Explainable and interpretable remaining useful life prediction for right-censored critical machine components

Bannink, Anne W.

Award date:
2023

Department of Industrial Engineering and Innovation Sciences
Operations Planning Accounting and Control Research Group

Explainable and interpretable remaining useful life prediction for right-censored critical machine components

Master thesis

A.W. (Anne) Bannink 1225173

Supervisors:

dr. C. (Claudia) Fecarotti,	Eindhoven University of Technology
dr. Z. (Zaharah) Bukhsh,	Eindhoven University of Technology
dr. L. (Laura) Genga,	Eindhoven University of Technology
T. (Thijs) Gerrits,	DPD Netherlands

Eindhoven, August 4, 2023

Eindhoven University of Technology
School of Industrial Engineering
Master Thesis Operations Management and Logistics

Keywords: remaining useful life prediction, right-censored data, time-to-event data, survival ensembles, interpretable and explainable AI.

Abstract

Data-driven remaining useful life prediction methods have rapidly increased in popularity due to the availability of Big Data and the Internet of Things. This increased popularity originates from their ability to improve machine reliability and availability and to decrease maintenance costs and downtime. These improvements are necessary for companies to ensure service to their customers. Numerous studies demonstrate the effectiveness of several machine learning models, but limited studies focus on the application of remaining useful life prediction to real-world data. These data pose additional challenges. Most importantly, the data are often censored, as run-to-failure observations are rare and data are noisy. Therefore, this paper studies the applicability of survival RUL prediction methods to real-world, noisy and right-censored data. An important extension is the model explainability and interpretability. A survival tree, gradient-boosting model and random survival forest were fitted to the sorting machine of DPD Netherlands, which are all decision tree-based methods. These trees cluster the components based on usage and fit an estimator to each cluster. This study concludes that these models provide excellent tools for maintenance prioritisation, as their estimation of the order of failure is satisfactory. However, the exact moment of failure remains difficult to estimate, especially when the independent variables are subject to high variance. Therefore, this study extends previous research with the notion that prediction accuracy is insufficient for implementation in practice. However, the methodology seems promising as the order of failure can already be accurately assessed with right-censored, noisy data. With further research into data-driven remaining useful life prediction to data containing high variance, the applicability of these methods in the industry seems encouraging.

Executive summary

Introduction

Nowadays, companies are preparing for Industry 4.0 which is characterized by intelligent systems and Internet-based solutions (Li, Wang, & He, 2016). Within Industry 4.0, the rise of these advanced analytics and the Internet of Things (IoT) augments the possibilities for predicting the useful life of assets. Remaining useful life (RUL) prediction of assets improves maintenance decision-making by allowing timely and better-informed maintenance decisions. Therefore, many companies are interested in increasing their machine availability and reliability and decreasing maintenance costs and machine downtime to improve customer satisfaction (Li et al., 2016).

DPD Netherlands is one of the companies that are interested in remaining useful life predictions. Their sorting system is essential for the timely delivery of parcels to their clients. This system consists of four critical components: carts, MCB units, motors and crossbelts. The sorting system contains 1640 of each component, which are all critical. If one of them fails, the entire system is down. This has a significant impact on their customer satisfaction. Therefore, an accurate maintenance strategy is crucially important to DPD.

Component replacements have three triggers: crashes, failures and preventive replacements. Crashes occur due to parcels moving from their original location and destroying components. Failures during operation follow regular degradation. Finally, maintenance engineers perform preventive maintenance when a degraded component is found during the inspections. Currently, the sorting machine is inspected based on a fixed interval, irrespective of degradation. However, the usage of the machine is a determining factor in degradation. Therefore, usage features should be included in the RUL prediction model. As noted, not all components are replaced due to failure. Therefore, the actual lifetime of these components is unknown, which is referred to as censored observations. However, these observations cannot be excluded from the model, as there are too few uncensored observations, and these censored observations still hold valuable information. Thus, the RUL prediction model should be able to deal with these observations. Overall, this research aimed to investigate possibilities for data-driven remaining useful life prediction based on the usage of the sortation system of DPD.

Literature

Data-driven remaining useful life prediction models have gained popularity as a result of their capacity to manage large amounts of data and capture non-linear correlations between features and the remaining useful life. Accordingly, several approaches have been suggested, such as artificial neural networks, support vector machines, Cox's proportional hazards model, and ensemble methods. Ensemble methods can deal with the challenges presented by real-world, noisy, tabular and censored data. Dealing with censored observations is often studied in the bioinformatics field. Although these models are extremely relevant for the remaining useful life of machine components, very few studies have been published that study their accuracy in this field. Moreover, trusting the model and its predictions is essential for deployment. Therefore, this study extends the applicability of ensemble methods with their explainability and interpretability.

Data preparation

This study was performed on one dataset per component. These datasets were created by integrating data on the parcels and system alarm notifications with the data on the component lifetimes. Missing values were filled, and erroneous values were corrected. Consequently, relevant features were determined by interviews with system experts. Thereafter, the dataset contained 1839, 1940, 1941 and 1943 observations for the cart, MCB, motor and crossbelt, respectively. Table 1 shows the descriptives of these datasets. As a significant amount of data is censored, the observations are undersampled to balance the datasets.

Component		Degraded components	Crashed components	Functioning components	Total number of observations	Percentage censored
Cart		172	27	1640	1839	90.50%
	MCB	264	36	1640	1940	89.39%
Carrier	Motor	255	36	1640	1931	86.79%
	Crossbelt	267	36	1640	1943	86.26%

Table 1: Description of the number of units per label per subcomponent

Methodology and modelling

As noted, model explainability is essential for trusting artificial intelligence models. Model explainability refers to the logic of the algorithm, and is inherent to the type of model used (Linardatos et al., 2020). Decision tree-based algorithms quickly generate findings with familiar instances that are explainable. Therefore, we fit three decision tree-based models: a survival tree, a gradient-boosting model and a random survival forest. The latter two methods combine several decision trees. These methods are evaluated based on their concordance index, root mean squared error (RMSE), mean absolute percentage error (MAPE), and the trade-off between underestimating and overestimating the remaining useful life. We split the dataset into 70% for training and 30% for testing. The training set is split into 80% training and 20% validation datasets to perform hyperparameter tuning over the minimum number of samples per split and the number of trees for the gradient-boosting model and random survival forest. Thereafter, the results are evaluated based on the test set (the remaining 25% of the data).

Results

For each of the components, a satisfactory concordance index is achieved. The best value for the cart, MCB, motor and crossbelt are 82.35%, 80.53%, 82.47% and 82.78%, respectively. For the cart and crossbelt, the random survival forest is optimal, whereas the survival tree produces the best result for the MCB and motor. Figure 1 shows the comparison of these models. The gradient-boosting model never performs the best in terms of concordance index. However, this model produces the best RMSE for all components. The survival tree generally gives the best MAPE and is the shortest to train. In general, the RMSE and MAPE of the three RUL prediction models were very high. This means that the prediction differs significantly from the actual value.

There can be several reasons for the models' high RMSE and MAPE in general. First off, there is an excessive amount of long-term uncertainty, leading to bigger prediction errors over time. As a result, the order of failure (C-index) remains accurate even while the RMSE grows. Additionally, the size of the dataset may be responsible for the high RMSE and MAPE values.

The likelihood is that there are more failure modes than there are observations along each path.

The survival tree, gradient-boosting model and random survival forest were also investigated for their global and local interpretability. Global interpretability refers to the inner logic of the models. The partial dependence plots showed that the three RUL prediction models have very different interpretations of the data. Then, local interpretability refers to the logic behind an individual prediction. Local Interpretable Model-agnostic Explanations (LIME) were deployed to locally interpret the models. Here, the three models showed similar relations between the feature (total weight) and its effect on the remaining useful life.

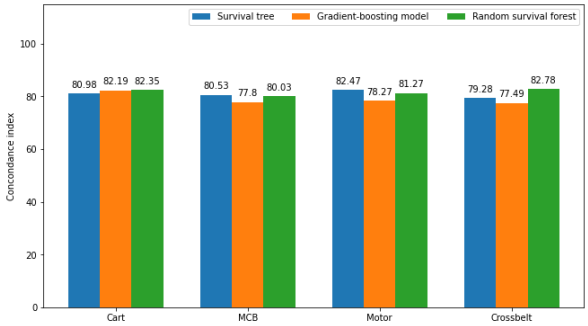


Figure 1: Comparison of concordance index per component for the gradient-boosting model and random survival forest

Conclusion and recommendations

To summarize, this study aimed to create a model that would anticipate, depending on use, how long essential parts of the sortation machine at DPD Netherlands' Oirschot facility would remain functional. The survival tree, gradient-boosting model and random survival forest provide excellent estimates of the sequence of failure. Therefore, these three models are useful for inspection prioritisation. However, the estimations of the moment of failure are inaccurate. As the survival tree is the most explainable model and interpretability is achievable for all three models, we recommend DPD to use the survival tree for maintenance prioritisation of the carts, MCB units, motors and crossbelts. Once more component failures have been observed, the models should be evaluated again.

Foreword

This Master thesis is submitted for the Master program 'Operations Management and Logistics' with the specialization track 'Manufacturing Systems Engineering' at the Eindhoven University of Technology (TU/e) and has been performed for the 'Operations Planning, Accounting and Control' group of the faculty of 'Industrial Engineering Innovation Sciences'. In this Master thesis, a business problem from the field of operations management and logistics is analysed at DPD Netherlands. DPD Netherlands is provided with a remaining useful life prediction model. In addition, directions for improvement will be suggested. The report follows the guidelines provided by the Eindhoven University of Technology and aims to satisfy the demands of DPD Netherlands. I would like to take this opportunity to thank the people who guided and supported me throughout this final project.

I am particularly thankful to Dr. Claudia Fecarotti, my first supervisor and mentor, whose unwavering support has been invaluable throughout my Master's Degree. During my thesis, she looked out for me and provided me with valuable input for my research. Moreover, I would like to thank my second assessor Dr. Zaharah Bukhsh for her guidance and assistance throughout my thesis.

I would like to express my gratitude to Thijs Gerrits, my company supervisor, for giving me the opportunity to be involved in this interesting project. His continuous guidance and profound understanding of the system have been instrumental in shaping my research. Additionally, I extend my thanks to Sander Lans, Yannick Jacobs, and my other colleagues at DPD Netherlands for their support and for creating a pleasant working environment. You all provided me with valuable insights and a fun conversation when needed.

Lastly, I would like to thank my friends and family for their unwavering support throughout the past months of my thesis and my entire academic journey. Your encouragement and countless moments of laughter and much-needed coffee breaks have made the stress melt away. I am forever grateful for the love and strength you have provided me, making my time in Eindhoven filled with joy and cherished memories.

Thank you all,

Anne Bannink

List of Figures

List of Tables

List of Abbreviations

The table below presents an overview of all abbreviations used in this research in alphabetical order.

Abbreviations	Description
ANNs	Artificial neural networks
B2B	Business-to-business
B2C	Business-to-customer
C2B	Customer-to-business
C-index	(Harrells) Concordance Index
CPH	Cox's proportional harzard (model)
CRISP-DM	CRoss Industry Standard Process for Data Mining
DBS	Deflected Bellow Supervision (fault)
FS	Feature Selection
GB	Gradient-Boosted (model)
IOB	Item Overhanging Belt (fault)
IoT	Internet of Things
KPI	Key Performance Indicators
Local Interpretable Model-agnostic Explanations	LIME
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
MCB	Miniature Circuit Board
MSE	Mean Squared Error
P2P	Peer-to-peer
PDPs	Partial Dependence Plots
RMSE	Root Mean Squared Error
RSF	Random Survival Forest
RUL	Remaining Useful Life
ST	Survival (decision) tree
SVM	Support Vector Machine

Contents

1. Introduction

Nowadays, companies are preparing for Industry 4.0, referring to the fourth industrial revolution characterized by intelligent systems and Internet-based solutions (Li, Wang, & He, 2016). Within Industry 4.0, the rise of these advanced analytics and the Internet of Things (IoT) augments the possibilities for predicting the useful life of assets. Predicting this remaining useful life (RUL) of assets improves maintenance decision-making by allowing timely and better-informed maintenance decisions. Hence, RUL predictions can significantly impact maintenance strategies (Barlow, 2015). Therefore, many companies are willing to face the challenge of assessing the diversity of developments summarized in the term Industry 4.0 and developing appropriate RUL prediction strategies (Li et al., 2016).

One of these companies is DPD Netherlands (onwards referred to as DPD). DPD is a parcel delivery service provider based in the Netherlands. As part of the GeoPost group, a leading European parcel delivery network, DPD offers domestic and international shipping services to consumers and businesses. Therefore, the shipping services include business-to-customer (B2C), person-to-person/peer-to-peer (P2P) and return streams (C2B) and business-to-business (B2B). DPD combines cutting-edge technology and their European network to ensure rapid and effective delivery to clients, with an emphasis on offering a reliable, flexible, and sustainable delivery experience. This network consists of 11 local depots and two international hubs. From these hubs, parcels are shipped to 230 countries. These packages are picked up at the sender and transported to an unloading dock at the local depot or international hub, where the parcels are sorted by a sortation machine and routed to their loading dock. From these docks, these packages are loaded into vans and shipped to the receiver, which concludes their operations. Thus, DPD's performance is widely dependent on its sortation systems. These sorting systems have become vital to DPD's operations as the number of packages increases tremendously due to the rapid expansion of e-commerce. In this industry, ensuring timely delivery is decisive for customer satisfaction. Therefore, minimizing unplanned system downtime is essential to DPD. Limiting unplanned downtime is often managed by designing maintenance strategies, which has proved challenging for this system. To provide a comprehensive problem understanding, Section 1.1 describes the problem context and Section 1.2 delves into a detailed explanation of the problem.

1.1 Problem context

This section discusses the sortation system installed in the international hub located in Oirschot, which is the focus of this thesis. The sorting of packages at the international hub is automated by a sizable parcel handling system. Parcels are loaded from trucks onto the parcel sorting system via an induct at one of the 66 unloading doors. An induct can be seen as the entryway onto the highway. Then, the sorting system (e.g. the highway) routes the packages to the corresponding chute, which can be seen as the exit of the highway. In total, the system encompasses 2.1 kilometres of conveyor belts which can sort 25.000 parcels per hour. When the parcel arrives at the correct chute, the package is dropped off and loaded into the truck. This completes the sortation process which is summarized in Figure 1.2.

The parcels are transported from origin A to destination B via a carriage system comprised of two distinct components: a cart and a carrier. Although physically integrated, these elements are treated as distinct entities. The cart travels through the rails of the system to transport the carrier, which in turn is equipped with a conveyor belt, facilitating the

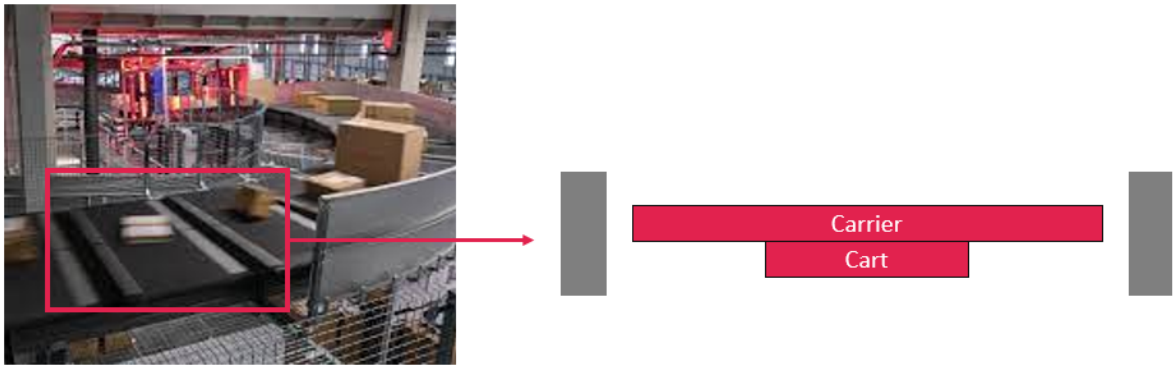


Figure 1.1: The sortation system (left) and a schematic representation of the cart and carrier (right)

loading and unloading of parcels. These components are shown in Figure 1.1, where the left of the picture shows the real system and the right side of the figure represents the highlighted components. During transportation, sensors scan each package at the entry and exit of the sortation system. These sensors also weigh each parcel and measure its length, width, and height at the induct. The system keeps track of various information related to each package, such as which chute the parcel exits the system from, whether it was transported on one or two components, how many times it circled the system and the time it spent in the system. Additionally, the system records whether any alarms were triggered, and if so, it registers the date, alarm type, duration, and the component on which the alarm occurred. Finally, for each cart or carrier, the location in the sortation system is recorded, and when it has been replaced.

This sortation system consists of two sorting machines, labelled one and two, which are connected. Sorter one facilitates the loading and unloading of 17.000 parcels per hour for both the hub and depot, whereas Sorter two adds a capacity of 8.000 parcels for the hub. Sorter one consists of 1120 carts and 1120 carriers, whereas the smaller Sorter two consists of 520 carts and 520 carriers - each carrier being attached to one cart. These components operate in series, so each component is critical, as a failure of either one cart or carrier leads to a complete machine failure. If Sorter two experiences a malfunction, Sorter one can still operate, albeit with a reduced capacity of 17.000. However, if Sorter one fails, the entire system is down for at least 30 minutes, causing a delay for 12.500 parcels. As the system consists of 3280 critical components, an effective maintenance strategy is fundamental.

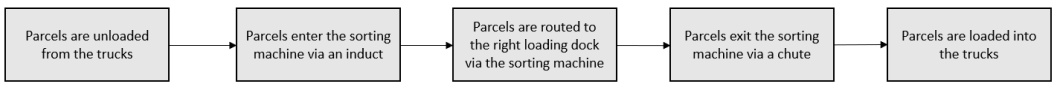


Figure 1.2: Internal logistics process

1.2 Problem definition

To design an effective maintenance strategy, DPD aspires to gain insights into the quality of components by predicting their remaining useful life (RUL). Therefore, the explainability of the RUL prediction is essential to provide this insight. The remaining useful life of an item is the pertinent life left of that component at a particular time in operation (Si, Wang, Hu,

& Zhou, 2011). Chapter 2 presents a further explanation of RUL prediction. An accurately predicted RUL provides practical information for maintenance scheduling because it allows the plan to be tailored to each component's expected residual lifespan. Moreover, unexpected breakdowns can be prevented, resulting in increased production time, lower downtime and more simple spare parts planning (Buchaiah & Shakya, 2022).

To avoid these issues, DPD seeks to enhance its maintenance strategy by fully utilizing the useful life of its parts. This approach is subject to several constraints. Firstly, the remaining useful life (RUL) prediction must be data-driven to facilitate generalization to other locations, as it is less reliant on the specific characteristics of each sortation system, which is detailed in Section 1.1. Furthermore, the complex nature of the system limits the possibility of modelling it based on physics. Secondly, the system's complexity poses a challenge to measuring its condition using sensors. As a result, DPD aims to employ system usage information, such as parcel weight and volume, as a means of predicting RUL since system usage is inherently linked to degradation. Lastly, the RUL prediction approach should be comprehensible to DPD's maintenance engineers and other employees. This understandability is essential to minimize resistance to maintenance decisions by enabling employees to understand the rationale behind specific part replacements. The current maintenance practices are described in Subsection 1.2.1.

1.2.1 Maintenance strategy

DPD observes three triggers for component replacement. Firstly, components might get critically damaged due to a crash. Such a crash can occur when a parcel moves from its initial position, which is due to the instability of that package. Secondly, regular wear encountered during inspection triggers a preventive component replacement, which is fully based on the judgement of the maintenance engineer. DPD agreed on a maintenance contract with the supplier of the machine. Thirdly, a component can fail during operation, after which the maintenance team immediately replaces the component to restore the machine's availability. A crash, regular wear and failed components all trigger a component replacement, but the crashes and failures demand immediate replacement during operation. Crashes and preventive replacements influence the data as the end-of-life of the component remains unknown. This is also known as censored data. Censored data occurs when the failure data are incomplete because units are removed from operation before their failure (Ebeling, 2010). In this case, the failure times for some units are known only to be after a specified time, which is defined as right-censored data. Figure 1.3 visualises this concept.

The supplier maintains DPD's sorting system based on a fixed time interval. Each component is inspected every 26 weeks during their scheduled downs. Due to the nature of the parcel industry, DPD's sorting system is operational from 13:00 until 07:00. During the morning, from 07:00 to 13:00, trucks are still driving to deliver the parcels, causing the system to be idle. However, not all components can be replaced in this daily maintenance window due to spare parts and time constraints. Therefore, DPD groups the maintenance of these components, meaning a few items are reviewed per day, the following couple of units on the next day, and so forth. For example, components 1-6 are maintained on day one and items 7-12 the day after until each of the 1640 parts has been inspected. The inspection frequency is determined by the supplier's judgment of the expected useful life of the items, the order is solely based on convenience. Therefore, DPD's sortation machine is currently maintained by a strategy

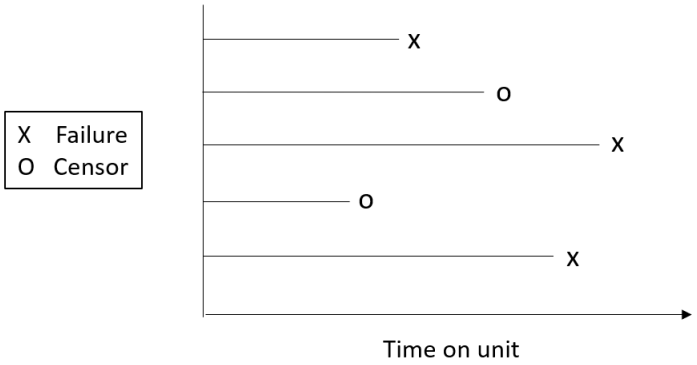


Figure 1.3: Censoring (from Ebeling (2010))

that completely disregards its degradation.

This constant-interval inspection policy does not consider data on component use. Instead, components are examined after a set time interval, irrespective of failure behaviour (Price & Mathew, 2000). As a result, this technique may result in the replacement of comparably decent components more frequently than necessary, resulting in resource waste. While the actual failure behaviour of the components is currently not part of DPD's maintenance plan, this information can assist them in determining a more advantageous approach in terms of part usage and machine availability. Moreover, if the number of operating hours vary widely, as is the case at DPD, this constant interval replacement policy is unsuitable (Tinga & Janssen, 2014). Therefore, this research aims to develop a remaining useful life prediction model, that allows DPD to perform maintenance to their critical components, based on the RUL of each component. Further details of this research are provided in the next section.

1.2.2 Research definition

This subsection presents the research questions and briefly introduces the methods used to answer these questions. As discussed, this research aspires to predict the RUL of the critical components (cart and carrier) for DPD Netherlands based on their usage. Usage data serves as input to the model as the component degradation is influenced by the system's usage. As noted by the system's supplier, the wear largely depends on the severity of the usage. Other clients with limited usage, for example, small and light shipments, report significantly lower degradation rates. Moreover, most companies have not yet implemented sensors into their machines. Luckily, usage information is generally available in companies, increasing the practical relevance of this project. Therefore, the main question posed in this research is:

1. *How to estimate the remaining useful life of the carts and carriers within DPD Netherlands' sorting system in Oirschot based on their usage?*

The remaining useful life prediction process consists of four main steps: data extraction, feature extraction and selection, model building and training and RUL prediction and evaluation (Ferreira & Gonçalves, 2022). Thus, the input data is the priority for developing a data-driven remaining useful life prediction model. Therefore, the first and second subquestions hold:

1.1 What data on system usage is recorded by DPD?
1.2 Which usage-based indicators are relevant for the remaining useful life prediction?

Question 1.1 will be answered by interviews with two parties: DPD and the system's supplier. Both parties are knowledgeable about the system and record information about it. Therefore, we will host several interviews with maintenance managers from both companies. As the system is so complex, we expect that a simple inspection of the available data, will not yield a clear definitive answer on the useful indicators. Therefore, expert opinion is used to define useful indicators. After the indicators are identified, the data are extracted from the information systems. After the data are collected, the model should be built and trained. Thus, the next subquestion is as follows:

1.3 Which data-driven remaining useful life prediction method should be used?

The literature will form the basis for finding an accurate RUL prediction method that can deal with the constraints of this context (e.g. usage-based, noisy and right-censored data and model interpretability). The consideration of censored values is crucial in this research since these observations tell us something about the expected lifetime.

Next, performance indicators should be defined to evaluate the remaining useful life prediction method. Then, the model's performance should be investigated. This leads to the following subquestions:

1.4 Which performance metrics should be used to evaluate the remaining useful life prediction for the key components?
1.5 How to evaluate model performance?

These questions will be answered in three stages. First, suitable performance indicators will be sought in the literature to assess the predictive models. Second, from the usual key performance indicators (KPIs) in data-driven methods, such as the mean absolute error, mean squared error etc. the applicable indicators are selected based on interviews with the system experts. Third, the model's performance will be evaluated using these KPIs. Finally, DPD necessitates that the model is explainable and interpretable. Therefore, the factors that influenced the prediction are essential to investigate:

1.6 What factors influenced the model to predict this remaining useful life?

Machine learning interpretability is a highly active research area. Therefore, the literature will be used to search for suitable interpretability methods. Based on these answers, we can provide a comprehensive overview to DPD with definitive recommendations for the RUL prediction of the critical components of the sortation machine located at DPD's hub in Oirschot.

1.3 Research scope

This research focused on developing an RUL prediction model for the critical components of DPD Netherlands, as their failures significantly interrupt operations. As noted by (Tiddens,

Braaksma, & Tinga, 2018), suitable candidates for predictive maintenance are components whose failures have the highest impact on the availability and maintainability of the system. Moreover, these components should have a low failure rate, as components with a high failure rate should be eliminated from the system. As mentioned, in this case, carts and carriers are these critical components. The latter unit is split into three parts: the motor controller board (MCB), motor and crossbelt. The MCB is a physical platform and electrical circuit to interconnect the various electronic components and enables the functionality of the motor. The motor drives the movement of the crossbelt, which is a small conveyor belt that moves perpendicular to the cart when parcels enter or leave the carrier.

Another restraint is that this research focuses on the main sorter one and two of the location in Oirschot. Therefore, this research includes the hub and depot area. Moreover, we incorporate the parcels and alarm notifications that have been registered from the system's installation in March 2019 until April 2023 for this location. However, the model design should effortlessly be generalised to other facilities of DPD with a similar sortation system.

Finally, this master thesis excludes the deployment of the methodology into DPD's systems. The deployment phase consists of integrating the model into DPD's systems and using the outcome after we have evaluated the model. Wirth & Hipp (2000) presents these stages in their Cross Industry Standard Process for Data Mining (CRISP-DM) visualised in Figure 1.4. Thus, the final stage of this thesis is evaluation, and the deployment phase is excluded from the scope of this thesis.

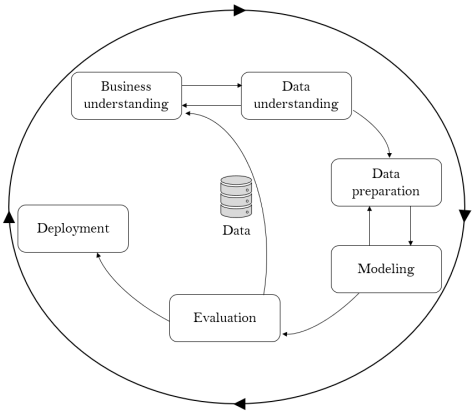


Figure 1.4: CRISP-DM framework

1.4 Research structure

This research is structured according to the RUL prediction process as defined by Ferreira & Gonçalves (2022). First, we summarize the relevant academic literature and introduce topics that are pertinent to this research in Chapter 2. Next, we move to the first step of the RUL prediction process, data preparation, in Chapter 3. Thereafter, we present the methodology and model building and training in Chapter 4. Then, Chapter 5 discusses the modelling, and Chapter 6 discusses the results. Finally, Chapter 7 concludes this report with the research conclusion, implications, recommendations, limitations and future research.

2. Literature review

This chapter summarizes the relevant literature for this study. Firstly, Section 2.1 presents maintenance strategies and their advantages and disadvantages. Secondly, Section 2.2 introduces remaining useful life prediction approaches, challenges and solutions. Thirdly, Section 2.3 describes the gap in literature to which this research will contribute.

2.1 Maintenance strategies

The availability of capital goods is essential to remain operational. Maintenance strategies are designed to improve this availability and can be defined as a set of rules describing the triggering mechanism for maintenance actions (Arts, 2017). Such efforts can be either unplanned or planned. Unplanned maintenance actions, also known as breakdown corrective maintenance, consist of interventions after a component has failed (Arts, 2017). However, this strategy poses significant requirements for spare parts availability. Therefore, breakdown corrective maintenance is mainly suitable for components not subject to wear. For other parts, planning maintenance actions can have significant benefits (Wu, Jennings, Terpenny, Gao, & Kumara, 2017). These interventions are known as preventive maintenance, which aspires to replace parts before failures to avoid breakdowns (Arts, 2017). These planned maintenance interventions can be determined by component age or usage (Arts, Basten, & Houtum, 2019). The main advantage of applying planned maintenance compared to unplanned maintenance is that it is typically executed much faster, as the interventions are known beforehand (Arts et al., 2019).

An advancement in planned maintenance is predicting maintenance tasks, in other words, predicting component failures. Prognosis deals with forecasting these failures by predicting the RUL of components (Buchaiah & Shakya, 2022). This remaining useful life is expressed according to the primary system measurement, which is industry-specific (Ferreira & Gonçalves, 2022). RUL prediction has become a predominant subject in quality and reliability research. Section 2.2 explains this active research area in more detail.

2.2 Remaining Useful Life prediction

Remaining useful life can be defined as the period during which an asset or property is expected usable for the purpose it was acquired. A more general definition for the RUL is the length from the current time to the end of the useful life or the time during which the component is able to perform a desired function (Si, Wang, Hu, & Zhou, 2011; Javed, Gouriveau, Zemouri, & Zerhouni, 2012). Therefore, RUL prediction is crucial in maintenance, reliability engineering, and prognostics, as this prediction allows for proactive maintenance and decision-making, improving system reliability and reducing maintenance costs. However, predicting the remaining useful life of components causes high demands on data access and quality as well as the capability to deal with these data. Moreover, prediction accuracy is of crucial importance for the magnitude of the impact on the number of unexpected failures (Li et al., 2016). Nonetheless, the reduction in unexpected failures, improved reliability, and financial gains from correctly predicting failures before they occur and acting upon that ensure that RUL is an active goal for many companies (Prytz, Nowaczyk, Rögnvaldsson, & Byttner, 2015). Thus, RUL prediction is a promising activity that benefits in the form of planning, safety, availability and maintenance cost reduction (Javed et al., 2012).

The RUL of an asset is a random variable, and it depends on the current age of the component, operation environment and health information. Frisk, Krysander, & Larsson (2014) illustrate this dependency. Let T be the random variable indicating the failure time, then the reliability or survival function at a given time t is the probability that the component survives past time t i.e. $T \geq t$:

$$R(t) = P(T \geq t) \tag{2.1}$$

Then, Frisk et al. (2014) note that the expected RUL is the difference between the expected total useful life and the current age. In other words, it is the expected life that the component has left, given the current age. Therefore, it is the conditional reliability, defined as:

$$E[RUL(t_0)] = \frac{1}{R(t_0)} \int_{t_0}^{\infty} R(t)dt - t_0 \tag{2.2}$$

where t_0 is the current component age. Therefore, the RUL of a component is a random variable dependent.

These formulas can be estimated in varying ways. Ferreira & Gonçalves (2022) and Buchaiah & Shakya (2022) define three types of RUL prediction methodologies: model-based, data-driven and hybrid methods (Figure 2.1). Model-based prognostics refer to approaches based on mathematical models of system behaviour derived from physical laws or probability distribution. To complement these models, data-driven prognostics refer to techniques that build predictive models using learning algorithms and large volumes of training data. Hybrid models combine data-driven and model-based methods to produce more accurate results (Ferreira & Gonçalves, 2022; Ahmadzadeh & Lundberg, 2014; Prytz et al., 2015).

All methodology types have explicit advantages and disadvantages. Model-based approaches include the proportional hazard rate and cumulative damage models. These techniques are challenging, expensive and time-consuming to develop as significant prior system knowledge is required. Assumptions and simplifications aid in dealing with this complexity but also reduce model accuracy. However, these models often provide high-precision prognostics (Ferreira & Gonçalves, 2022). Moreover, model-based approaches generally require less data than data-driven techniques (Ahmadzadeh & Lundberg, 2014). The latter methodologies include machine learning and statistical methods (Si et al., 2011). Data-driven approaches are less complex and expensive and are more applicable to the industry, as they provide a trade-off between complexity, cost, precision and applicability (Ferreira & Gonçalves, 2022). However, these techniques require large amounts of data with high quality to ensure these merits. Nonetheless, their high accuracy and fast response outweigh this disadvantage. Therefore, Section 2.2.1 will highlight these data-driven methods.

2.2.1 Data-driven Remaining Useful Life prediction

Recently, data-driven RUL prediction models have become increasingly popular due to their ability to handle large amounts of data and capture non-linear relationships between features and RUL. Numerous studies have demonstrated the effectiveness of several machine learning models, including artificial neural networks (ANNs), support vector machines (SVMs), Cox's proportional hazards model, and ensemble methods (Ahmadzadeh & Lundberg, 2014; Wu, Jennings, Terpenny, Gao, & Kumara, 2017).

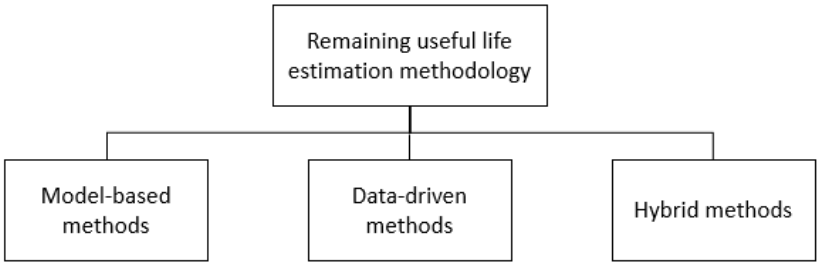


Figure 2.1: Remaining useful life prediction methodologies

Firstly, artificial neural networks (ANNs) simulate the intricate working process of human brains by establishing connections among numerous nodes within a complex layered structure. Therefore, these models are noted to have exceptional performance in the RUL prediction of complex systems (Lei et al., 2018). However, these models have low transparency and require significant training time (Ahmadzadeh & Lundberg, 2014). Moreover, they demand high-quality training data, which is often not available in industrial settings (Lei et al., 2018). Secondly, support vector machines (SVM) are superior to ANNs to deal with small sample sizes (Lei et al., 2018). Thus, they may be more suitable for the issues of RUL prediction where only limited measurements are available, as is often the case in real-world settings. However, parameter estimation remains a challenge for SVMs. Moreover, their performance is highly dependent on the selected kernel functions. Thirdly, Cox's proportional hazards model is a statistical technique that develops a semi-parametric model limiting the assumptions about the relationship between dependent and independent variables. Therefore, it is a widely accepted model for analysing failures (Liao, Zhao, & Guo, 2006). The technique calculates the hazard ratio and assumes that the hazard rate ratio between any two individuals remains constant throughout time. In other words, the relative risk associated with a certain factor remains constant during the research. The hazard ratio describes how the risk of encountering a failure event increases when the predictor variables vary. However, the model assumes that the relationship between the log hazard and each covariate is linear, which is a significant limitation in most applications. Finally, ensemble methods, such as random survival forests and gradient-boosting models yield flexible predictors which are known to remain stable in real applications (Hothorn, Bühlmann, Dudoit, Molinaro, & Van Der Laan, 2006; Wang & Li, 2017). Moreover, these models are easy to interpret and do not require elaborate statistical background (Kundu, Darpe, & Kulkarni, 2020). However, these methods require decisions on several main parameters as defined by Frisk et al. (2014), which require some background knowledge. Luckily, they have an inbuilt best health indicator selection capability, which reduces complexity. Moreover, these ensemble methods generate more accurate predictions than ANNs (Wu et al., 2017). Furthermore, ensemble methods are non-parametric, and therefore, do not rely on any assumptions like the proportional hazards model.

Due to crucial data characteristics such as noisy, high-dimensional, aggregated and tabular data and right-censored observations, most methods are unsuitable. Furthermore, the logic underlying the aforementioned prediction methods is difficult to explain. Moreover, we cannot assume that the relationship between the log hazard and each covariate is linear, ruling out the application of Cox's proportional hazards model. However, ensemble methods for survival analysis can deal with these challenges (Hothorn et al., 2006). For a detailed review of survival ensembles, we refer to Hothorn et al. (2006). For a comprehensive review of data-driven

techniques in general, we refer to Ferreira & Gonçalves (2022). They also define a framework for the RUL prediction process, which is shown in Figure 2.2. The first step, data extraction, consists of raw data extraction and pre-processing. Thereafter, the relevant features are extracted from the pre-processed data and classified on their sensitivity to detecting the degradation path. The next step, model building and training, deals with defining the type of model to use to predict RUL and training the model on the training dataset. The RUL prediction is established and evaluated based on predefined performance metrics concluding the remaining useful life prediction process (Ferreira & Gonçalves, 2022).

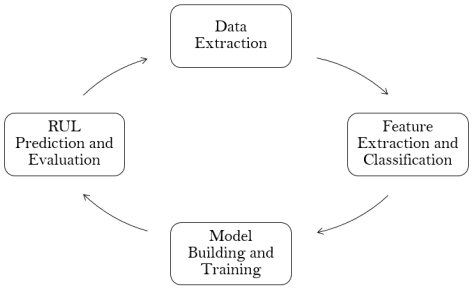


Figure 2.2: Data-driven Remaining Useful Life prediction process

2.2.2 Feature selection

An important step of this RUL prediction process is feature selection (Ferreira & Gonçalves, 2022). Features are the independent variables, so the characteristics of the component are used to predict the remaining useful life. Feature selection methods do not alter the original data, but solely select a subset of these data features (Saeys, Inza, & Larranaga, 2007). The main objectives of such methods are to (1) prevent overfitting and improve model performance (i.e. prediction performance), (2) produce faster and more affordable models, and (3) gain a deeper understanding of the underlying mechanisms that produced the data (Saeys et al., 2007). The modelling procedure becomes more complicated by choosing the pertinent characteristics, but by including the classifier's bias in the search, it becomes possible to build more accurate classifiers. This increased accuracy is achieved by deleting features that contain redundant information (Buchaiah & Shakya, 2022). For example, Prytz et al. (2015) test two feature selection methods: a wrapper approach based on the beam search algorithm and a new filter method based on the Kolmogorov-Smirnov test. On the other hand, Frisk et al. (2014) present two approaches: the first is based on the receiver operating characteristics curve (ROC-curve) whereas the second is a multivariate analysis based on the error rate. Finally, Breiman (2001) used permutation-based feature importance techniques to identify the impact of each feature on the prediction output. This technique permutes the data and evaluates the model performance. In other words, noise is added to the data and compared to the true value to inspect the effect of permuting the variable on the prediction accuracy (Breiman, 2001). Therefore, permutation-based feature importance allows the understanding of the interaction of variables that provide predictive accuracy.

2.2.3 Data balancing

A common issue with survival analysis is class imbalance. This problem occurs when one class in the data (for example uncensored observations) represents a circumscribed concept

(for example failures), while the other class represents its counterpart (for example censored observations) so that examples from the counterpart heavily outnumber examples from the positive class (Batista, Prati, & Monard, 2004). A practical example is that of continuous fault-monitoring tasks where non-faulty examples heavily outnumber faulty examples, where the faulty components are of interest. This class dominance escalates the probability that the majority class will be the dominant class of the model's leaf nodes. Such a controlling class dominates the model predictions. For instance, a 99% accuracy is easily achieved by predicting all instances as the majority class if the remaining 1% should be classified as the minority class. Therefore, data balancing methods significantly impact the predictive power of a model.

Due to this impact, selecting the appropriate data balancing approach is essential. Several data balancing approaches exist (e.g. under-, oversampling and synthetic sampling). Under-sampling reduces the number of samples in the majority class until it equals the number of samples in the minority class. Therefore, this technique reduces the size of the dataset (Batista et al., 2004). Over-sampling, on the other hand, increases the minority size to the proportions of the majority class by replicating instances of the minority class (Afrin, Illangovan, Srivatsa, & Bukkapatnam, 2018). However, this method leads to overfitting. In other words, the model will be trained too closely to the training dataset, performing worse in different instances. The latter approach, synthetic sampling, introduces artificial minority class instances until the minority class achieves the size of the majority class. Thus, several data balancing methods exist, but the choice is highly dependent on data characteristics such as the risk of overfitting and the size of the dataset.

2.2.4 Performance measures for RUL prediction

Performance metrics must be specified to identify the best-performing RUL prediction algorithm. In this section, we present five performance measures applicable to right-censored data and easily interpretable: Harrell's concordance index (C-index), the mean squared error (MSE), the root mean squared error (RMSE), the mean absolute percentage error (MAPE) and the Mean Absolute Error (MAE).

Harrell's concordance index

The C-index is perhaps the most popular performance indicator in survival analysis (Afrin et al., 2018). The C-index consists of two main elements: permissible pairs and their concordance values. Permissible pairs refer to the pairs of samples in the dataset for which a meaningful comparison can be made in terms of remaining useful life and therefore their order of expected failure. In survival analysis, this boils down to two possibilities: (1) if both samples in the pair experienced the event of interest (e.g., failure), or (2) if neither sample experienced the event by the conclusion of the research (e.g., censored observations). For these cases, the concordance values are 1, which are then summed and divided over all possible pairs in the prediction. This index can be expressed as the ratio of the sum of concordance values and the total number of permissible pairs. In other words, the index measures the extent to which the model correctly predicts the order in which the components will fail:

$$C = \frac{\sum_{i,j \in \beta} \mathscr{I}}{|\beta|} \tag{2.3}$$

Where β denotes the set of permissible pairs, and $\mathscr{I}$ is the concordance value per pair, which equals 1 if the order is predicted correctly. As C represents the classification probability of the model, a higher value is desirable. A concordance index of 0 is useless, and a value of 50 % is essentially no better than random guessing (Afrin et al., 2018; Wang & Li, 2017).

Mean Squared Error

The Mean Squared Error (MSE) is another popular evaluation measure. The MSE quantifies the average squared difference between the predicted values (F_i) and the actual values (A_i) in the dataset (Nahmias & Olsen, 2015):

$$MSE = \sum_{i=1}^{n} \frac{(A_i - F_i)^2}{n} \tag{2.4}$$

The MSE is a method for calculating the average size of a regression model's prediction errors. The squaring operation accentuates greater faults.

Root Mean Squared Error

The Mean Squared Error is a popular evaluation measure. However, its root, the root mean squared error (RMSE) is more straightforward as it denotes the average deviation between the actual value (A_i) and the forecasted value (F_i) in the same unit of measurement as the input (Nahmias & Olsen, 2015):

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(A_i - F_i)^2}{n}} \tag{2.5}$$

Thus, the RMSE focuses on the overall accuracy of the predictions. This metric denotes the average deviation in the same unit as the original input. For example, if the input is in the number of days, then the RMSE depicts the deviation in the same time unit, whereas the MSE displays it as unit^2, complicating interpretation.

Mean Absolute Percentage Error

The Mean Absolute Percentage Error (MAPE) is an indicator which is independent of the magnitude of the values (Nahmias & Olsen, 2015):

$$MAPE = \left[\frac{1}{n} \sum_{i=1}^{n} \frac{|A_i - F_i|}{A_i} \right] \cdot 100\% \tag{2.6}$$

Thus, this metric denotes the average deviation between the predicted and actual value in percentage. It emphasises the relative error and is suitable especially when the scale or magnitude of the predicted value is important.

Mean Absolute Error

Another common forecast accuracy measure is the Mean Absolute Error (MAE) (Nahmias & Olsen, 2015). This measure gives the prediction error based on the absolute value of the difference between the actual value (A_i) and the forecasted value (F_i):

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|A_i - F_i| \tag{2.7}$$

Therefore, this indicator does not discriminate between overestimation or underestimation of the target value.

2.2.5 Model explainability and interpretability

Several model interpretation strategies are available to get a further understanding of prognostic models. First, we define the concepts relevant to these strategies, after which the techniques are highlighted. Logically, there is an evident trade-off between the performance and the ability to produce interpretable and explainable predictions of the model (Linardatos, Papastefanopoulos, & Kotsiantis, 2020). Although these terms are often used interchangeably, Linardatos et al. (2020) note that there is a slight difference between the two concepts. Interpretability is defined as the ability to present the concept or cause of a decision in understandable terms to a human. Therefore, it is mostly connected with the logic behind the model's predictions. However, interpretability alone is insufficient to fully understand the model. For this complete understanding, model explainability is essential. This explainability refers to the internal logic of the machine learning algorithm. Therefore, model explainability should be considered in the model building and training phase of the remaining useful life prediction process of Figure 2.2.

This model interpretability refers to the transparent decision logic of the model, which allows potential insights into the decision-making process (Ferreira & Gonçalves, 2022). These insights are vital for ensuring trust in machine-learning tools (Ribeiro, Singh, & Guestrin, 2016). If trust is not in place, the prediction model is useless. As noted, this trust exists in two forms: (1) trusting an individual RUL prediction sufficiently to take action, and (2) trusting a model to behave in reasonable ways. The former refers to local explanations, and the latter to a global explanation of the overall model (Linardatos et al., 2020). An example to increase the first trust form is partial dependence plots (PDPs). These plots provide insight into the relation between one input feature and the predicted output based on the relation established by the model. More specifically, (Friedman, 2001) proposed PDPs to interpret any black box predictive model by showing how a feature affects the average predicted value. As noted by (Linardatos et al., 2020) PDPs can often greatly assist in interpreting black box models and visualising the interactions between features.

To increase the second trust form, (Ribeiro et al., 2016) introduced a method for Local Interpretable Model-agnostic Explanations (LIME) which has arisen into one of the most popular interpretability methods (Linardatos et al., 2020). As this algorithm is straightforward yet powerful, we highlight this method in more detail here. This algorithm can explain the predictions of a model by approximating it locally with an interpretable model. In other words, a local estimation is made, through which the initial black box model can be interpreted.

Therefore, LIME can interpret any type of model. Another method is Shapley Additive explanations (SHAP) (Linardatos et al., 2020). This method aims at enhancing interpretability by calculating the relevance values for each characteristic for individual prediction. However, it is noted to over-weigh unlikely data points. All in all, two methods, LIME and SHAP are well-known to address the problem of model interpretability to increase users' trust in the prediction model.

2.3 Research gap and conclusion

Due to the increased use of data-driven models, a significant body of scientific research exists. However, applying these advanced RUL prediction methods remains challenging for the industry (Javed et al., 2012) as labelled failure data is often scarce or most of the data is censored (Ferreira & Gonçalves, 2022). Moreover, few case studies are available that illustrate the application of prognostic models to real-world problems in realistic operating environments (Sikorska, Hodkiewicz, & Ma, 2011). Yet, survival ensembles, in particular, are extremely important for the industry due to their capacity to cope with right-censored, time-to-event data and explainability. Despite its considerable practical importance, survival ensembles have seen minimal use in machine reliability. For example, Frisk et al. (2014) apply a random survival forest to lead-acid batteries. Other applications mainly exist in the biostatistics field. For instance, Hothorn et al. (2006) introduce a random forest and gradient-boosting algorithm for right-censored survival data for predicting the survival time of patients suffering from leukaemia. Additionally, Afrin et al. (2018) propose a balanced random survival forest for extremely unbalanced, right-censored data for acute cardiac patients. Thus, this method has been widely researched in bioinformatics, but despite its relevance, the application to the reliability of machine components is lacking. Moreover, none of the works listed above, consider model explainability and interpretability in their research. In the words of Sikorska et al. (2011), considerably more research is needed to confirm that prognostic models are beneficial for everyday asset management decision-making. According to Chen, Huang, Chen, Mao, & Li (2023), this lack is caused by two core challenges in RUL prediction tasks. The industrial application of a predictive strategy is challenging. After implementation, the interpretability of the strategies causes added reluctance. Exploring this interpretability of prediction models is beneficial for engineers to assure decision-making based on the prediction. Therefore, the main challenges of this study are two-fold: (1) developing a data-driven RUL prediction model that can deal with right-censored, tabular data and (2) that is explainable and interpretable. Thus, this research will add to this gap by applying the RUL prediction process in a practical context and investigating its explainability.

In conclusion, Remaining Useful Life prediction is an active area of research with ongoing efforts to improve the accuracy, applicability and reliability of the models used. Three groups of methods are defined: model-based, data-driven and hybrid RUL prediction techniques. Data-driven models are researched extensively due to their practical application and promising results. Moreover, as the amount of data recorded increases, machine-learning models will likely play an even larger role in the future. The RUL prediction methods are assessed on their prediction accuracy and interpretability to select the most appropriate methods for DPD. Therefore, this thesis will add to the research of practical applications of data-driven RUL prediction models.

3. Data preparation

This chapter elaborates on the data that was used in this research. This chapter consists of three sections. Section 3.1 describes the data that was acquired for the study, answering Research Question 1.1. Next, Section 3.2 describes which features are extracted from the raw data and how the data is prepared for analysis, after which Section 3.3 describes the cleaning process, and the final input for the modelling phase.

3.1 Data extraction

The first step, data extraction, involves acquiring data about the sorting installation (Ferreira & Gonçalves, 2022). The required data was established from three datasets with different contents: components, packages and alarms. These three datasets can be characterized as follows:

1. **Components**: contains all components (1839 Carts, 1940 MCB units, 1931 Motors and 1943 Crossbelts) installed in the sortation system between 03/03/2019 and 01/04/2023, their replacement dates and a short text description of the maintenance action.
   The first dataset, extracted from the supplier's information system, stores the maintenance reports. Each row contained an item replacement for a location in the system from which the replacement intervals per system location were determined. Based on these intervals, the lifetime per component in that specific location could be derived, such that each part received a unique ID.

2. **Parcels**: contains all 166 million parcels shipped by DPD between 11/02/2020 and 01/04/2023 and their shipment date, weight, volume, time on the sorter, number of recirculations, carrier count and corresponding induct, chute and component.
   The second dataset was obtained from DPD's data management system, which collects information about the sortation system and its usage. Different sensor measurements are included in the dataset that measured the characteristics of each parcel and its location. The inducts indicate whether the parcel entered the components from the left or right, whereas the chutes show whether the parcel exited the component to the left or right. The number of recirculations of a shipment denotes the number of times a parcel has travelled past the entire installation, which happens when the package cannot be discharged from the system. A failed discharge can be due to operational causes such as a blocked chute. The carrier count of a parcel denotes if the item was shipped on one or two carriers. The item is placed onto two carriers if the length exceeds 55 centimetres to avoid collisions between packages on adjacent components.

3. **Alarms**: contains the 45713 alarm notifications of the system from 17/03/2020 to 01/04/2023, the type and duration of the alarm, and the component that caused the warning.
   The third dataset was extracted from DPD's data management system, which collects information about the sortation system and its alarms. DPD distinguishes three types of alarms for their sortation system: Item overhanging belt (IOB), Deflected bellow fault (DB), and item retract. An IOB notification denotes that an item is hanging over the side of the component, which can be due to a failure when the parcel enters the system or package instability causing it to move during operation. These alarm sensors were installed onto the system on 17/03/2020. A deflected bellow fault occurs when

such an unstable package moves to the front or back of the component and lands on the bellow between two components. Such an incident significantly increases the chance of a crash. Finally, an item retract alarm denotes that the parcel has shifted from its original location, but the shift was detected in time for the system to correct. The item retract fault sensors and deflected bellow sensors were installed in the system on 09/07/2021.

The following section, Section 3.2, describes the approach for extracting relevant features from these datasets.

3.2 Feature extraction

Ferreira & Gonçalves (2022) define feature extraction as "transforming raw data collected from the running system into relevant information about the running status of that system". These initial parameters are established during interviews with system experts, as the system is highly complex. The technical maintenance manager and technical contract manager of DPD and the maintenance manager and a project manager of the system's supplier shared their insights on the components' degradation. Based on these insights, 44 features were identified that might be relevant to the RUL prediction. These features were aggregated to the total, mean, and maximum of the following features: parcel volume, weight, density, time on sorter and load (e.g. the time during which the parcel's weight was putting stress on the component in $kg \cdot sec$). In addition, the alarm count per type and the parcel count were listed per component (where the total equals the observed count and the extrapolated observations) as well as the count that the crossbelt unit turned clockwise and counterclockwise. Finally, the utilisation, the fraction of time that a component was actively used to transport parcels, is calculated per component. The data is aggregated in this manner to ensure that only relevant information for the system degradation is included. Moreover, this aggregation limits the size of the dataset. Time series data instead of aggregated data would be an interesting extension. However, the inclusion of such measures would increase the dataset with 166 million observations per parcel feature. In addition, Frisk et al. (2014) reported adequate results using similar aggregated data. Therefore, this study is limited to these aggregated measures.

3.3 Data pre-processing

After the correct data has been gathered, it should be prepared for analysis. Ferreira & Gonçalves (2022) defined data pre-processing as exploiting features associated with the degradation of the system, which are hidden in the raw data. In the context of this thesis, data pre-processing concerns filling in missing values and observations, and associating the data on the packages and alarms with the components to extract relevant information on the parts and their usage.

3.3.1 Missing values

The first step of data pre-processing is to handle missing values. For the component dataset, missing values were present regarding the replacement reason for a component. This label is

necessary as the prediction of wear and tear of components is the focus of this thesis. However, items can also be preventively replaced, in operation or replaced due to damage caused by a crash. These components still inform us about the degradation of the system, but the outcome is unknown. Therefore, these observations are used for analysis, but regarded as censored observations. These missing labels were complemented based on the maintenance reports. The parcel dataset contained missing values for the weight, volume and time on the sorter of that parcel which were rectified with the mean of these features. In addition, the incoming and departure directions are deducted from the induct and chute labels, respectively. Based on the system layout, each induct and chute has a set direction to enter or leave the system. Therefore, based on our understanding of the sorting machine, we can register whether the crossbelt turned clockwise or counterclockwise.

In addition to missing values, we also encountered missing data, as can be derived from the start dates per dataset, as presented in Section 3.1: the parcel and alarm data are partially unavailable. For the parcel data, observations between 03/03/2019 and 11/02/2020 are missing. For the alarm data, the missing observations differ per alarm type as the installation date of the sensors differs. The installation date is 17/03/2020 for the IOB alarms and 09/07/2021 for the DBS and retract alarms. To correct these values, per component, the number of alarm notifications and the number of packages were split into four periods, as visualised in Figure 3.1. In interval (4), all parcel information and alarm information is known. In period (3), the DBS and retract alarm notifications are unknown, but the parcel volumes and IOB alarm notifications are clear. In time frame (2), we only obtained the parcel information. Finally, in interval (1), only estimates of the parcel information are known, as the parcel count per day has been registered. Therefore, these missing values can be estimated by computing the average number of parcels per component based on these volumes and the number of days that the part is missing parcel information. For example, a component that was operational between 03/03/2019 and 01/04/2020 misses every parcel volume between 03/03/2019 and 11/02/2020 (period 1) and is corrected with the average for each of these days, whereas a component that was only operational between 03/03/2019 and 20/11/2019 would only be corrected for this period and not the entire interval between 03/03/2019 and 11/02/2020. In short, the missing data were extrapolated from the available information. So, the IOB alarm notifications for period (1) were extrapolated based on IOB notifications and parcels registered during periods (2), (3) and (4), whereas deflected bellow supervision and retract warnings for periods (1) and (2) were extrapolated based on interval (3) and (4).

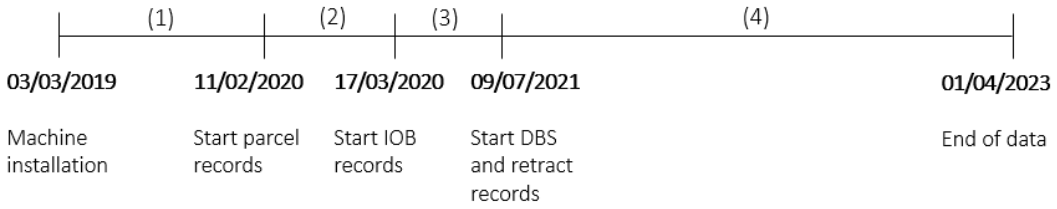


Figure 3.1: Timeline of data records

3.3.2 Data cleaning

Data cleaning corrects or removes incorrect or incomplete data within a dataset. In this case, this mainly consisted of correcting noisy data and identifying erroneous values. First,

components were identified that failed due to wear but were hardly used. When inspecting the maintenance reports of these components, mistakes in component failure times were identified. For example, no failure was registered in the maintenance reports, but a failure was logged in the system. Thus, the component lifetimes were corrected. Second, incorrect entries were found for the weight of parcels, as some shipments received a negative weight value. As this is impossible, the value was set to 0 as the weight will likely have been negligible. Overall, the other entries are assumed to be correct.

3.3.3 Data integration

Data integration refers to the process of combining data from different sources into a coherent view. Challenges arise due to variations in data purpose and formats. In this case, the supplier data on the components (dataset 1) should be combined with DPD's data on their shipments and alarm notifications (dataset 2 and 3). This process creates extra challenges as the data is gathered in different systems. Therefore, data and business understanding are essential. As mentioned before, based on the replacement intervals per system location, the lifetime of each component was determined based on replacement intervals specific to its system location. The component's labelling is determined by its position within the system and the sequential order of its installation. By analyzing the installation and removal dates, the useful life of each component was calculated. This information facilitated the matching of parcel and alarm data to the components, where the date and component details of each occurrence were utilized to identify the correct unique identifier associated with the component in question. The matching procedure was performed using SQL to deal with the large amount of data.

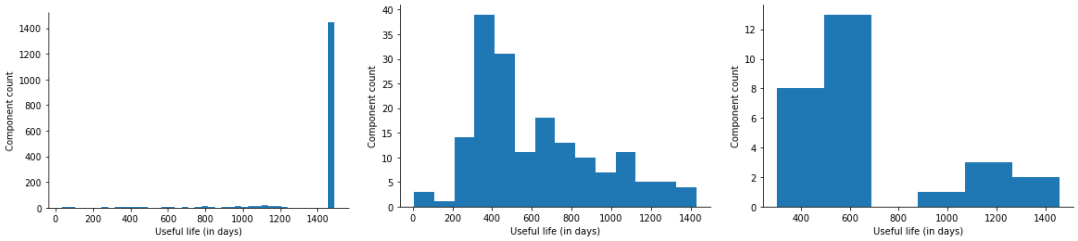
After data cleaning and integration, we obtain a dataset consisting of 1842, 1940, 1931 and 1943 components for the cart, MCB, motor and crossbelt, respectively. A significant amount of data is censored (i.e. still operational or crashed), 90.5%, 86.4%, 86.8% and 86.3% for these components, respectively. Table 3.1 summarizes these data descriptions. Please note that the system consists of 1640 operational components at any point in time, as it is a serial system, and therefore 1640 censored components. The censored observations are included in the study since their values tell us how long they will last relative to failed components. Finally, there is no apparent trend in the frequency of crashes, which makes sense given that they are caused by unstable parcels on the component, which is fully independent of lifespan.

Component		Degraded components	Crashed components	Functioning components	Total number of observations	Percentage censored
Cart		172	27	1640	1839	90.50%
	MCB	264	36	1640	1940	89.39%
Carrier	Motor	255	36	1640	1931	86.79%
	Crossbelt	267	36	1640	1943	86.26%

Table 3.1: Description of the number of units per label per subcomponent

3.3.4 Data exploration

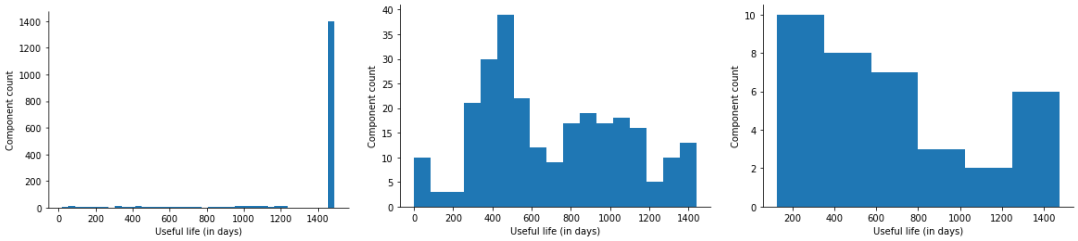
To gain insight into the degradation of these components, we perform exploratory data analysis. First, we inspect the average lifetime of a component per replacement trigger (censored, failed or crashed). Figures 3.2, 3.3, 3.4 and 3.5 present the histograms per category for the carts, MCB units, motors and crossbelts, respectively. Please note that the scale of the vertical axis differs per censoring category to ensure the readability of the graphs. We can see that each component has a clear peak for components with a lifetime of 1490 days, which



(a) Censored (operational) **(b)** Failed due to regular wear **(c)** Crashed

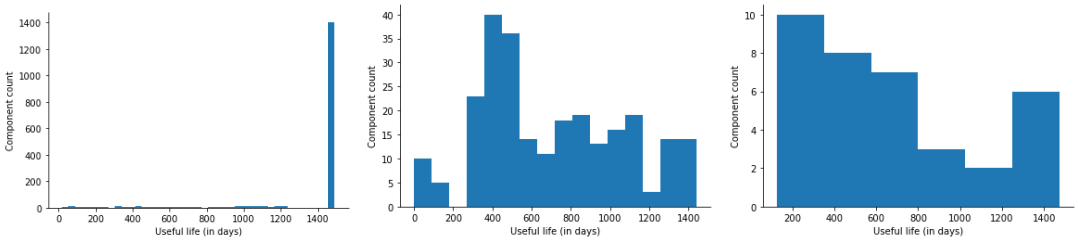
Figure 3.2: Histogram of carts per replacement label (operational, failed, crashed)

is the time since the machine was installed. Therefore, these are the components that have been installed at the beginning and have not failed yet. Please note that the censored values with a useful life lower than 1490 are due to the component being replaced recently, so the component has been installed a few weeks ago but has not failed yet. Moreover, we observe that the histograms of crashed MCB units, motor and crossbelt look reasonably similar, this is because these components are part of the carrier, and if a carrier crashes, the entire component is replaced. In addition, the cart, MCB, motor and crossbelt each have a peak of about 40 components that failed around 400, 480, 440 and 500 days, respectively. After that, there seems to be a moderately decreasing trend in the lifetime.



(a) Censored (operational) **(b)** Failed due to regular wear **(c)** Crashed

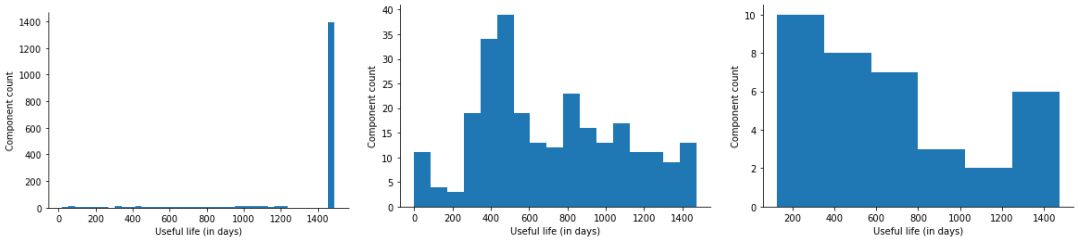
Figure 3.3: Histogram of MCB units per replacement label (operational, failed, crashed)



(a) Censored (operational) **(b)** Failed due to regular wear **(c)** Crashed

Figure 3.4: Histogram of motors per replacement label (operational, failed, crashed)

The average useful life (i.e. the average number of days that the component was functional in the system) of a cart, MCB, motor and crossbelt are 1328, 1269, 1273 and 1266 days, respectively. From all components, 172, 264, 255 and 267 carts, MCB units, motor and crossbelt have failed due to regular degradation. Their mean time to failure of the failed components is 615, 700, 702 and 705 days, respectively. Finally, we note that the standard deviation is substantial, namely 308, 364, 369 and 373 days for the cart, MCB, motor and crossbelt, respectively.



(a) Censored (operational) **(b)** Failed due to regular wear **(c)** Crashed

Figure 3.5: Histogram of crossbelts per replacement label (operational, failed, crashed)

	Mean lifetime (days)	Number of failed components	Mean time to failure (days)	Standard deviation in time to failure
Cart	1328	172	615	308
MCB	1269	264	700	364
Motor	1273	255	702	369
Crossbelt	1266	267	705	373

Table 3.2: Lifetime descriptives

Overall, this chapter detailed the data utilized in this study, as well as the process by which it was cleaned and integrated. Furthermore, a first look at the data was offered. The methodology of this thesis, for which this data was prepared, is described in the next chapter, Chapter 4.

4. Methodology

This chapter presents the methodology of this thesis. First, Section 4.1 highlights the prerequisite of model explainability and how this relates to model building. Then, Section 4.2 discusses the model-building phase of the remaining useful life prediction process, which answers Research Question 1.3. Section 4.3 explains the evaluation measures which are used in this study to evaluate the performance of the prediction models, providing the answer to Research Question 1.4.

4.1 Model explainability

As noted in Chapter 1, DPD requests for the model to be both explainable and interpretable for communication purposes. Model explainability is inherent to the type of model used (Linardatos et al., 2020). Therefore, this should be discussed before model selection and building. As defined in Chapter 2, model explainability deals with the internal logic of the machine learning model. In other words, it deals with the clarity of the inner workings of the prediction model. As noted, Neural Networks and other deep learning methods are often difficult to explain. However, decision tree-based models easily produce explainable results with common examples (Linardatos et al., 2020). Therefore, Chapter 4.2 deals with the decision tree-based methods used in this research.

4.2 Model building

This section highlights the chosen RUL prediction methods and their characteristics. As noted in Chapter 2, survival ensemble methods can deal with the challenges posed by the real-world dataset, such as right-censored tabular data. These approaches yield flexible predictors and remain stable in high-dimensional settings (Hothorn et al., 2006). Therefore, three methods are selected for comparison: the survival regression tree (ST), random survival forest (RSF) and gradient-boosting (GB) model. The former method is chosen to provide insight into the benefits of the latter two as Wang & Li (2017) reported that these methods have high precision and are highly flexible due to their non-parametric nature. Therefore, these approaches alleviate the problematic assumptions of the proportional hazards model. These characteristics ensure the high practical relevance of RSF and GB models. Moreover, these models are based on regression instead of time-series, which is common for RUL prediction. As noted in Chapter 3, the dataset contains information on 166 million parcels. Therefore, time-series data for numerous features per parcel would rapidly increase the data volumes, which poses significant challenges for dealing with these data. Thus, we choose to use aggregate data instead of time-series data, which is enabled by survival regression trees, random survival forests and gradient-boosting models.

Both gradient-boosting models and random survival forests are tree-based ensemble methods, yet they differ as to how the regression trees are combined. First, Subsection 4.2.1 introduces survival regression trees to ensure the required base comprehension. Subsections 4.2.2 and 4.2.3 describe the two methods of combination for the random survival forest and gradient-boosting models, respectively.

4.2.1 Survival regression trees

Each regression tree has three main parts: a root node, leaf nodes, and decision nodes. Root nodes are split into decision nodes, and decision nodes are branched as leaf nodes. The leaf nodes are not split further and thus are the end nodes, as visualised in Figure 4.1. Therefore, these leaf nodes contain clusters of components with similar usage features. In each leaf node, an estimator is fitted to predict the target variable. In the case of survival regression trees, the estimator is tailored to censored data. In other words, Formula 2.2 is estimated per cluster/leaf node. Therefore, Frisk et al. (2014) adapted Formula 2.2 to consider a usage profile per cluster $\mathcal{V}$, where the expected remaining useful life for that usage profile is:

$$E[RUL(t_0, \mathcal{V})] = \frac{1}{R^{\mathcal{V}}(t_0)} \int_{t_0}^{\infty} R^{\mathcal{V}(t)} dt - t_0 \tag{4.1}$$

Thus, regression trees are for dependent variables that take continuous ordered values, with the prediction error measured by the difference between the observed and predicted values (Loh, 2011).

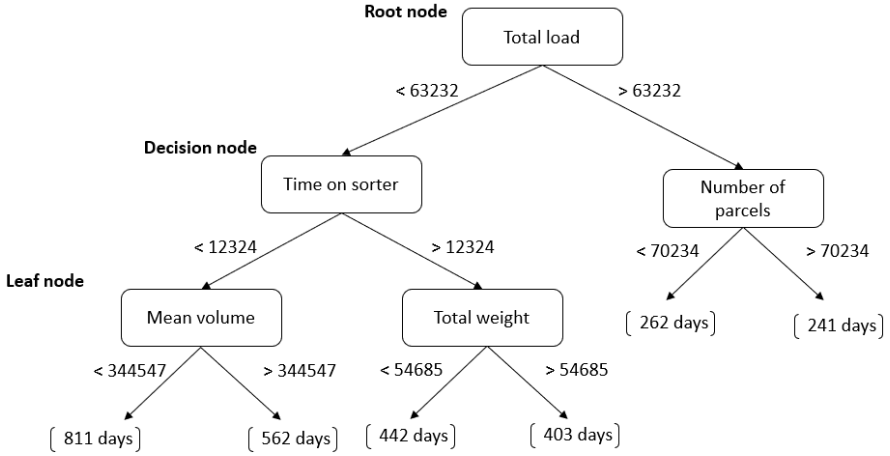


Figure 4.1: An example regression tree

The algorithm to construct a regression tree is defined by (Loh, 2011), where X is the set of features and Y is the remaining useful life:

1. Start at the root node
2. For each X, find the set S that minimizes the sum of the squared deviations about the mean and the node predicting the sample mean of Y in the two child nodes and choose the split that gives the overall minimum overall X and S.
3. If a stopping criterion is reached, exit. Otherwise, apply step 2 to each child node in turn.

Multiple of these survival regression trees are combined into a prediction forest in gradient-boosting models and random survival forests. Random forests average the prediction outcome per regression tree, whereas boosting applies a functional gradient descent algorithm minimizing residuals (Hothorn et al., 2006). These trees first cluster the observations and then

estimate the reliability per cluster. This increases their applicability in this context as the components are first clustered based on similar usage, and a reliability estimate is made based on this similarity. Sections 4.2.2 and 4.2.3 introduce the random survival forest and the gradient boosting model, respectively.

4.2.2 Random survival forest

Ishwaran, Kogalur, Blackstone, & Lauer (2008) introduced random survival forests by extending the random forests method for the analysis of right-censored survival data. Survival analysis is used to analyse the time-to-event, where an event can be a failure or censored observation. A random survival forest is an ensemble method that combines regression trees adapted for survival analysis. Several regression trees (n) are grown that each predict remaining useful life. The final prediction is the average of all these individual estimations. The random survival forest network is shown in Figure 4.2. The random survival forest incorporates randomization in two forms (Ishwaran et al., 2008). First, a random bootstrap sample is randomly drawn to grow a tree. Second, at each node of the tree, a subset of variables are randomly selected as candidate variables for splitting. This allows the forest to approximate rich classes of functions while sustaining a low generalization error (Ishwaran et al., 2008). The random survival forest algorithm consists of the following steps (Wang & Li, 2017):

1. Drawing **L** bootstrap samples from a training dataset of size **n**. The remaining out-of-bag (OOB) observations will not appear in the bootstrap sample.

2. For each bootstrap sample, grow a full-size survival tree based on the log-rank splitting rule. At each internal node, randomly select a subset of candidate covariates out of all covariates.

3. For each tree, the survival probability is estimated. At each terminal node **k** at the time point **t**, the reliability is estimated by the Kaplan-Meier estimator (Ebeling, 2010):

$$\hat{R}_i(t) = \prod_{\{j:t_j \leq t\}} \left(1 - \frac{1}{n_j}\right) \tag{4.2}$$

   where t_j are the ordered failure times and n_j are the number remaining at risk just before the jth failure.

4. To predict the survival function of a new observation **x**, the average over all survival functions from all the **L** trees is used to obtain the ensemble function of the forest:

$$\hat{R}_E(t|x) = \frac{1}{L} \sum_{i=1}^{L} \hat{R}_i(t|x) \tag{4.3}$$

   where $\hat{R}_i(t|x)$ denotes the reliability of the tree grown from the i-th bootstrap sample.

Note that the Kaplan-Meier estimator is tailored to censored data. This estimator adapts the survival function to the survival probabilities at the moment of censoring. Censored observations are included in the estimator until the moment of censoring. Thereafter, it is not considered in the risk calculation for an event.

Next to the survival function, the algorithm also produces an RUL estimation based on the average of the RUL prediction from all trees. It builds on the idea of bagging, but it provides an improvement because it de-correlates the trees by adding a random subset of features (m) to be considered, which usually is determined by $m \approx \sqrt{p}$.

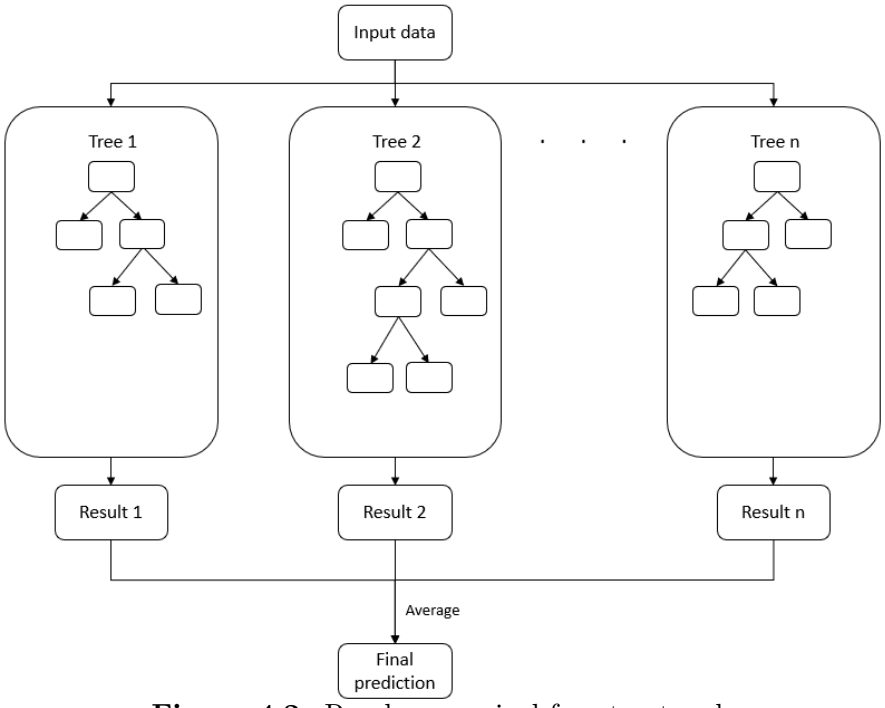


Figure 4.2: Random survival forest network

Model parameters

A random survival forest requires several decisions on parameters. The experiments in this study are conducted using the Python library Sci-kit Survival (Pölsterl, 2020). The combinations of parameters tested in this study are highlighted in Section 5.4. The main six parameters to be chosen in the software are:

- number of trees to grow in the forest
- minimum number of observations per split
- maximum depth of each tree
- maximum number of leaf nodes
- minimum number of observations per leaf node
- maximum number of features to consider when looking for the best split

The choice of these parameters has a considerable influence on the model's performance. The selection of these parameters is explained further in Section 5.4.

All in all, a random forest builds a predefined number of trees based on a random subset of features and averages their predictions to obtain a final prediction. This prediction is at least comparable to the state-of-the-art machine learning methods (Ishwaran et al., 2008). Additionally, random survival forests inherit the robustness and desirable properties of ensemble methods, such as increased accuracy and limited bias and variance.

4.2.3 Gradient-boosting model

Friedman (2001) introduced a general gradient-boosting machine. This article concludes that gradient-boosting of regression trees produces highly robust, interpretable procedures especially noisy data. Hothorn et al. (2006) extended this algorithm for right-censored data. Several decision trees are grown based on gradient descent optimization to iteratively fit new models to the residuals of the previous trees. These residuals ensure that the observations that had the highest prediction error received the most attention in the next regression tree. The predictions are combined in an additive manner, where the addition of each base model improves (or "boosts") the overall model. The gradient-boosting algorithm is as follows (Friedman, 2001):

1. Initialize the model with the median of the target value.
2. For each iteration: calculate the residuals between the predicted values and the true values concerning the current model's predictions.
3. Fit a decision tree to the negative gradient by minimizing this loss function.
4. Update the model by adding a weak learner, multiplying it by the learning rate, and adjusting the predictions.
5. To predict the survival function, similar to step 4 in the random survival forest algorithm, the average of all predictions is used.

As can be concluded from this algorithm, gradient-boosting applies a greedy approach to the loss function and combines each improvement based on the learning rate to decrease overfitting.

Model parameters

A gradient-boosting model involves numerous parameter choices. The model is built using the Python library Sci-kit Survival (Pölsterl, 2020). The combinations of parameters tested in this study are highlighted in Section 5.4. The main eight parameters to investigate:

- number of trees to grow in the forest
- loss function to be optimized
- the learning rate
- minimum number of observations per split

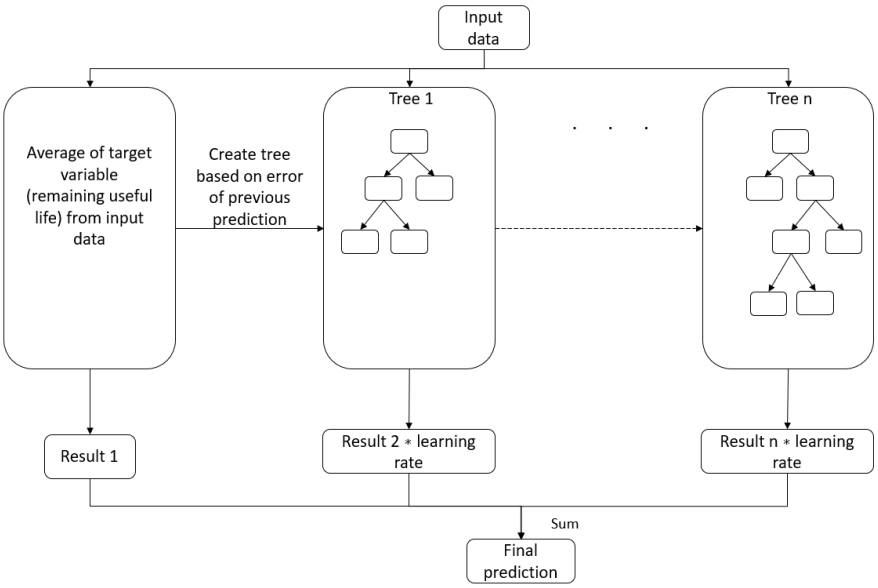


Figure 4.3: Gradient-boosting network

- maximum depth of each tree
- maximum number of leaf nodes
- minimum number of observations per leaf node
- maximum number of features to consider when looking for the best split

The selection of these parameters has a significant impact on the model's performance. Section 5.4 goes into further detail on how these criteria were chosen.

Overall, gradient-boosted models are known to be flexible predictors that remain stable in high-dimensional settings. However, it is important to note that overfitting is a significant risk for gradient-boosted models. Similar to a random survival forest, these models inherit the desirable properties of ensemble methods by combining multiple regression trees.

4.3 Evaluation measures

To evaluate the performance of the survival tree, random survival forest and gradient-boosting model, the general measures for data-driven models were discussed with DPD to select the most applicable indicators which are also relevant for their interpretation of the performance. Hence, we select four evaluation measures. First, to evaluate the basic ordering of events, Harrell's concordance index is chosen as this metric is tailored to problems containing censored observations. Second, the RMSE is chosen to evaluate the deviation between the prediction and the actual value, as this measure is more straightforward to interpret than the MSE. Third, the mean absolute percentage error is used to evaluate the performance independent of the magnitude of the values. Fourth, DPD emphasized the trade-off between underestimation and overestimation of the remaining useful life of a component. The overestimation of the remaining useful life of a component has a significant impact on the downtime of the system.

Therefore, DPD prefers underestimation over overestimation. However, the mean absolute error does not consider this trade-off. Therefore, we separately report the overestimation and underestimation. Finally, we list the training times of the model to gain insight into the speed of the model and the time consumption of the training phase.

All in all, we train survival regression trees and two different survival ensemble methods: a random survival forest and a gradient-boosted model. The random forest seeks to minimize the empirical risk indirectly via the stabilization of randomized base learners. In contrast, gradient boosting employs a functional gradient descent algorithm for minimizing the empirical risk (Hothorn et al., 2006). These models are evaluated based on several measures to pick the most suitable model for DPD.

5. Modelling

This chapter presents the modelling phase including feature selection, defining dependent and independent variables, data balancing and model training and testing. Section 5.1 presents the feature selection. In other words, this section provides the answer to Research Question 1.2 by discussing the relevant usage indicators. Afterwards, Section 5.2 describes the definition of both dependent and independent variables. Section 5.3 describes the data balancing for the datasets. Finally, Section 5.4 describes the parameters to be optimized in the models.

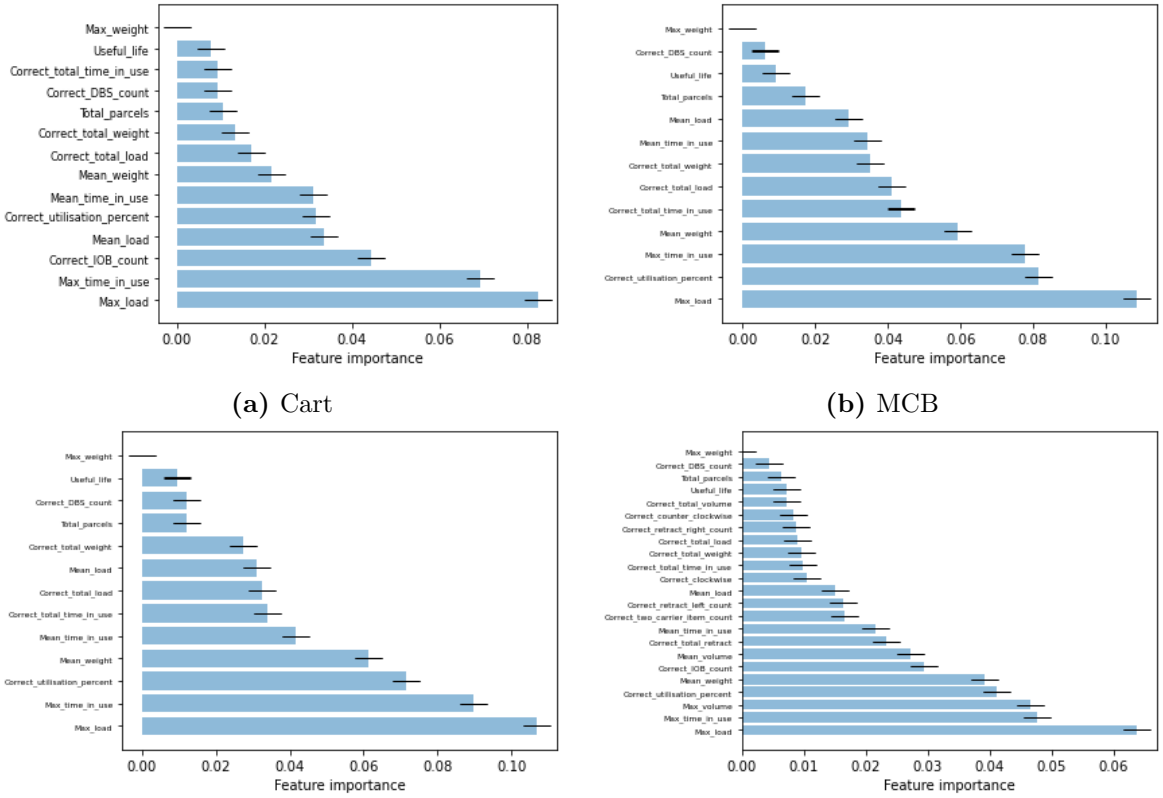
5.1 Feature selection

Feature selection includes determining which extracted features are more sensitive to detecting the degradation path (Ferreira & Gonçalves, 2022). In other words, this process selects the features with the strongest predictive power from the full set as described in Section 3.1, as too many irrelevant features will lead to a poor prediction result of the model (Ma, Chen, Cao, Yao, & Liu, 2020; Prytz, Nowaczyk, Rögnvaldsson, & Byttner, 2015). Moreover, feature selection aids in preventing overfitting and reduces complexity and training times (Saeys et al., 2007; Buchaiah & Shakya, 2022). Therefore, this step consists of selecting the most important features from the features identified together with the system experts.

As noted by (Javed et al., 2012), feature selection and the remaining useful life prediction method are decisions that both impact performance and should thereby be considered simultaneously. Moreover, tree-based algorithms such as the random survival forest and gradient-boosting model, have embedded feature selection strategies (Saeys et al., 2007). Therefore, we fit a random survival forest (150 trees and a minimum of 100 samples per split) and use permutation-based feature importance to inspect the influence of each indicator on the remaining useful life of a component. Permutation-based feature importance is used to assess the impact of each feature on the final prediction. On the dataset, a baseline measure is first examined. The measure is then assessed again after a feature column is permuted. The difference between the baseline metric and the metric from permuting the feature column is defined as the permutation significance (Breiman, 2001). This assessment is performed 15 times to ensure a valid result. Then, we obtain the mean and standard deviation of the feature importance. As noted by Kundu et al. (2020), Prytz et al. (2015) and Frisk et al. (2014) random forests have a built-in health indicator selection capability. Prytz et al. (2015) elaborates that this method outperforms human experts. Hence, we use the random survival forest and permutation-based feature importance to select the most powerful subset. Per component, we retain the 20 most impactful features, and its useful life so far. Figures 5.1a, 5.1b, 5.1c and 5.1d show the mean and standard deviation of the feature importance for the carts, MCB units, motors and crossbelts, respectively. Thereafter, Table 5.1 shows an overview of the features that were selected for each component. The selected features per component are shown with a ✓and the features that were selected by the expert, but excluded after permutation-based feature importance with a $\cdot$. Based on the discussion with system experts and the permutation-based feature importance, we can draw 11 main conclusions:

1. The utilisation is relevant for the degradation of each component,

2. The direction that the crossbelt turned for each parcel is more important than the number of parcels for the crossbelt,



(a) Cart **(b)** MCB

(c) Motor **(d)** Crossbelt

Figure 5.1: Mean and standard deviation of the feature importance per component

3. The total and mean weight is impactful for the degradation of each component,
4. The maximum weight is a feature that adds noise to the model and is excluded after the feature selection for every component,
5. The crossbelt is the only component impacted by the volume of each parcel,
6. The mean, total and maximum time that a component is used to transport a parcel (time in use) is relevant for the degradation of all components,
7. The mean, total and maximum load (time · parcel weight) is relevant for the degradation of all components,
8. The IOB alarms are only relevant for the degradation of the cart and crossbelt,
9. The number of DBS alarms has an impact on the degradation of the cart, MCB and motor, but not on the degradation of the crossbelt,
10. The retract alarm occurrences on the right, left and the total number of occurrences are only relevant for the crossbelt,
11. The number of two carrier items is only impacting the degradation of the crossbelt.

	Cart	MCB	Motor	Crossbelt
Useful life	✓	✓	✓	✓
Component utilisation	✓	✓	✓	✓
Total number of parcels	✓	✓	✓	·
Mean weight	✓	✓	✓	✓
Total weight	✓	✓	✓	✓
Maximum weight	·	·	·	·
Mean volume				✓
Total volume				✓
Maximum volume				✓
Mean time in use	✓	✓	✓	✓
Total time in use	✓	✓	✓	✓
Maximum time in use	✓	✓	✓	✓
Mean load	✓	✓	✓	✓
Total load	✓	✓	✓	✓
Maximum load	✓	✓	✓	✓
Number of IOB alarms	✓			✓
Number of DBS alarms	✓	✓	✓	·
Number of retract alarms				✓
Number of retract alarms right				✓
Number of retract alarms left				✓
Number of two carrier items				✓
Number of rotations clockwise				✓
Number of rotations counter-clockwise				✓

Table 5.1: Overview of selected features per component

This subset of features will serve as input to the model and form the basis for its RUL prediction. The input variables are defined in more detail in the next section.

5.2 Dependent and independent variables

The survival tree, gradient-boosting model and random survival forest predict the remaining useful life (dependent variable) based on the selected features as shown in Table 5.1 (independent variables). This section will further explain this input data.

The independent variables (X) are the features that were selected as the most powerful subset (as shown in Table 5.1). Thus, the model takes 13, 12, 12, and 20 independent variables as inputs for the cart, MCB, motor, and crossbelt, respectively. This subset is provided to the model to base the remaining useful life prediction on. As we aspire to predict the remaining useful life, this is the dependent variable (Y). Therefore, the input data should be in the same format. However, the components listed in Table 3.1 are currently described using their entire lifetime from installation date to a failure or censoring event instead of time-to-event data. Therefore, we split these observations into weekly intervals, with the corresponding useful life and remaining useful life. This interval is suitable as DPD foresees that the prediction will be updated every week when operational. Figure 5.2 visualises the split. In short, the component lifetime (c), from installation date (t_0) until end-of-life (T), is available. However,

the remaining useful life (b) is of interest, which is the difference between the cut-off moment (t) and the end of life (T). As the entire lifetime is known, we can arbitrarily choose our cut-off time (t). We choose to pick multiple values for (t) for each week in the lifetime of the component. For instance, component 1, having a lifetime of 26 weeks (c), can be split into component 1.1 having useful life (a) = 1 week and a remaining useful life (b) = 25 weeks, and component 1.2 having a useful life (a) = 2 weeks and a remaining useful life (b) = 24 weeks. Note that both components still have a life span (c) of 26 weeks etcetera. For each component, it always holds that (c) corresponds with the original lifetime of the components, the only difference is that (t) shifts one week, changing the length of periods (a) and (b). This weekly division also increases the number of observations available, as multiple entries per component become available. Figure 5.3 provides an example of this division into parts. Here, X denotes the independent variables being the usage features of the component, and y equals the dependent variable, being the remaining useful life after the cut-off moment (t).

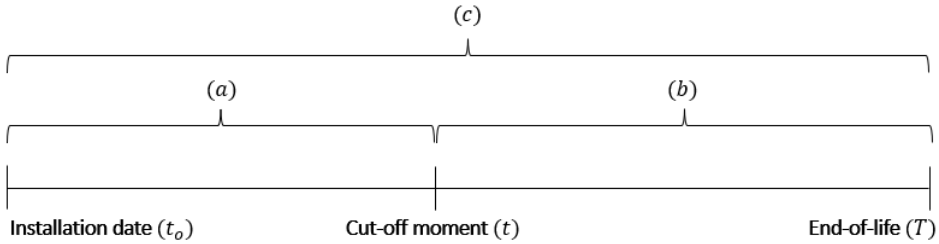


Figure 5.2: Component lifetime (c) split into useful life (a) and remaining useful life (b)

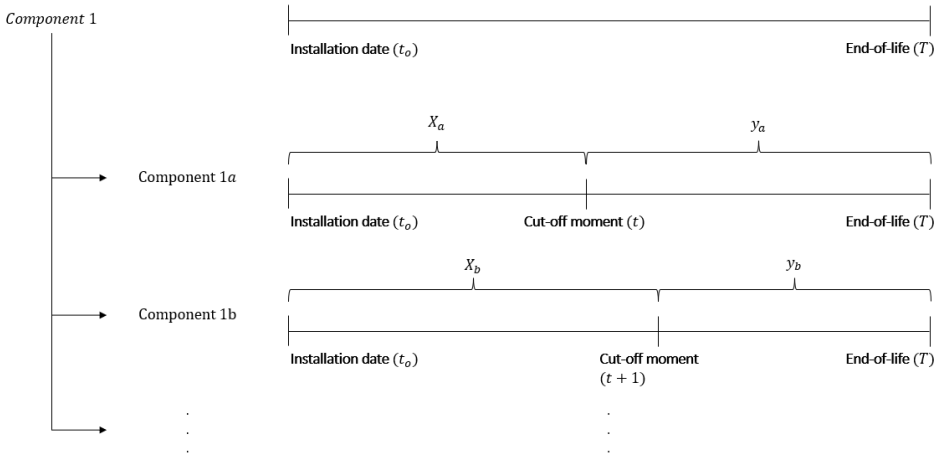


Figure 5.3: Example of how components were split into input format

5.3 Data balancing

As noted in Table 3.1, all four datasets are highly imbalanced. In this case, the censored observations (e.g. crashed components or components that are yet to fail) exceed the observations of components that failed due to regular wear. In this case, we undersample the observations for each censored component. This sampling method does decrease the number of observations, which could lead to an information loss. On the other hand, a decrease in observations also reduces training and testing times, and memory usage. As noted by Batista

et al. (2004) the imbalance problem is a relative problem depending on the concept complexity and overall size of the training set. Due to the high system complexity and large variance in loads, each component contains valuable information on the degradation. However, as we obtained multiple entries per component, overfitting is also a significant risk. Therefore, we undersample per censored component and retain the observations of degraded components to ensure that each component is represented in the dataset. Ultimately, each dataset consists of 50% uncensored observations.

After feature selection and data balancing, the final dataset consists of data on 1640 locations in the sorting chain between 03-03-2019 and 01-04-2023. In total, we obtained 13.610, 27.006, 26.895 and 29.326 RUL intervals for the cart, MCB, motor and crossbelt, respectively, of which 50% is uncensored. For the cart, 13 features are recorded, for the MCB and motor 12 features and for the crossbelt 20 features are included in this study. These measures contain aggregated data on the usage of the components, such as the number of parcels transported, the mean weight of these parcels and the number of alarms triggered by this component.

5.4 Model training and testing

This section presents the model training phase, focusing on parameter optimization, within the context of the thesis. During this phase, a relationship is established between the health state and the selected features (Ferreira & Gonçalves, 2022). The data for each component is divided into training and test sets. Specifically, the data is split into 70% for training and 30% for testing. Consequently, the training set comprises 9.527, 18.904, 18.826, and 20.528 observations for the cart, MCB, motor, and crossbelt, respectively. The remaining 30 % is used for the test set, to evaluate the model's performance. To tune hyperparameters, 20 % of the training set is set aside for validation. The desired output for training includes the remaining useful life in days and a censoring indication (0 = censored, meaning the component is still operational or crashed, and 1 = failure due to wear). The optimization of the models revolves around two parameters: the number of trees and the minimum number of samples per split. The number of trees is varied from 1 to 750, while the minimum number of samples per split is tested at 10, 20, 100, and 200. Another option would be to restrain the maximum depth of the tree instead of the minimum number of samples per split. However, identifying the right clusters is crucial to obtain an appropriate reliability estimate for those observations. Therefore, we restrain the minimum number of samples per split instead of the maximum depth. The optimal combination of the number of trees and minimum number of samples per split is investigated using a grid search where the number of trees varies from 1 to 750 and the minimum number of samples per split is either 10, 20, 100 or 200. Another significant issue is overfitting, especially for the gradient-boosting model. Regularization methods attempt to prevent overfitting by constraining the fitting procedure (Friedman, 2001). Thus, we use a smaller learning rate of 0.3 instead of 1.0, which restricts how fast the model learns the data and limits overfitting. Finally, both ensemble methods are allowed to consider a maximum number of features (m) that is equal to the square root of the total number of features available (p): $m \approx \sqrt{p}$.

As the remaining model parameters discussed in the Subsection 4.2.2 and 4.2.3 are linked, we chose to optimize over the number of trees in the forest and the minimum number of samples per split. As the number of samples per split is linked to the depth of each tree,

we do not iterate over the tree depth as well. Moreover, the number of observations per leaf node is not restricted as the system is highly complex. Finally, Frisk et al. (2014) noted that the evaluation of the remaining useful life (Formula 2.2) requires integration to infinity. Unfortunately, the estimated reliability functions have a high degree of uncertainty for large values of t, especially since we have no observations past $t = 1490$ days. Therefore, the lifetime prediction is used as the time to perform maintenance on the component. This timing is defined as the first moment that the probability of a failure of that component becomes larger than $\mathcal{P}$. In other words, we perform maintenance when the reliability becomes smaller than $\mathcal{J}$, where $\mathcal{J} = 1 - \mathcal{P}$. So,

$$T_{maintenance} \leq \arg\min_{t} \left(R^{\mathcal{V}}(t; t_0) < \mathcal{J} \right) \tag{5.1}$$

Overall, the modelling process included feature selection, dependent and independent variable formulation, data balancing, and the generation of training, validation and testing sets. We obtain a balanced dataset via undersampling with weekly remaining useful life intervals, which we divide into a training and test set. These datasets are used to train the three models to predict the remaining useful life based on the input features.

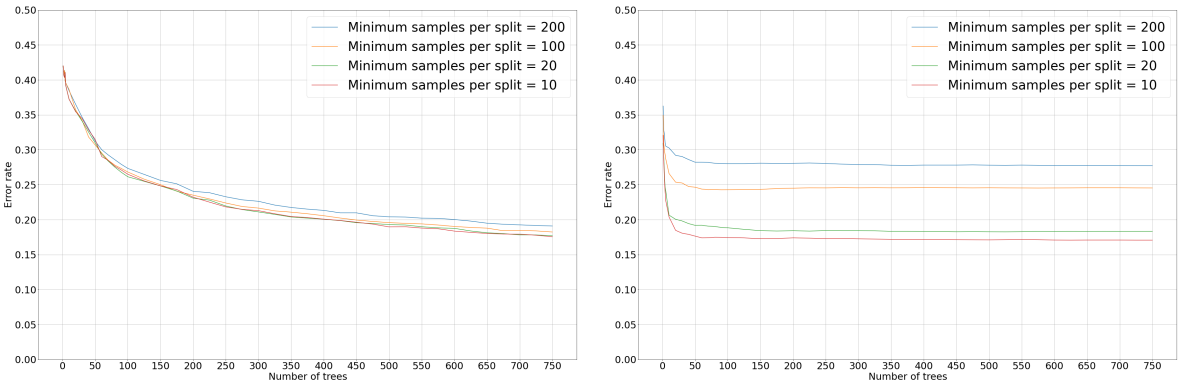
6. Results and discussion

This chapter presents the remaining useful life prediction and evaluation phase of the RUL prediction process of Ferreira & Gonçalves (2022). RUL prediction calculates the time to failure before the failure effectively occurs using the selected features, which the model has never seen before (Ferreira & Gonçalves, 2022). Therefore, the model predicts on the test set. Next, the evaluation phase is about evaluating the prediction results based on specific metrics and relating the prediction to the maintenance approach (Ferreira & Gonçalves, 2022). Thus, this chapter will highlight the important results, the performance of every tested combination is presented in Appendix A. Sections 6.1, 6.2, 6.3 and 6.4 present the results for the carts, MCB units, motors and crossbelt units, respectively. First, the hyperparameters are optimized for the trained model using the validation set. For visualisation purposes, the error rate is shown, which is the rate at which the order of events is predicted incorrectly. Thus, this equals $1 - C$, where C is Harrell's concordance index (Frisk et al., 2014). Then, the other metrics discussed in Section 4.3 are presented (RMSE, MAPE, average underestimation, average overestimation and training time), based on the testset. Thereafter, Section 6.5 illustrates model interpretability by presenting an example of one component for the three RUL prediction methods. Please note that these results are presented for $\mathcal{J} = 0.5$, meaning that a component is replaced when the estimated reliability drops below 0.5. This threshold is defined based on DPD's preference for underestimating the remaining useful life compared to overestimating it. Finally, Section 6.6 compares the overall performance of the survival tree, random survival forest and gradient-boosting model. Thus, this chapter provides the answers to the final Research Questions 1.5 and 1.6.

6.1 Results carts

This section presents the results for the survival tree, gradient-boosting model and random survival forest for the first component, the cart. Figure 6.1 shows the error rate for the gradient-boosting model (6.1a) and the random survival forest (6.1b) for all values for the minimum number of samples per split. We observe that the random survival forest converges considerably faster than the gradient-boosting model. The error rate stabilizes with 150 trees in the random survival forest, whereas the gradient-boosting model still decreases slightly at 700 trees. Moreover, Figure 6.1b shows that the random survival forest is much more sensitive to the parameter denoting the minimum number of samples per split. Therefore, the optimal value for the minimum number of splits for the random survival forest is 10. However, for the gradient-boosting model, this value is not as noticeable. Table 6.1 shows the best-performing configuration for the three RUL prediction methods and the values for all performance indicators. From this table, we can conclude that for the gradient-boosting model, a number of trees of 750 and a minimum number of samples per split of 20 is optimal. For the survival tree, a minimum of 10 samples per split is optimal. This parameter setting gives a concordance index of 80.98%, RMSE of 488.49 days and a MAPE of 444.44%. The gradient-boosting model with 750 trees and a minimum of 20 samples per split provides a better concordance index of 82.19%. This configuration gives better values for the RMSE and MAPE of 461.62 days and 442.63%, respectively. The random survival forest with 725 trees and a minimum of 10 samples per split gives the best concordance index at 82.35%. However, the RMSE and MAPE are 2.00% and 14.09% worse than the gradient-boosting model with values of 470.86 days and 504.98%, respectively. For all three models, it holds that the RMSE and MAPE are extremely high. The survival tree, gradient-boosting model

and random survival forest all have a higher tendency to overestimate the remaining useful life with on average 401.46, 408.70 and 372.31 days, respectively. The underestimation is on average lower at 194.48, 215.27 and 235.63 days for the survival tree, gradient-boosting model and random survival forest, respectively. Finally, the training times vary widely. The survival tree is the quickest RUL prediction model to train with 147.66 seconds, whereas the gradient-boosting model and random survival forest are much slower with 690.92 and 2264.30 seconds of training time for the optimal configurations of these models, respectively. Further interpretation of these results is presented in Section 6.6.



(a) Gradient-boosting model **(b)** Random survival forest

Figure 6.1: The error rate of the gradient-boosting model and random survival forest for different values for the minimum number of samples per split for the cart

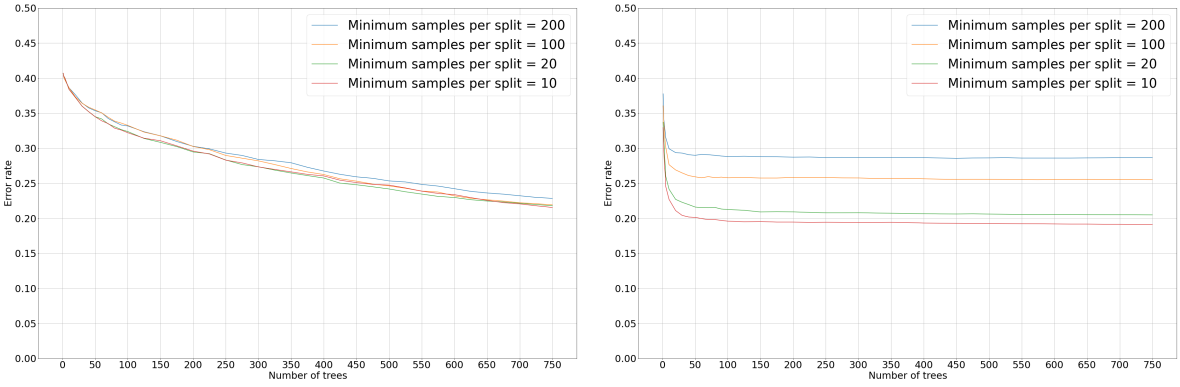
Model type (number of trees, minimum samples per split)	Evaluation metric					
	Concordance index	RMSE	MAPE	Average underestimation	Average overestimation	Training time (seconds)
ST (1, 10)	80.98	488.49	444.44	194.48	401.46	147.66
GB (750, 20)	82.19	461.62	442.63	215.27	408.70	690.92
RSF (725, 10)	82.35	470.86	504.98	235.63	372.31	2264.30

Table 6.1: Best model performance for the cart when $\mathcal{J} = 0.5$

Overall, all three models produce satisfactory results for the concordance index. On the other hand, the estimations of the useful life left are inadequate. As a result, the models accurately anticipate the sequence of failure yet predicting specific future decline is difficult. As a result, the random survival forest approach is most suited to the cart, as it has the greatest concordance index. However, the gradient-boosting model achieves a 69.49% reduction in training time for a decline of 0.16% in the concordance index. The survival tree reduces the training time by another 78.63% compared to the gradient-boosting model at the cost of a 1.21% reduction in the concordance index. Overall, the best-performing model in terms of the concordance index is the random survival forest with 725 trees and a minimum of 10 samples per split.

6.2 Results MCB units

This section presents the results for the gradient-boosting model and the random survival forest for the second component, the MCB. The error rate for the gradient-boosting model and the random survival forest are shown in Figure 6.2. Firstly, we observe that the minimum number of samples per split is nearly unimpactful to the error rate of the gradient-boosting model, whereas this parameter significantly influences the performance of the random survival forest. Moreover, we observe that the random survival forest rapidly converges while the gradient-boosting model slowly improves in terms of error rate. More specifically, the gradient-boosting model still slightly decreases the error rate at 700 trees, whilst the random survival forest is stable after approximately 150 trees. The optimal configuration for the gradient-boosting model and random survival forest is 750 trees with a minimum of 20 samples per split. The random survival forest achieves an extraordinary concordance index of 80.03% meaning that the order of component failure is predicted correctly for 80.03% of the cases. The gradient-boosting model also achieves a reliable concordance index of 77.80%. Surprisingly, the survival tree outperforms the other two models with a concordance index of 80.53%. The MAPE of this model is also best at 448.83% compared to 450.02% and 530.63% for the gradient-boosting model and random survival forest, respectively. On the other hand, the RMSE is best for the gradient-boosting model at 401.50 whereas the survival tree achieves the worst value at 482.18 and the random survival forest is at 474.79. Finally, the models all tend to overestimate more than underestimate. The average underestimation is 168.39, 174.67 and 172.33 for the survival tree, gradient-boosting model and random survival forest, respectively. Section 6.6 provides additional clarification of these findings.



(a) Gradient-boosting model **(b)** Random survival forest

Figure 6.2: The error rate of the gradient-boosting model and random survival forest for different values for the minimum number of samples per split for the MCB

Model type (number of trees, minimum samples per split)	Evaluation metric					
	Concordance index	RMSE	MAPE	Average underestimation	Average overestimation	Training time (seconds)
ST (1, 10)	80.53	482.18	448.83	168.39	399.69	482.92
GB (750, 20)	77.80	401.50	450.02	174.67	357.46	2450.64
RSF (750, 20)	80.03	474.79	530.63	172.33	404.20	6785.37

Table 6.2: Best model performance for the MCB when $\mathcal{J} = 0.5$

Overall, the three models produce acceptable concordance indexes. However, the remaining useful life estimations are erroneous. As a result, the models accurately anticipate the failure sequence. In other words, while the model can estimate the order of failure rather well, predicting specific future events is tough. As a result, the survival tree approach with a concordance index of 80.53% achieves the most suitable result for the MCB.

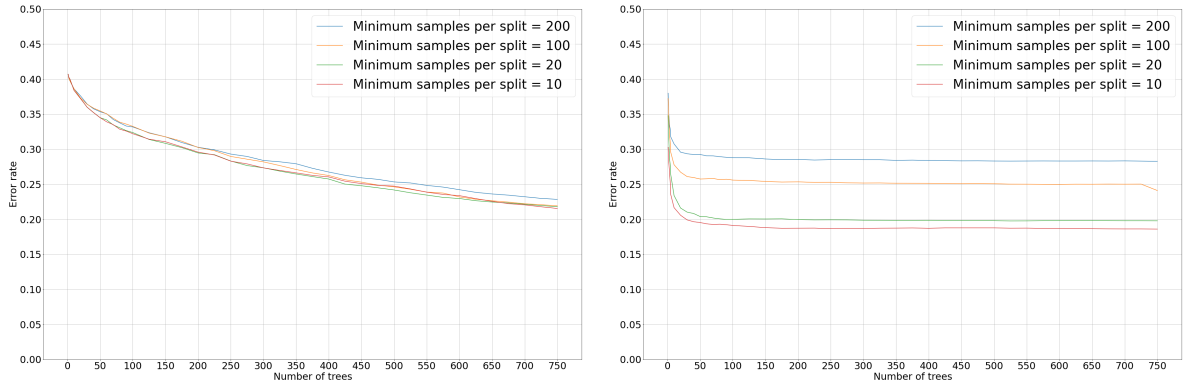
6.3 Results motors

This section discusses the findings for the survival tree, gradient-boosting model and random survival forest for the third component, the motor. Figure 6.3 provides a comparison of the error rate for the gradient-boosting model and random survival forest for the different values of minimum samples per split. Firstly, the random survival forest converges much quicker than the gradient-boosting model. In the random survival forest, the error rate stabilizes around 150 trees, while the gradient-boosting model still declines marginally at 700 trees. Furthermore, Figure 6.3 implies that the random survival forest is significantly more sensitive to the parameter representing the minimum number of samples per split. As a result, the ideal number of samples per split for the random survival forest is 10. However, this number is not as apparent for the gradient-boosting model. The best-performing configuration for the RUL prediction methods, as well as the values for all performance measures are shown in Table 6.3. Based on this table, we can deduce that 750 trees and a minimum number of samples per split of 10 are best for the gradient-boosting model. This combination has a 78.27% concordance index, RMSE of 402.57 days and MAPE of 452.28%. The random survival forest is optimal for 600 trees and a minimum of 10 samples per split. This configuration gives a concordance index of 81.27%, RMSE of 472.51 days and MAPE of 509.90%. Surprisingly, the survival tree provides the best concordance index of 82.47% with a minimum of 10 samples per split. However, the RMSE is the worst for this model. Therefore, the survival tree trained on this dataset can estimate the order of failure the best of the prediction models, but the timing of failure is estimated poorly. For all three models, it holds that the RMSE and MAPE are extremely high. The survival tree, gradient-boosting model and random survival forest all have a higher tendency to overestimate the remaining useful life with on average 478.32, 402.57 and 472.51 days, respectively. The underestimation is on average lower at 159.35, 171.85 and 171.09 days for the survival tree, gradient-boosting model and random survival forest, respectively. Finally, the training times vary widely. The survival tree is the quickest RUL prediction model to train with 375.73 seconds, whereas the gradient-boosting model and random survival forest are much slower with 2950.08 and 5521.34 seconds of training time for the optimal configurations of these models, respectively. Further interpretation of these results is presented in Section 6.6.

Model type (number of trees, minimum samples per split)	Evaluation metric					
	Concordance index	RMSE	MAPE	Average underestimation	Average overestimation	Training time (seconds)
ST (1, 10)	82.47	478.32	400.44	159.35	395.31	375.73
GB (750, 10)	78.27	402.57	452.28	171.85	360.08	2950.08
RSF (600, 10) [1]	81.27	472.51	509.90	171.09	391.34	5521.34

Table 6.3: Best model performance for the motor when $\mathcal{J} = 0.5$

Overall, the three models generate acceptable results regarding the concordance index. How-



(a) Gradient-boosting model **(b)** Random survival forest

Figure 6.3: The error rate of the gradient-boosting model and random survival forest for different values for the minimum number of samples per split for the motor

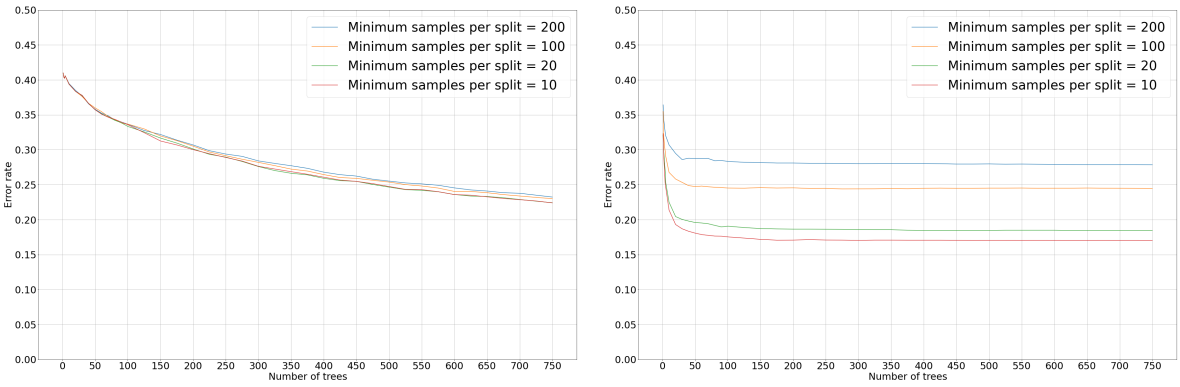
ever, the remaining useful life estimates are insufficient. As a consequence, the models correctly forecast the order of failure, although it is challenging to predict a precise moment of failure. As a result, the survival tree approach is most suited to MCB units, as it has the greatest concordance index and the best values for the MAPE and underestimation. Moreover, it is the quickest to train. Overall, the best-performing model in terms of the concordance index is the survival tree with a minimum of 10 samples per split.

6.4 Results crossbelt units

This section highlights the findings for the survival tree, gradient-boosting model and the random survival forest for the fourth component, the crossbelt. Figure 6.4 depicts the error rate for the gradient-boosting model and the random survival forest. To begin, we notice that the minimal number of samples per split has little effect on the gradient-boosting model's error rate, however, it has a considerable impact on the random survival forest's performance. The error rate decreases by approximately 0.1 when using 10 samples per split compared to 200 samples. Furthermore, the figure shows that the random survival forest quickly converges, but the gradient-boosting model gradually improves with regard to the error rate. In particular, the error rate of the gradient-boosting model still marginally reduces after 700 trees, but the random survival forest is level after roughly 150 trees. The optimal configuration for the random survival forest is with 475 trees and a minimum of 10 samples per split. The gradient-boosting model is optimal with 750 trees and a minimum of 10 samples per split. The survival tree is optimal for a minimum of 10 samples per split. The best concordance index is 79.28%, 77.49% and 82.78% for the survival tree, gradient-boosting model and random survival forest, respectively. The gradient-boosting model has the lowest concordance index, however, the values for the RMSE and average overestimation are better compared to the other RUL prediction methods at 417.30 and 373.35. The RMSE of the survival tree and random survival forest are 80.40 and 76.82 days higher. The average overestimation is 410.80 and 414.39 for the survival tree and random survival forest. The lowest average underestimation is achieved by the random survival forest with an average underestimation of 162.65 days,

[1]This model was only trained until 600 trees due to memory constraints. However, as can be seen in Figure 6.3b the performance hardly changes at that point.

compared to 174.38 and 188.28 for the gradient-boosting model and survival tree. Regarding the MAPE, the survival tree outperforms the other two models with a MAPE of 457.11 compared to 462.92 and 521.95 for the gradient-boosting model and random survival forest, respectively. Section 6.6 interprets these results in further detail.



(a) Gradient-boosting model **(b)** Random survival forest

Figure 6.4: The error rate of the gradient-boosting model and random survival forest for different values for the minimum number of samples per split for the crossbelt

Model type (number of trees, minimum samples per split)	Evaluation metric					
	Concordance index	RMSE	MAPE	Average underestimation	Average overestimation	Training time (seconds)
ST (1, 10)	79.28	497.70	457.11	188.28	410.80	616.24
GB (750, 10)	77.49	417.30	462.92	174.38	373.35	3064.49
RSF (475, 10)[2]	82.78	494.12	521.95	162.65	414.39	7392.92

Table 6.4: Best model performance for the crossbelt when $\mathcal{J} = 0.5$

Overall, both models provide adequate concordance indexes. Their RUL predictions, on the other hand, are erroneous. As a result, the RUL prediction models accurately anticipate the failure sequence, yet predicting specific future failures is tough. As a result, the random survival forest approach with 475 trees and a minimum of 10 samples per split achieves the greatest concordance index and is most suited for crossbelt.

6.5 Case study: model interpretability

As defined in Chapter 2, two types of trust are required for machine learning models. Regarding the model in general, global interpretability methods are used. Friedman (2001) introduced partial dependence plots to explain the interaction between a feature and the model output globally. As noted by Linardatos et al. (2020), Friedman's PDPs remain a popular choice for global model interpretability. Regarding the logic behind one individual prediction, local interpretability methods are available, such as LIME and SHAP. LIME and SHAP are by far the most comprehensive and dominant methods for visualising feature interactions (Linardatos et al., 2020). As SHAP is noted to over-weigh unlikely data points,

[2]This model was only trained until 475 trees due to memory constraints. However, as can be seen in Figure 6.4b the performance hardly changes at that point.

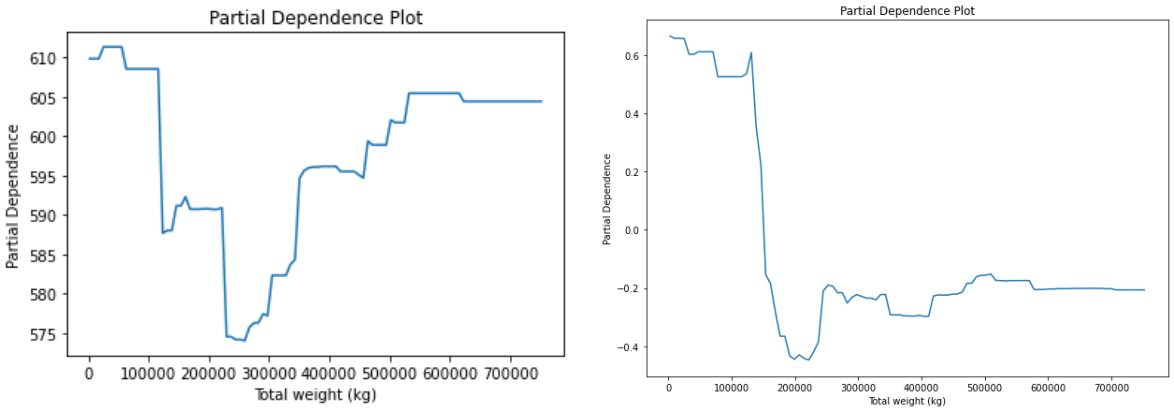
we use LIME here to explain one example. First, we discuss the global interpretability of the models using PDPs, after which we highlight the local interpretability utilizing LIME.

6.5.1 Global interpretability

Model interpretability is essential before deployment (Ribeiro et al., 2016). As noted, we use partial dependence plots to inspect the model's inner logic. Figure 6.5 shows the PDPs of the total weight and remaining useful life for the survival tree, gradient-boosting model and the random survival forest for the optimal models of the cart, as shown in Table 6.1. Partial dependence plots visualize the relationship between the target variable (remaining useful life) and a specific input feature (total weight). The y-axis of these plots represents the average predicted outcome of the target variable (the response) based on the input features while holding all other input features constant. Figures 6.5a, 6.5b and 6.5c all show different relations between the total weight and the remaining useful life. The partial dependence plots of the gradient-boosting model and random survival forest show somewhat decreasing trends. Especially the gradient-boosting model seems logical, which shows that the remaining useful life of a component drops steeply when the total weight increases from 120.000 to 200.000 kilograms. A similar decline is visualised in the PDP of the survival tree (Figure 6.5a). On the contrary, the PDP of the random survival forest shows a two-stage decline in remaining useful life when the total weight ranges from 150.000 to 200.000 and from 270.000 to 350.000 kilograms. Both the partial dependence plots of the survival tree and random survival forest show an increase in remaining useful life after the component has moved around 300.000 kilograms of weight. One coinciding feature between these three plots is the uneven flow in the graphs caused by the noise and large variation in values for the input features. Finally, the largest difference is the scale of the y-axis. The range of this axis for the gradient-boosting model is different compared to the survival tree and random survival forest. This is due to the methodology of these models. The gradient-boosting model focuses on the largest prediction error to grow a new tree. Due to this focus, this model might find different relations between the features and the remaining useful life. Moreover, the gradient-boosting model starts model building from the median RUL value. Hence, the interaction between the input features and the RUL might be smaller in magnitude. Overall, all three models have their interpretation of the data, that can globally be visualised by partial dependence plots. Moreover, these plots inform us about the degradation path of the components, that can be used to adapt operations.

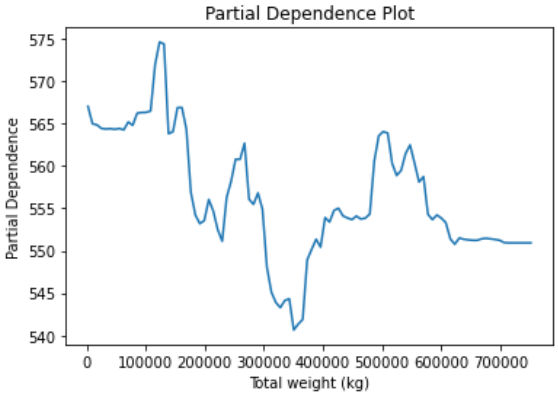
6.5.2 Local interpretability

Local interpretability increases the trust in one prediction. In other words, this is essential for the user's trust in an individual prediction to take action based on it (Ribeiro et al., 2016). To increase this trust, Local Interpretable Model-agnostic Explanations (LIME) is employed. Figure 6.6 shows the visualisations for the survival tree, gradient-boosting model and the random survival forest for a cart that has been in operation for 720 days. During these 720 days, the cart has transported 58225 parcels with a total weight of 326249 kilograms. The red lines to the left indicate decreasing relations between the feature and the remaining useful life, whereas a green line to the right means a positive/increasing relation between them. Firstly, we observe that all three plots look fairly similar. This component has a maximum time in use larger than 999.27 seconds which all three RUL prediction models point out as the largest



(a) Survival tree

(b) Gradient-boosting model

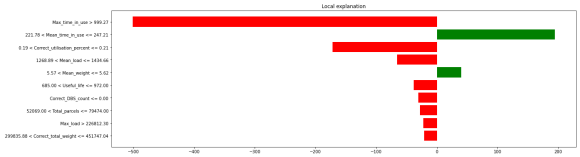


(c) Random survival forest

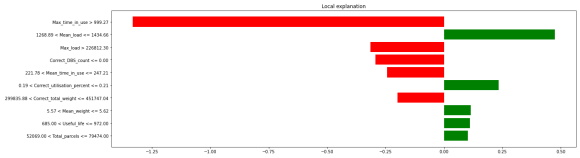
Figure 6.5: Partial dependence plots for the total weight for the survival tree, gradient-boosting model and the random survival forest

negative impact on the RUL of this cart. In other words, this large time that a parcel was on the component decreases the predicted remaining useful life of this component. Then, the second most influential characteristic is the mean load for the gradient-boosting model and the random survival forest, which is noted to have a positive effect on the RUL. The most notable difference is in the scale of the x-axis, which is similar for the survival tree and random survival forest but different for the gradient-boosting model. As the gradient-boosting model starts model building from the median RUL value, the interaction between the input features and the RUL might be smaller in magnitude.

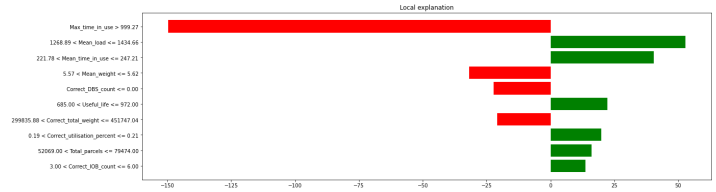
Overall, the three models interpret the data slightly differently, which is shown by the difference in partial dependence plots and LIME visualisations. As noted, the survival tree and random survival forest interpret the data similarly, whereas the gradient-boosting model differs slightly. Although all three models are suitable for model interpretation using PDPs and LIME, the survival tree and random survival forest seem more logical. All three models can be interpreted globally and locally by using partial dependence plots and LIME, which is essential to decide on maintenance actions and deploying the model.



(a) Survival tree



(b) Gradient-boosting model



(c) Random survival forest

Figure 6.6: LIME visualisation for the example cart for the survival tree, gradient-boosting model and the random survival forest

6.6 Comparison survival tree, gradient-boosting model and random survival forest

Based on the results per component, we can compare the three methods on their performance. Figure 6.7 shows a comparison of the concordance index of the models per component. From this figure, the first observation is that the concordance indexes are very close for the three models for each component. The largest performance difference is 5.19%, where the random survival forest outperforms the gradient-boosting model for the crossbelt by this amount. The random survival forest for the crossbelt then achieves the best performance of all models with a concordance index of 82.78%. The best performance for the gradient-boosting model is 82.19% for the cart, whereas the survival tree performs best with a concordance index of 82.47% for the motor. For the MCB, motor and crossbelt, it holds that the gradient-boosting model is outperformed by both the survival tree and the random survival forest. For the cart, it holds that the random survival forest performs best, then the gradient-boosting model and finally the 80.98% achieved by the survival tree is the worst.

In addition to the concordance index, we can also draw conclusions from the other performance indicators (see Table 6.1 to 6.4). For every dataset (i.e. every component), all three RUL prediction models have an extremely high RMSE and MAPE. In general, the gradient-boosting model achieves the lowest RMSE, followed by the random survival forest and then the survival tree. On the other hand, the survival tree generally outperforms the gradient-boosting model and random survival forest regarding the MAPE and training time of the RUL prediction methods. Compared to the high RMSE combined with the lower MAPE, this suggests that the survival tree and gradient-boosting model make smaller percentage errors. In other words, the survival tree and gradient-boosting model have a smaller relative error, even though the absolute differences between the predicted and actual values might be larger, as reflected by the high RMSE. Finally, the survival tree has the shortest training time, followed by the gradient-boosting model. The random survival forest takes on average 91.52 minutes to train for the optimal configuration.

The final criteria for the RUL prediction model were explainability and interpretability. As

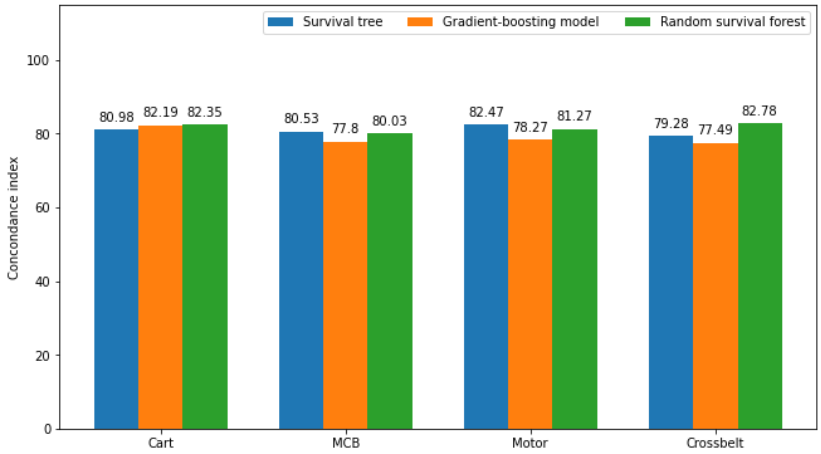


Figure 6.7: Comparison of concordance index per component for the gradient-boosting model and random survival forest

all three models are decision tree-based, the explainability is sufficient. However, as a single survival tree can easily be visualised, the survival tree is easiest to explain and use for communication purposes. Section 6.5 showed the model interpretability for all three RUL prediction models. The survival tree, gradient-boosting model and random survival forest can all be interpreted using partial dependence plots (PDPs) and local interpretable model-agnostic explanations (LIME). Therefore, the models are equal in interpretability. However, the global interpretations are different for the RUL prediction models. Mainly the survival tree and random survival forest seem logical. Therefore, these two RUL prediction models are favourable in terms of interpretability.

The high RMSE and MAPE for the three RUL prediction models in general can have several causes. Firstly, the long-term uncertainty is too high as also described by the variance in Table 3.2, causing larger errors over time. Therefore, the RMSE increases but the order of failure (C-index) is still accurate. Moreover, the high values for the RMSE and MAPE could be caused by the size of the dataset. Most likely, the number of different degradation paths is too high compared to the number of observations per path.

Another difference between the three methods is the improvement pace of the error rate. As noted, the random survival forest converges more rapidly than the gradient-boosting model. Firstly, this is caused by the nature of the methodology as presented in Chapter 4. The gradient-boosting model is subject to a learning rate to prevent overfitting. Logically, the model will converge more rapidly if the learning rate is higher, but it also increases the risk of overfitting. Moreover, the models differ in their sensitivity to the hyperparameters. The survival tree and random survival forest are more sensitive to varying values for the minimum number of samples per split.

Finally, the random survival forest poses another limitation, memory usage. As noted in Chapter 6, the random survival forest with a minimum number of samples per split equal to 10 was only partially trained for the MCB, motor and crossbelt due to memory issues. Neither DPD's server nor local computer programmes were able to deal with the memory consumed by this configuration. However, this is not a significant issue as the model achieves similar performance for a much lower number of trees.

All in all, all three RUL prediction models perform well in terms of the concordance index but are poor with regard to the root mean squared error and the mean absolute percentage errors. Thus, the order of failure is straightforward to predict, while the timing has proven to be more challenging due to the high variance. In general, the random survival forest performs better in terms of the concordance index. The gradient-boosting model outperforms the random survival forest and the survival tree in terms of RMSE. The survival tree outperforms the other two models in terms of MAPE, training time and model explainability and interpretability. Overall, the remaining useful life predictions are inaccurate, so the best-performing model based on the concordance index is the survival tree for the MCB and motor and the random survival forest for the cart and crossbelt.

7. Conclusion

This chapter discusses the conclusions, implications, limitations and suggestions for future research following this study. Firstly, Section 7.1 discusses the main findings. Secondly, Section 7.2 discusses the business and academic implications and recommendations. Finally, Section 7.3 describes the limitations and suggestions for future research.

7.1 Research conclusion

This research aimed to develop a model to predict the remaining useful life of the critical components of the sortation machine of the location in Oirschot of DPD Netherlands based on its usage. This development was subject to several constraints. The model needed to be suitable for noisy, aggregated tabular data from right-censored observations. In addition, the model should be both interpretable and explainable. Therefore, we deployed three models for comparison: a survival tree, a gradient-boosting model and a random survival forest. In terms of concordance index performance, all three models delivered satisfactory results. Thus, the survival tree, gradient-boosting model and random survival forest estimate the order of failure accurately. This suggests that the usage data does influence the degradation, but the data is currently insufficient to accurately predict the moment of failure. This follows from the high values for the RMSE and MAPE which can be attributed to a variety of factors. For starters, the long-term uncertainty is excessive due to the wide range of usable life. As a result, the RMSE increases yet the order of failure (C-index) remains correct. Furthermore, the large RMSE and MAPE values might be due to the size of the dataset. The number of various deterioration routes is most likely too large in comparison to the number of observations per path. Moreover, future degradation varies largely, increasing the difficulty in anticipating the exact moment of failure. Thus, the models identify the clusters of operational profile well, but within these clusters, the variance in remaining useful life is too high. Finally, both local and global model interpretation techniques showed that the survival tree and random survival forest show the most logical relation between input features and the remaining useful life prediction. Therefore, we conclude that the survival tree, gradient-boosting models and random survival forest are promising methods to predict the remaining useful life of the critical components of the sorting machine of DPD Netherlands based on their usage, but data quality improvements are required to ensure the applicability of these models.

7.2 Implications and recommendations

This section discusses the business and academic implications, as well as recommendations to DPD Netherlands.

7.2.1 Academic implications

As noted, the main contribution of this research was to investigate the applicability of survival ensemble methods to complex machinery. The application of these methods has already been widely proven to be accurate in the bioinformatics field. However, the application to everyday asset management decision-making was lacking. This research contributed to this gap by concluding that these methods are not suitable yet. As noted by Frisk et al. (2014); Hothorn et al. (2006); Afrin et al. (2018) amongst others, the models perform well. However, these papers solely discuss the concordance index, whereas the prediction accuracy is

hardly discussed. This research adds that the concordance index is indeed excellent, however, the prediction accuracy is insufficient for deployment in practice when censoring occurs, and the components have a long expected lifetime. Thus, we conclude that survival trees, random survival forests and gradient-boosting models apply to right-censored tabular data on machine components solely for predicting the sequence of failure and not yet for remaining useful life prediction. Moreover, the underestimation and overestimation of these models were investigated. From that, we conclude that when handling a large number of critical components, favouring underestimation of the remaining useful life, the gradient-boosting model is favourable. Finally, we note that when using noisy data with a large expected lifetime the widely-noted benefits of ensemble methods are not apparent compared to a single survival tree. However, when the data contain less variance, ensemble methods might become more relevant.

7.2.2 Business implications and recommendations

As noted in the previous sections, the concordance index of the three RUL prediction models is adequate. Therefore, the models accurately assess the risk of failure. Unfortunately, the remaining useful life prediction is inaccurate. Further data has to be gathered until more component failures have occurred during the coming years. Then, the models should be able to detect trends in the data better for the remaining useful life prediction. Therefore, we advise DPD to use the survival tree, which yields the best trade-off between the concordance index and model explainability, for maintenance prioritisation. As the order of failure can be accurately predicted, this research suggests a clear relation between component usage and degradation. Therefore, maintenance prioritisation based on component usage instead of a simple constant time interval policy is suggested. As noted by Arunraj & Maiti (2007), the concept of risk-based maintenance was developed to inspect the component with high risk more frequently. Therefore, the risk-based maintenance framework comprises two main phases: risk assessment and maintenance planning based on risk (Arunraj & Maiti, 2007). The first stage consists of assessing the risk of failure of the components, whereas the second stage prioritizes the inspection based on this quantified risk so that the total risk can be minimized using risk-based maintenance. Moreover, we advise reevaluating these models, once data availability and data quality improves. Thus, as the usage data is promising for maintenance prioritisation, we advise DPD to adjust their current strategy. Thus, DPD should shift from a constant interval replacement policy to a risk-based inspection planning, where the risk estimation is based on the survival tree. This RUL prediction method achieves a comparable concordance index, but is much more explainable to maintenance engineers.

7.3 Limitations and future research

As noted by Li et al. (2016), RUL prediction causes high demands on data access and quality as well as the capability to deal with these data. These demands also proved to be the difficulty in this research. The data quality is insufficient to predict the remaining useful life accurately. Therefore, additional research is needed on the prediction accuracy (in terms of RMSE and MAPE, not concordance index) of survival trees, gradient-boosting models and random survival forests when data contain a high amount of variation. Consequently, the impact of data discretization techniques on variation in RUL prediction requires additional research. Moreover, additional effort is needed from the industry to gather the right data

and ensure that the data is of sufficient quality as a significant part of the data was missing in this research. This prediction accuracy is expected to improve when more data becomes available. However, the companies should be able to deal with this data which is also an important area for research. Moreover, as this research proved that component reliability is dependent on usage, a more practical research suggestion is to see how DPD can elongate the useful life of components by optimizing their usage, especially since DPD's usage is subject to peak seasons. Moreover, a promising extension of this thesis would be to optimize the maintenance planning based on the risk of failure predicted by the survival trees, subject to the limited time available in the daily maintenance window, considering the locations of the components. Finally, as noted this system is too complex to exclude system experts from the feature selection process, although (Prytz et al., 2015) noted that this procedure is suboptimal. Therefore, to increase practical relevance for complex machinery, other approaches should be designed to include expert opinion and maximize prediction accuracy.

To conclude, this research proved that survival trees, random survival forests and gradient-boosting models assess the risk of failure of a component noticeably well. However, the prediction errors require more attention in academic literature before their implementation in practice is in order. Moreover, we would like to stress the importance of data gathering by companies in advance of considering remaining useful life predictions. Finally, usage data proved promising for maintenance prioritisation of the critical components on the sorting installation of DPD's location in Oirschot.

References

Afrin, K., Illangovan, G., Srivatsa, S. S., & Bukkapatnam, S. T. (2018). Balanced random survival forests for extremely unbalanced, right censored data. *arXiv preprint arXiv:1803.09177*.

Ahmadzadeh, F., & Lundberg, J. (2014). Remaining useful life estimation. *International Journal of System Assurance Engineering and Management*, *5*(4), 461–474.

Arts, J. (2017). Maintenance modeling and optimization. *TU/e Eindhoven: Eindhoven, The Netherlands*, *526*.

Arts, J., Basten, R., & Houtum, G.-J. v. (2019). Maintenance service logistics. In *Operations, logistics and supply chain management* (pp. 493–517). Springer.

Arunraj, N., & Maiti, J. (2007). Risk-based maintenance—techniques and applications. *Journal of hazardous materials*, *142*(3), 653–661.

Barlow, M. A. (2015). *Predictive maintenance: A world of zero unplanned downtime*. O'Reilly Media.

Batista, G. E., Prati, R. C., & Monard, M. C. (2004). A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD explorations newsletter*, *6*(1), 20–29.

Breiman, L. (2001). Random forests. *Machine learning*, *45*, 5–32.

Buchaiah, S., & Shakya, P. (2022). Bearing fault diagnosis and prognosis using data fusion based feature extraction and feature selection. *Measurement*, *188*, 110506.

Chen, J., Huang, R., Chen, Z., Mao, W., & Li, W. (2023). Transfer learning algorithms for bearing remaining useful life prediction: A comprehensive review from an industrial application perspective. *Mechanical Systems and Signal Processing*, *193*, 110239.

Ebeling, C. E. (2010). *An introduction to reliability and maintainability engineering*. Waveland Press.

Ferreira, C., & Gonçalves, G. (2022). Remaining useful life prediction and challenges: A literature review on the use of machine learning methods. *Journal of Manufacturing Systems*, *63*, 550–562.

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, 1189–1232.

Frisk, E., Krysander, M., & Larsson, E. (2014). Data-driven lead-acid battery prognostics using random survival forests. In *Annual conference of the phm society* (Vol. 6).

Hothorn, T., Bühlmann, P., Dudoit, S., Molinaro, A., & Van Der Laan, M. J. (2006). Survival ensembles. *Biostatistics*, *7*(3), 355–373.

Ishwaran, H., Kogalur, U. B., Blackstone, E. H., & Lauer, M. S. (2008). Random survival forests.

Javed, K., Gouriveau, R., Zemouri, R., & Zerhouni, N. (2012). Features selection procedure for prognostics: An approach based on predictability. *IFAC Proceedings Volumes*, *45*(20), 25–30.

Kundu, P., Darpe, A. K., & Kulkarni, M. S. (2020). An ensemble decision tree methodology for remaining useful life prediction of spur gears under natural pitting progression. *Structural Health Monitoring*, *19*(3), 854–872.

Lei, Y., Li, N., Guo, L., Li, N., Yan, T., & Lin, J. (2018). Machinery health prognostics: A systematic review from data acquisition to rul prediction. *Mechanical systems and signal processing*, *104*, 799–834.

Li, Z., Wang, K., & He, Y. (2016). Industry 4.0-potentials for predictive maintenance. In *6th international workshop of advanced manufacturing and automation* (pp. 42–46).

Liao, H., Zhao, W., & Guo, H. (2006). Predicting remaining useful life of an individual unit using proportional hazards model and logistic regression model. In *Rams'06. annual reliability and maintainability symposium, 2006.* (pp. 127–132).

Linardatos, P., Papastefanopoulos, V., & Kotsiantis, S. (2020). Explainable ai: A review of machine learning interpretability methods. *Entropy*, *23*(1), 18.

Loh, W.-Y. (2011). Classification and regression trees. *Wiley interdisciplinary reviews: data mining and knowledge discovery*, *1*(1), 14–23.

Ma, Z., Chen, M., Cao, J., Yao, Y., & Liu, Y. (2020). Suitable feature selection for prediction of lithium-ion batteries remaining useful life. In *2020 7th international conference on information, cybernetics, and computational social systems (iccss)* (pp. 728–732).

Nahmias, S., & Olsen, T. L. (2015). *Production and operations analysis.* Waveland Press.

Pölsterl, S. (2020). scikit-survival: A library for time-to-event analysis built on top of scikit-learn. *Journal of Machine Learning Research*, *21*(212), 1-6. Retrieved from `http://jmlr.org/papers/v21/20-729.html`

Price, J., & Mathew, J. (2000). The constant-interval replacement model for preventive maintenance: A new perspective. *International Journal of Quality & Reliability Management*.

Prytz, R., Nowaczyk, S., Rögnvaldsson, T., & Byttner, S. (2015). Predicting the need for vehicle compressor repairs using maintenance records and logged vehicle data. *Engineering applications of artificial intelligence*, *41*, 139–150.

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (pp. 1135–1144).

Saeys, Y., Inza, I., & Larranaga, P. (2007). A review of feature selection techniques in bioinformatics. *bioinformatics*, *23*(19), 2507–2517.

Si, X.-S., Wang, W., Hu, C.-H., & Zhou, D.-H. (2011). Remaining useful life estimation–a review on the statistical data driven approaches. *European journal of operational research*, *213*(1), 1–14.

Sikorska, J. Z., Hodkiewicz, M., & Ma, L. (2011). Prognostic modelling options for remaining useful life estimation by industry. *Mechanical systems and signal processing*, *25*(5), 1803–1836.

Tiddens, W. W., Braaksma, A. J. J., & Tinga, T. (2018). Selecting suitable candidates for predictive maintenance. *International Journal of Prognostics and Health Management*, *9*(1).

Tinga, T., & Janssen, R. (2014). Simulation based comparison of predictive maintenance policies.

Wang, H., & Li, G. (2017). A selective review on random survival forests for high dimensional data. *Quantitative bio-science*, *36*(2), 85.

Wirth, R., & Hipp, J. (2000). 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, pp. 29–39).

Wu, D., Jennings, C., Terpenny, J., Gao, R. X., & Kumara, S. (2017). A comparative study on machine learning algorithms for smart manufacturing: tool wear prediction using random forests. *Journal of Manufacturing Science and Engineering*, *139*(7).

A. Results

This appendix presents a more elaborate presentation of the results of the testing dataset. The table below shows the optimal number of trees, training time, concordance index, RMSE, MAPE, and average overestimation and underestimation for the single survival tree, gradient-boosting model, and random survival forest for a minimum number samples per split of 10, 20, 100 and 200. The best values for all the performance indicators are depicted in bold.

	Survival tree				Gradient-boosting				Random survival forest			
Minimum number of samples per split	10	20	100	200	10	20	100	200	10	20	100	200
Cart												
Optimal number of trees	1	1	1	1	750	750	750	750	725	525	575	750
Training time (sec)	147.66	138.64	144.80	137.61	636.28	690.92	641.17	650.75	2264.30	3036.19	1930.95	2494.32
C-index	**80.98**	80.33	75.32	72.71	82.10	**82.19**	81.81	80.82	**82.35**	81.05	75.16	72.03
RMSE	488.49	**481.71**	493.02	488.02	461.85	461.62	**461.11**	464.65	**470.86**	475.55	497.88	508.97
MAPE	**444.44**	459.51	580.65	572.33	445.37	**442.63**	447.11	454.85	**504.98**	549.54	625.80	661.51
Average overestimation	**401.46**	403.96	437.24	428.02	404.39	408.70	**401.01**	408.75	**372.31**	389.86	437.08	455.62
Average underestimation	**194.48**	196.98	226.42	251.26	224.33	**215.27**	228.50	230.01	235.63	**233.76**	262.27	271.05
MCB												
Optimal number of trees	1	1	1	1	750	750	750	750	625 *	750	750	450
Training time (sec)	482.92	449.03	452.17	459.58	2749.24	2450.64	2675.62	2451.42	2749.24	6785.37	6635.90	4002.73
C-index	**80.53**	79.13	72.70	69.07	77.42	**77.80**	77.55	76.82	77.42	**80.03**	74.41	70.94
RMSE	482.18	484.67	481.16	**480.45**	**397.73**	401.50	402.23	405.92	**397.73**	474.79	484.47	488.13
MAPE	**448.83**	467.57	518.02	538.99	**443.75**	450.02	448.16	455.16	**443.75**	530.64	584.40	608.93
Average overestimation	**399.69**	409.36	427.39	440.58	**355.67**	357.46	363.74	363.59	**355.67**	404.20	439.59	451.20
Average underestimation	**168.39**	178.14	195.60	200.08	175.79	174.67	**172.69**	176.09	175.79	**172.33**	182.58	186.78
Motor												
Optimal number of trees	1	1	1	1	750	750	750	750	600 **	525	600	750
Training time (sec)	375.73	375.27	374.15	371.47	2950.08	2724.57	2501.80	2497.15	5521.34	4661.59	6796.64	6512.99
C-index	**82.47**	81.13	74.12	71.10	**78.27**	78.08	78.15	77.11	**81.27**	80.51	75.48	71.91
RMSE	478.32	483.73	498.68	**478.04**	**402.57**	407.43	403.54	406.47	**472.51**	484.81	493.39	496.01
MAPE	**400.44**	434.18	533.96	522.56	**452.28**	455.08	**452.28**	463.37	**509.90**	560.91	623.48	664.51
Average overestimation	**395.31**	410.10	447.13	431.94	**360.08**	365.70	360.38	363.15	**391.34**	415.70	477.99	458.73
Average underestimation	**159.35**	164.38	203.37	203.22	171.85	**170.75**	172.28	173.96	**171.09**	174.00	181.28	184.27
Crossbelt												
Optimal number of trees	1	1	1	1	750	750	750	750	475***	450	275	750
Training time (sec)	616.24	608.38	599.64	595.56	3064.49	3080.53	2997.93	2978.50	7392.92	5415.42	3312.46	10414.14
C-index	**79.28**	77.81	73.26	70.52	**77.49**	77.26	76.82	76.66	**82.78**	81.50	75.39	72.23
RMSE	497.70	498.57	494.38	**491.38**	**417.30**	420.63	420.62	425.19	494.12	495.20	501.53	497.23
MAPE	**457.11**	495.95	520.72	513.44	**462.92**	465.93	472.04	482.56	**521.95**	542.58	610.37	623.51
Average overestimation	**410.80**	417.12	435.93	435.05	**373.35**	375.11	378.26	379.64	**414.39**	425.36	455.94	460.10
Average underestimation	**188.28**	195.14	210.42	229.35	**174.38**	177.03	174.77	175.68	162.65	**162.55**	172.87	174.61

* This model was only trained until 625 trees due to memory constraints. However, as can be seen in Figure 6.2b the performance hardly changes at that point.
** This model was only trained until 600 trees due to memory constraints. However, as can be seen in Figure 6.3b the performance hardly changes at that point.
*** This model was only trained until 425 trees due to memory constraints. However, as can be seen in Figure 6.4b the performance hardly changes at that point.