC
RF	None	78.02	0.64	0.54	0.51	0.47	0.29
RF	Up	75.02	0.61	0.56	0.53	0.53	0.26
RF	Down	59.73	0.63	0.56	0.59	0.54	0.28
XgBoost	None	74.22	0.62	0.55	0.55	0.54	0.28
XgBoost	Up	69.26	0.62	0.55	0.56	0.55	0.29
XgBoost	Down	53.04	0.62	0.55	0.58	0.50	0.28

Table 6.2.20: Experiment three: PdM for FWN 207 days using resample approach results

Algorithm	SAMPLING	ACCURACY	AUC	PRECISION	RECALL	F-SCORE	AU-PRC
RF	None	61.35	0.62	0.59	0.57	0.56	0.51
RF	Up	61.16	0.62	0.59	0.58	0.58	0.50
RF	Down	60.19	0.61	0.59	0.60	0.59	0.50
XgBoost	None	61.26	0.64	0.59	0.59	0.59	0.53
XgBoost	Up	59.61	0.64	0.58	0.58	0.58	0.52
XgBoost	Down	61.06	0.64	0.61	0.61	0.60	0.51

After performing all three experiments with three different failure ranges for the resampled equal interval data, we found that overall the results were not upto the required standard. The highest F-score achieved was 0.61-0.62, for both the 78 and 207 days across two experiments. The resampling technique results in the loss of data due to aggregation, which is one of the reasons for the poor performance along with the over-arching increased class imbalance that id not allow the model to perform well. The loss of data can be acceptable at times when there is sufficient data available at frequent intervals or even if most of the data lost was non-failures. However, in this case failure data was also significantly lost, which did not help the models at all.

6.2.3 Summary

The results shown in Table 6.2.1 and 6.2.2 can be summarized by taking the top performing model from each experiment for each approach over the different failure date ranges.

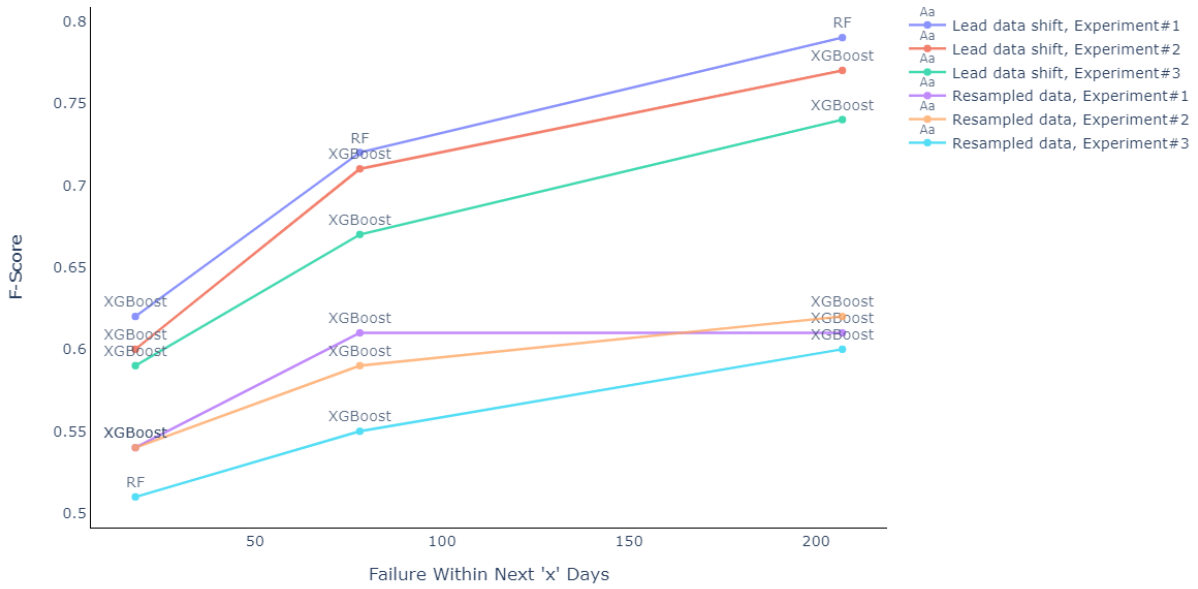

Figure 6.2.1: Model F-Score Performance Summary across all experiments

The top three performing models are from the lead data shift approach and are for the 207 time period. The general trend can be seen that as the window size increases the F-Score of the model increases. It is able to predict failures better with a lower number of false positives and negativves.

Surprisingly, Experiment one with the event only data came out on top as the top performing model with the hybrid approach in second and only sensor data in third.

This is possibly due to the event data being less complex and being reliant on the historical features with lower dimensionality and variance. While, the datasets with sensor data are more complex with higher dimensionality and variance. However, the performance was the complete opposite when it came to CM with experiment two and three outperforming experiment one by a big margin.

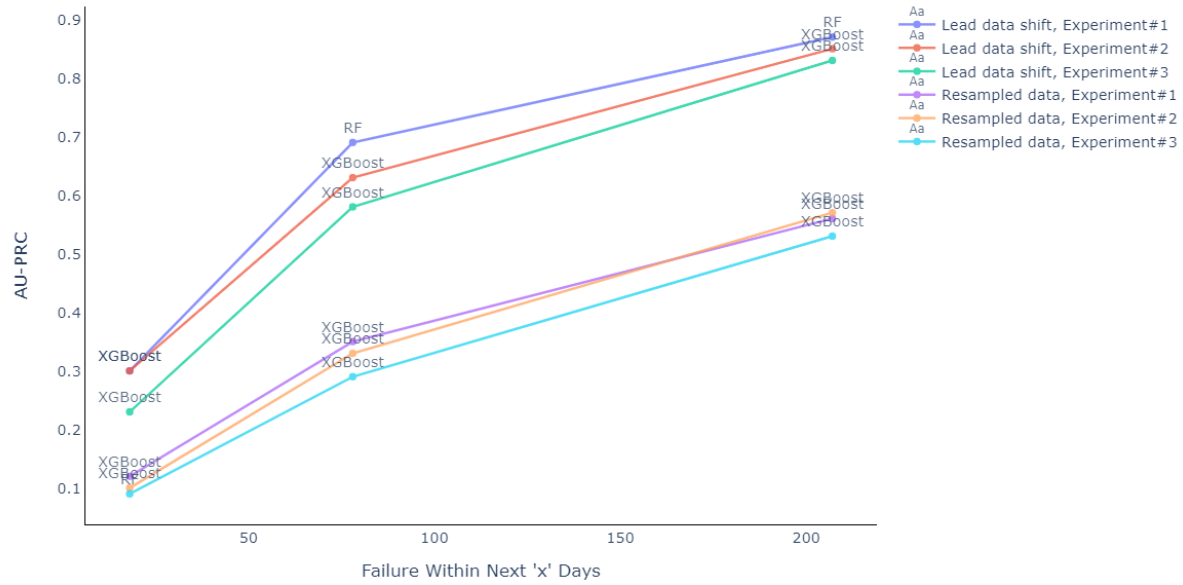

Figure 6.2.2: Model AU-PRC Performance Summary across all experiments

The AU-PRC model performance across all the experiments backed up the results of the F-Score summary as shown in Figure 6.2.1. The same approach and experiments were the top performers with the highest AU-PRC score of 0.87 indicating that the classifier has sufficient enough skill to distinguish between the classes with the score taking into account the effect of the minority class as well. In comparison to the commonly used AU-ROC (AUC), AU-PRC takes the impact of the minority class into account while determining the skill and ability of the classifier to distinguish between classes, whereas AUC does not cater for the minority class and it's score can be misleading when the data is imbalanced.

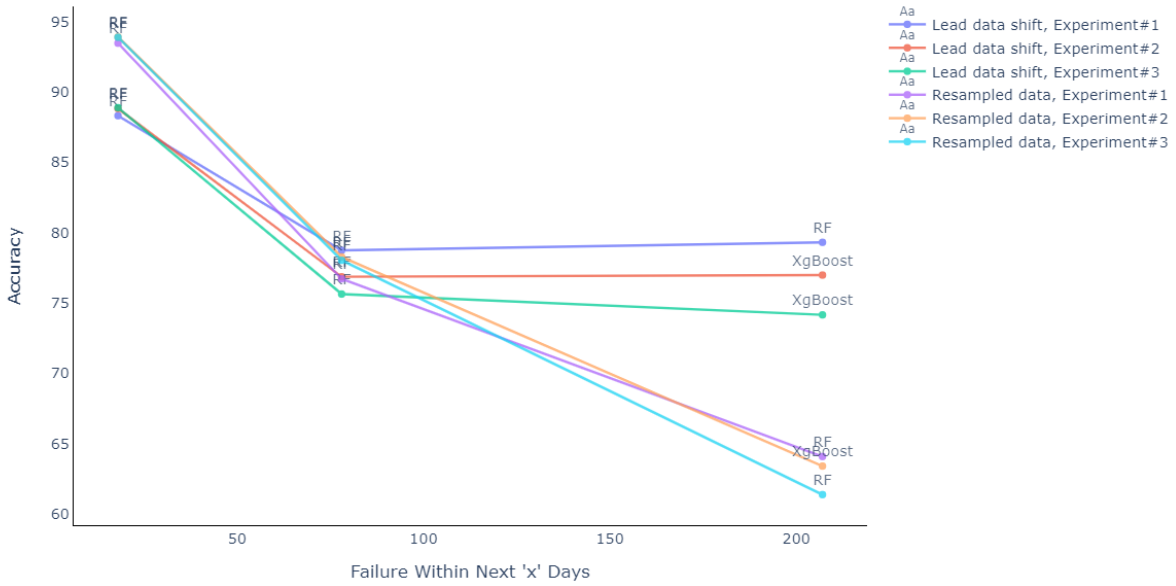

Figure 6.2.3: Model Accuracy Performance Summary across all experiments

We already know that accuracy is not a good metric for imbalanced data. However, it is important to show how its performance is the total opposite of what is indicated by the F-score. This is precisely because the high accuracy scores of up to 0.95 are due to large imbalance of data because of resampling with the majority class to minority class ratio rising up to 33:1. As the window size increases, the class imbalance decreases with more failure observations becoming a part of the window and therefore, the accuracy decreases as the models bias decreases towards predicting the majority non-failure class.

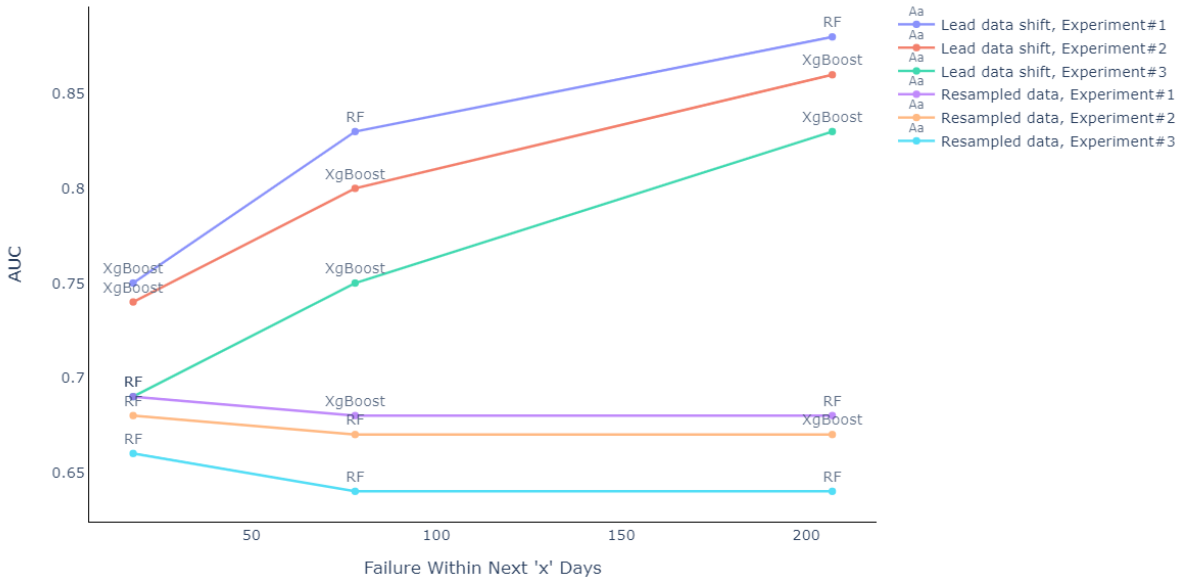

Figure 6.2.4: Model AUC Performance Summary across all experiments

The AUC performance follows a similar trend to the PR-AUC model performance summary as shown in Figure 6.2.2 for the lead data shift approach. However, for the resampling approach, the AUC score is nearly the same with a slight decrease with an increasing window size, indicating that the AUC is not reflective of characterizing the performance of the model when it comes to classifying the minority class. Therefore, even with an increasing window size resulting in a decreasing class imbalance, the AUC score does not take into account the effect of the minority failure class well.

Chapter 7

Conclusion

In this research work, we deeply explored the area of PdM using log data. We presented ways on how to prepare both event and component based sensor data using data pre-processing and feature engineering techniques to ensure that it is ready for both CM and PdM. Using this prepared data, we used two different approaches to create the target look-ahead class and carried out three different experiments respectively with different combinations of data imbalance and window size of the look-ahead time for future prediction.

These experiments gave us an insight on how model performance is affected based on using different approaches, different data, window sizes and most importantly, which of these settings is optimal for PdM for construction equipment based on the real-world log data.

7.1 Discussion

There were many key-takeaways derived from this research work both on the data and model aspect respectively. These takeaways supported the recent topic of discussion in the ML community of whether the focus should be on a data-centric or model-centric approach to improve the performance of ML models.

On the data front, too many infrequent readings per vehicle was a hindrance to our models with only a median of six readings per vehicle. Not only that, but even these readings which existed had an extremely unequal time interval between them that resulted in highly skewed time intervals for nearly every vehicle. This clearly showed

the need for real-time logging of data at regular intervals, any specified interval as long as it is regular for e.g. daily. Without these regular intervals, the chance of knowing what the condition of the component of a specific vehicle was using the event or component-sensor data is only as good as your luck of how recent the latest reading is. If you are lucky, it is very recent within a few days of the failure, or it could be months old, which may not be as useful. Having said that, we achieved decent performance with F-scores ranging between 0.74 - 0.79. With regular interval data, this score can improve further leading to good enough models with very few false positive and false negative predictions that can be deployed. Therefore, the ”garbage in, garbage out” argument holds and backs up the argument of focusing more on a data-centric approach. Lastly, we found that log data is not being collected for predictive modelling purposes but rather more from a reporting standpoint. This gives us insight that with the rise of Industry 4.0, there is a need for new data design and systems that specifically cater for predictive modelling techniques. Organizations across the board will need to invest in such systems if they are to succeed in transitioning into making decisions supported by their products' data.

Even though our research work was largely focused on a data-centric approach, there were still key take-aways from a model perspective as well. The experiments showed that PdM model performance tends to increase as the window size of the look-ahead period increases. However, the smaller the look-ahead period for PdM the better, as if the look-ahead period is too large, the machine is bound to fail within that time-window. With the large look-ahead period, the prediction may not be of any use to the organization, therefore, there exists a trade-off between model performance and the look-ahead period which needs to be decided upon on a case-by-case basis dependent on which component or equipment under observation. Such decision making, requires input from domain experts that handle and maintain the equipment. Moreover, the lead-data shift approach outperforms the resampling approach quite comfortably. This is down to the fact that resampling loses key failure data due to aggregation and introduces more non-failure observation leading to an even higher class imbalance. This adversely effects the model performance and makes this approach not fit for purpose. Having said that, resampling can still be a good approach if there is sufficient data available where losing a bit of data may not be a problem. However, in our case, we could not afford to lose any failure data as we already had an existing class imbalance that was being accounted for by upsampling or downsampling in the experiments.

Lastly, CM is easy to do, however, PdM is not. With the same combination of approach, experiments and data, the highest F-Score achieved for CM was 0.91-0.92 whereas the highest of score for PdM was 0.77-0.79 for a 78/207 day period respectively.

7.2 Future Work

This research area can be explored further and our work can act as a foundation to build upon. A second step can be added to our PdM model by introducing a multi-class classification ML model built only using the component failure data. Once our initial model predicts failure in the next x days, the second model comes into action to predict what this exact failure would be with a certain probability. Moreover, additional data and features can be introduced into the model such as service and warranty claim data. This data source was not used in this research work as there was a doubt over its quality as customers can sometimes get their vehicles serviced or repaired by third party service providers which cannot be accounted for. Introducing this data could have made the historical life of the vehicles' time series unreliable, however, it is definitely something worth exploring. More vehicle models can be added to the data as long as they tend to follow the same type of data trend which can be determined by what tasks the machine performs on a day-to-day basis. For instance, if all different models of articulated haulers e.g. A25G, A30G, A40G perform similar work, then their data could be merged to see how performance is affected, or separate models could be built for each vehicle model and the performance could be compared. Along with this integration of data from different vehicle models, the one-hot encoding technique can be replaced with feature hashing for the vehicle IDs which will make the model scalable to the addition of new vehicles as well. Also, these solutions can be scaled to other components of the vehicle apart from engines, other components such as transmission, hydraulics, control systems, etc. Moreover, a prediction interval with a certain probability could be introduced along with the prediction made by the model to support the quality of the prediction. Lastly, instead of just running experiments for three different values of the look-ahead window size x, it can be run for many more values or a range of values using a bucketing approach to find the right trade-off / balance between model performance and the need for the right look-ahead period for the specific component in question.

References

[1] Afrinaldi, Feri, Taufik, Tasman, Andrea Marta, Zhang, Hong Chao, and Hasan, Alizar. “Minimizing economic and environmental impacts through an optimal preventive replacement schedule: Model and application”. English. In: *Journal of Cleaner Production* 143 (2017), pp. 882–893. issn: 0959-6526. doi: 10.1016/j.jclepro.2016.12.033.

[2] Ayvaz, Serkan and Alpay, Koray. “Predictive maintenance system for production lines in manufacturing: A machine learning approach using IoT data in real-time”. In: *Expert Systems with Applications* 173 (2021), p. 114598. issn: 0957-4174. doi: https://doi.org/10.1016/j.eswa.2021.114598. url: https://www.sciencedirect.com/science/article/pii/S0957417421000397.

[3] Beyan, Cigdem and Fisher, Robert. “Classifying imbalanced data sets using similarity based hierarchical decomposition”. In: *Pattern Recognition* 48.5 (2015), pp. 1653–1672. issn: 0031-3203. doi: https://doi.org/10.1016/j.patcog.2014.10.032. url: https://www.sciencedirect.com/science/article/pii/S003132031400449X.

[4] Bonnevay, Stéphane, Cugliari, Jairo, and Granger, Victoria. “Predictive Maintenance from Event Logs Using Wavelet-Based Features: An Industrial Application”. In: 2020, pp. 132–141. isbn: 978-3-658-07615-3. doi: 10.1007/978-3-030-20055-8_13.

[5] Branco, Paula, Torgo, Luís, and Ribeiro, Rita. *A Survey of Predictive Modelling under Imbalanced Distributions*. 2015.

[6] Butler, Karen L. “An expert system based framework for an incipient failure detection and predictive maintenance system”. In: *Proceedings of International Conference on Intelligent System Application to Power Systems*. IEEE. 1996, pp. 321–326.

[7] Chen, Tianqi and Guestrin, Carlos. “Xgboost: A scalable tree boosting system”. In: *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. 2016, pp. 785–794.

[8] Choudhary, Alok Kumar, Harding, Jenny A, and Tiwari, Manoj Kumar. “Data mining in manufacturing: a review based on the kind of knowledge”. In: *Journal of Intelligent Manufacturing* 20.5 (2009), pp. 501–521.

[9] Dhall, Rohit and Solanki, Vijender. “An IoT Based Predictive Connected Car Maintenance Approach”. In: *International Journal of Interactive Multimedia and Artificial Inteligence* 4 (2017), pp. 16–22. doi: 10.9781/ijimai.2017.433.

[10] Dong, Guozhu and Pei, Jian. *Sequence data mining*. Vol. 33. Springer Science & Business Media, 2007.

[11] Edward, Kozłowski, Mazurkiewicz, Dariusz, Żabiński, Tomasz, Prucnal, Sławomir, and Se˛p, Jarosław. “Machining sensor data management for operation-level predictive model”. In: *Expert Systems with Applications* 159 (2020), p. 113600. doi: 10.1016/j.eswa.2020.113600.

[12] Erfani, Sarah M., Rajasegarar, Sutharshan, Karunasekera, Shanika, and Leckie, Christopher. “High-Dimensional and Large-Scale Anomaly Detection Using a Linear One-Class SVM with Deep Learning”. In: *Pattern Recogn.* 58.C (2016), pp. 121–134. issn: 0031-3203. doi: 10.1016/j.patcog.2016.03.028. url: https://doi.org/10.1016/j.patcog.2016.03.028.

[13] Erfani, Sarah M., Rajasegarar, Sutharshan, Karunasekera, Shanika, and Leckie, Christopher. “High-dimensional and large-scale anomaly detection using a linear one-class SVM with deep learning”. In: *Pattern Recognition* 58 (2016), pp. 121–134. issn: 0031-3203. doi: https://doi.org/10.1016/j.patcog.2016.03.028. url: https://www.sciencedirect.com/science/article/pii/S0031320316300267.

[14] Fan, H. and Jin, Z. “A Study on the Factors Affecting the Economical Life of Heavy Construction Equipment”. In: 2011.

[15] Franciosi, Chiara, Iung, Benoit, Miranda, Salvatore, and Riemma, Stefano. “Maintenance for Sustainability in the Industry 4.0 context: a Scoping Literature Review”. In: *IFAC-PapersOnLine* 51.11 (2018). 16th IFAC Symposium on Information Control Problems in Manufacturing INCOM 2018, pp. 903–908. issn: 2405-8963. doi: https://doi.org/10.1016/j.ifacol.

2018.08.459. url: https://www.sciencedirect.com/science/article/pii/S2405896318315866.

[16] Francis, Freceena and Mohan, Maya. "ARIMA Model based Real Time Trend Analysis for Predictive Maintenance". In: *2019 3rd International conference on Electronics, Communication and Aerospace Technology (ICECA)*. 2019, pp. 735–739. doi: 10.1109/ICECA.2019.8822191.

[17] Fruehe, John and Pike, Jimmy. *PREDICTIVE MAINTENANCE: A PARADIGM SHIFT*. Tech. rep. United States: Moor Insights Strategy, 2016.

[18] Gensler, Andre, Henze, Janosch, Sick, B., and Raabe, Nils. "Deep Learning for solar power forecasting — An approach using AutoEncoder and LSTM Neural Networks". In: *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)* (2016), pp. 002858–002865.

[19] Group, Data Science. *Data Science and Machine Learning in the Internet of Things and Predictive Maintenance*. Tech. rep. Germany: SAP Predictive Maintenance and Service, 2016.

[20] Gutschi, Clemens, Furian, Nikolaus, Suschnigg, Josef, Neubacher, Dietmar, and Voessner, Siegfried. "Log-based predictive maintenance in discrete parts manufacturing". In: *Procedia CIRP* 79 (2019). 12th CIRP Conference on Intelligent Computation in Manufacturing Engineering, 18-20 July 2018, Gulf of Naples, Italy, pp. 528–533. issn: 2212-8271. doi: https://doi.org/10.1016/j.procir.2019.02.098. url: https://www.sciencedirect.com/science/article/pii/S221282711930215X.

[21] Haixiang, Guo, Yijing, Li, Shang, Jennifer, Mingyun, Gu, Yuanyue, Huang, and Bing, Gong. "Learning from class-imbalanced data: Review of methods and applications". In: *Expert Systems with Applications* 73 (2017), pp. 220–239. issn: 0957-4174. doi: https://doi.org/10.1016/j.eswa.2016.12.035. url: https://www.sciencedirect.com/science/article/pii/S0957417416307175.

[22] He, Pinjia, Zhu, Jieming, He, Shilin, Li, Jian, and Lyu, Michael R. "Towards Automated Log Parsing for Large-Scale Log Data Analysis". In: *IEEE Transactions on Dependable and Secure Computing* 15.6 (2018), pp. 931–944. doi: 10.1109/TDSC.2017.2762673.

[23] Ho, S.L, Xie, M, and Goh, T.N. “A comparative study of neural network and Box-Jenkins ARIMA modeling in time series prediction”. In: *Computers Industrial Engineering* 42.2 (2002), pp. 371–375. issn: 0360-8352. doi: https://doi.org/10.1016/S0360-8352(02)00036-0. url: https://www.sciencedirect.com/science/article/pii/S0360835202000360.

[24] Information, Reed Business. “Annual Report and Forecast”. In: *Supplement to Construction Equipment* (2007).

[25] Ishwaran, Hemant, Kogalur, Udaya B, Blackstone, Eugene H, Lauer, Michael S, et al. “Random survival forests”. In: *Annals of Applied Statistics* 2.3 (2008), pp. 841–860.

[26] Jadhav, Sujit Shivaji and Salgude, Rohit. “Downtime Cost of Construction Equipment”. In: *International Journal of Science Technology Engineering* (2019), pp. 20–23.

[27] Kleinbaum, David G, Dietz, K, Gail, M, Klein, Mitchel, and Klein, Mitchell. *Logistic regression*. Springer, 2002.

[28] Lane, Peter, Clarke, Daoud, and Hender, Paul. “On developing robust models for favourability analysis: Model choice, feature sets and imbalanced data”. In: *Decision Support Systems* 53 (2012), pp. 712–718. doi: 10.1016/j.dss.2012.05.028.

[29] Li, Jundong, Cheng, Kewei, Wang, Suhang, Morstatter, Fred, Trevino, Robert P., Tang, Jiliang, and Liu, Huan. “Feature Selection: A Data Perspective”. In: 50.6 (2017). issn: 0360-0300. doi: 10.1145/3136625. url: https://doi.org/10.1145/3136625.

[30] Li, Zhiguo, Zhou, Shiyu, Choubey, Suresh, and Sievenpiper, Crispian. “Failure event prediction using the Cox proportional hazard model driven by frequent failure signatures”. In: *IIE transactions* 39.3 (2007), pp. 303–315.

[31] Lima, R. and Pereira, Adriano M. “A Fraud Detection Model Based on Feature Selection and Undersampling Applied to Web Payment Systems”. In: *WI-IAT*. 2015.

[32] Lin, Wei-Chao and Tsai, Chih-Fong. “Missing value imputation: a review and analysis of the literature (2006–2017)”. In: *Artificial Intelligence Review* 53.2 (2020), pp. 1487–1509.

[33] Loyola-González, Octavio, Martínez-Trinidad, José Francisco, Carrasco-Ochoa, Jesús, and García-Borroto, Milton. “Study of the Impact of Resampling Methods for Contrast Pattern based Classifiers in Imbalanced Databases”. In: *Neurocomputing* In press (2015). doi: 10.1016/j.neucom.2015.04.120.

[34] Mabroukeh, Nizar R and Ezeife, Christie I. “A taxonomy of sequential pattern mining algorithms”. In: *ACM Computing Surveys (CSUR)* 43.1 (2010), pp. 1–41.

[35] Maldonado, Sebastián and López, Julio. “Imbalanced Data Classification Using Second-Order Cone Programming Support Vector Machines”. In: *Pattern Recogn.* 47.5 (2014), pp. 2070–2079. issn: 0031-3203. doi: 10.1016/j.patcog.2013.11.021. url: https://doi.org/10.1016/j.patcog.2013.11.021.

[36] Miller Jr, Rupert G. *Survival analysis, 2nd edition*. John Wiley & Sons, 2011. url: https://www.wiley.com/en-us/Survival+Analysis%5C%2C+2nd+Edition-p-9781118031063.

[37] Nations, United. *THE 17 GOALS - Sustainable Development*. https://sdgs.un.org/goals. Accessed: 2021-05-26.

[38] Prytz, Rune, Nowaczyk, Sławomir, Rögnvaldsson, Thorsteinn, and Byttner, Stefan. “Predicting the need for vehicle compressor repairs using maintenance records and logged vehicle data”. In: *Engineering applications of artificial intelligence* 41 (2015), pp. 139–150.

[39] Prytz, Rune, Nowaczyk, Sławomir, Rögnvaldsson, Thorsteinn, and Byttner, Stefan. “Predicting the need for vehicle compressor repairs using maintenance records and logged vehicle data”. In: *Engineering Applications of Artificial Intelligence* 41 (2015), pp. 139–150. issn: 0952-1976. doi: https://doi.org/10.1016/j.engappai.2015.02.009. url: https://www.sciencedirect.com/science/article/pii/S0952197615000391.

[40] Richards, SJ. “A handbook of parametric survival models for actuarial use”. In: *Scandinavian Actuarial Journal* 2012.4 (2012), pp. 233–257.

[41] Rivera, Domingo Llorente, Scholz, Markus R., Bühl, Christoph, Krauss, Markus, and Schilling, Klaus. “Is Big Data About to Retire Expert Knowledge? A Predictive Maintenance Study”. In: *IFAC-PapersOnLine* 52.24 (2019). 5th IFAC Symposium on Telematics Applications TA 2019, pp. 1–6. issn: 2405-8963. doi:

https://doi.org/10.1016/j.ifacol.2019.12.364. url: https://www.sciencedirect.com/science/article/pii/S2405896319322645.

[42] Sari, Emelia, Shaharoun, Awaluddin Mohamed, Ma'aram, Azanizawati, and Yazid, A. Mohd. "Sustainable Maintenance Performance Measures: A Pilot Survey in Malaysian Automotive Companies". In: *Procedia CIRP* 26 (2015). 12th Global Conference on Sustainable Manufacturing – Emerging Potentials, pp. 443–448. issn: 2212-8271. doi: https://doi.org/10.1016/j.procir.2014.07.163. url: https://www.sciencedirect.com/science/article/pii/S2212827114012128.

[43] Seger, Cedric. *An investigation of categorical variable encoding techniques in machine learning: binary versus one-hot and feature hashing*. 2018.

[44] Sipos, Ruben, Fradkin, Dmitriy, Moerchen, Fabian, and Wang, Zhuang. "Log-based predictive maintenance". In: *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2014). doi: 10.1145/2623330.2623340.

[45] Song, Xuan, Kanasugi, Hiroshi, and Shibasaki, Ryosuke. "Deeptransport: Prediction and simulation of human mobility and transportation mode at a citywide level". In: *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*. 2016, pp. 2618–2624.

[46] Svetnik, Vladimir, Liaw, Andy, Tong, Christopher, Culberson, J Christopher, Sheridan, Robert P, and Feuston, Bradley P. "Random forest: a classification and regression tool for compound classification and QSAR modeling". In: *Journal of chemical information and computer sciences* 43.6 (2003), pp. 1947–1958.

[47] Timm, Fabian. *Predictive Maintenance*. Tech. rep. Issue 7. Germany, Berlin: Deloitte Analytics Institute, 2017.

[48] Traini, Emiliano, Bruno, Giulia, D'Antonio, Gianluca, and Lombardi, Franco. "Machine Learning Framework for Predictive Maintenance in Milling". In: *IFAC-PapersOnLine* 52.13 (2019). 9th IFAC Conference on Manufacturing Modelling, Management and Control MIM 2019, pp. 177–182. issn: 2405-8963. doi: https://doi.org/10.1016/j.ifacol.2019.11.172. url: https://www.sciencedirect.com/science/article/pii/S240589631931122X.

[49] *Volvo Construction Equipment Global*. https://www.volvoce.com/global/en/. Accessed: 2021-05-26.

[50] *Volvo Construction Equipment Global: Articulated Haulers*. https://www.volvoce.com/global/en/our-offer/articulated-haulers/. Accessed: 2021-06-06.

[51] *Volvo Construction Equipment Media Library*. https://images.volvoce.com/categories. Accessed: 2021-06-06.

[52] Voronov, Sergii. "Machine Learning Models for Predictive Maintenance". PhD thesis. Linköping University, Vehicular Systems, Faculty of Science Engineering, 2020, p. 297. isbn: 9789179299231. doi: 10.3384/diss.diva-162649.

[53] Wang, J., Li, C., Han, S., Sarkar, S., and Zhou, X. "Predictive maintenance based on event-log analysis: A case study". In: *IBM Journal of Research and Development* 61.1 (2017), 11:121–11:132. doi: 10.1147/JRD.2017.2648298.

[54] Wang, Jinjiang, Zhang, Laibin, Duan, Lixiang, and Gao, Robert. "A new paradigm of cloud-based predictive maintenance for intelligent manufacturing". In: *Journal of Intelligent Manufacturing* 28 (2017), pp. 1125–1137. doi: 10.1007/s10845-015-1066-0.

[55] Wirth, Rüdiger and Hipp, Jochen. "CRISP-DM: Towards a standard process model for data mining". In: *Proceedings of the 4th international conference on the practical applications of knowledge discovery and data mining*. Vol. 1. Springer-Verlag London, UK. 2000.

[56] Xie, Xiaofeng, Wu, Di, Liu, Siping, and Li, Renfa. "IoT data analytics using deep learning". In: *arXiv preprint arXiv:1708.03854* (2017).

[57] Yam, RCM, Tse, PW, Li, L, and Tu, P. "Intelligent predictive decision support system for condition-based maintenance". In: *The International Journal of Advanced Manufacturing Technology* 17.5 (2001), pp. 383–391.

[58] Yozgatligil, Ceylan, Aslan, Sipan, Iyigun, Cem, and Batmaz, Inci. "Comparison of missing value imputation methods in time series: the case of Turkish meteorological data". In: *Theoretical and applied climatology* 112.1 (2013), pp. 143–167.

[59] Yu, Wenjin, Dillon, T., Mostafa, Fahed, Rahayu, W., and Liu, Y. "A Global Manufacturing Big Data Ecosystem for Fault Detection in Predictive Maintenance". In: *IEEE Transactions on Industrial Informatics* 16 (2020), pp. 183–192.

[60] Zaki, Mohammed Javeed, Lesh, Neal, and Ogihara, Mitsunori. “PlanMine: Sequence Mining for Plan Failures.” In: *KDD*. Vol. 98. 1998, pp. 369–373.

[61] Zhang, Jianjing, Wang, Peng, Yan, Ruqiang, and Gao, Robert X. “Deep Learning for Improved System Remaining Life Prediction”. In: *Procedia CIRP* 72 (2018). 51st CIRP Conference on Manufacturing Systems, pp. 1033–1038. issn: 2212-8271. doi: https://doi.org/10.1016/j.procir.2018.03.262. url: https://www.sciencedirect.com/science/article/pii/S2212827118304335.

[62] Zhang, Jianjing, Wang, Peng, Yan, Ruqiang, and Gao, Robert X. “Long short-term memory for machine remaining life prediction”. In: *Journal of Manufacturing Systems* 48 (2018). Special Issue on Smart Manufacturing, pp. 78–86. issn: 0278-6125. doi: https://doi.org/10.1016/j.jmsy.2018.05.011. url: https://www.sciencedirect.com/science/article/pii/S0278612518300803.

[63] Zhang, Weiting, Yang, Dong, and Wang, Hongchao. “Data-Driven Methods for Predictive Maintenance of Industrial Equipment: A Survey”. In: *IEEE Systems Journal* 13.3 (2019), pp. 2213–2227. doi: 10.1109/JSYST.2019.2905565.