



MASTER THESIS

Construction of a proactive alert management model by using artificial intelligence

P. C. Ventevogel (Pim Cornelis)

s1600672

Examination Committee

Dr. M. C. van der Heijden

Dr. E. Topan

Ir. K. Alizadeh

University of Twente

University of Twente

Independent
Aerospace Company

Educational Institution

University of Twente

Faculty of Behavioural Management and Social Sciences

Department of Industrial Engineering and Business Information Systems

Educational Program

MSc. Industrial Engineering and Management

Specialisation: Production and Logistics Management

Orientation: Supply Chain & Transport Management

UNIVERSITY OF TWENTE.

Preface

Dear reader,

This thesis is the result of my master graduation assignment for the Production & Logistics Management specialization of the Industrial Engineering and Management Master's degree at the University of Twente. While being immensely challenging and frustrating at times, executing this graduation assignment has taught me a lot and has been a delightful ending for my time as a student.

In this preface I would like to take the opportunity to express my gratitude to the people who assisted me with the realization of this thesis. First, I would like to thank Kaveh Alizadeh for being my intellectual sparring partner, his guidance, and feedback throughout the assignment.

Second, I would like to thank Dr. Matthieu van der Heijden and Dr. Engin Topan for their support, valuable input, and feedback on my draft reports.

Finally, I want to thank family, friends, and girlfriend for their support and help.

Pim Ventevogel

Utrecht, The Netherlands

18-11-2020

Executive Summary

Introduction

The following graduation thesis '*Construction of a proactive alert management model by using artificial intelligence*' elaborates the process of incorporating a machine learning algorithm in an alerting generating tool. This thesis is conducted at an Independent Aerospace Company (IAC), more specifically for the Component Maintenance & Availability (CMA) program. The CMA program concerns rotatable parts of aircraft, which are parts for which it is economically worthwhile to restore them to a 'as good as new' condition upon failure. The CMA program consists of two sub-programs, which are the forward exchange (FE) and the performance exchange (PE) program. The FE program is a program where the IAC keeps inventory, on behalf of its customers. If customers require a spare part, they can order it from the IAC, and the IAC will send the part to the customer. The PE program is a program which promises in time delivery of repairs, even if repair shops are slacking. To achieve this the IAC uses its inventory pool to ensure in time delivery. As the IAC is currently facing backorders on a frequent bases, the IAC aims to improve their performance. To aid in this process, they have requested the creation of a proactive alerting model, which notifies operational planners of potential future problems.

To assist in the construction of this model, the following main research question is defined:

“How can the Independent Aerospace Company construct a proactive alert generation tool, which automatically recognizes and prioritizes potential problematic situations and notifies the operational planners?”

Status Quo

Historically, the majority of demand came from the FE program. However, due to the changing landscape in the aviation industry, the demand for the FE program is decreasing, while the demand for PE is increasing. As the existing alerting tool is created during the time when the majority of transactions came from the FE program, it is no longer appropriate, as it does not incorporate the PE demand. To determine the influence on the inventory pool for the PE program, the IAC has to determine whether they will finish the repair within the agreed time or not, as failing to do so, will result in a delivery from the inventory pool.

Methodology

To determine whether the contracted turnaround time will be met, a machine learning algorithm is used. From the literature review, the Artificial Neural Network, which is a deep learning algorithm, seemed the most promising. The classifier uses historical repair shop information, repair type information and contracted turnaround times to predict whether the agreed turnaround time will be exceeded or not. The artificial neural network is able to predict with an accuracy of 75% if an order will be finished in time.

The output of this classification model, together with existing forecasting tools for the FE demand are used to determine the likelihood of backorders. This likelihood is approximated using a Monte Carlo simulation, which simulates the development of the inventory level over a two-week time scope. For the orders which have the highest chance of facing backorders, an alert is generated.

Results

To validate the quality of the Artificial Neural Network, its performance is compared against a random forest. Similar to results found in literature, the random forest outperformed the neural network, and achieved an accuracy of 75.5%. Furthermore, features regarding the historical performance of the repair shops were highly correlated. By dropping the less important, highly correlated features the performance was increased to 76.7% accuracy.

From evaluation of the input of the classification model, the most important input variables appeared to be the turnaround time which is contracted with the customer, and the short-term performance of the executing repair shop. These results are intuitive, as a shorter agreed turnaround time means that there is less room for error. Furthermore, if a repair shop has recently been performing well, it is likely that for the new repair order, this trend will continue.

The model is evaluated against the judgement of the planners at the IAC by creation of 50 fictional scenarios. These scenarios were then evaluated by two operational planners and a tactical planner of the IAC. A high correlation was found between the scores provided by the planners, and the chance of backorders put out by the model. Furthermore, the mean absolute deviation between the output of the planners and the new model was 45% lower than the mean absolute deviation between the planners and the old reactive model. Showing that the new model is more in line with the estimates of the planners, and thus is more suitable for providing alerts to the operational planners.

Recommendations

First, should try to make use of repair shop data. Although external shops are reluctant to share data, the internal data should be available. From this data, important determinants for turnaround times can be deducted, such as: availability of shop replaceable units, number of items in queue and scheduled date of repair. This data can all be used to determine if a repair will be finished in time, and thus if a performance exchange is likely to happen or not. Inclusion of these metrics is likely to improve the quality of the classification model.

Second, the IAC should investigate methods to include the core return times. The core return times, are the times between delivery by the IAC, and receiving the broken part back from the customer. In this research these are neglected, which limits the time scope that can be considered.

Third, the IAC could research the inclusion of customer desire for performance exchanges. Currently, the model assumes that every contracted repair that is exceeding the agreed TAT, will result in a performance exchange. In reality this is not the case, as customers get the option to obtain a performance exchange, but do not have to accept this. Therefore, the impact on the inventory pool of the IAC is currently overestimated.

Limitations

The first limitation to the results is the noise in the dependent variable of the classification models. This noise is caused interventions executed by the operational planners, reducing throughput times of contracted repairs. This caused shorter turnaround times than would have occurred if no intervention were used. This causes confusion to the model, as almost identical situations have different class labels.

The second limitation is the access to data. Due to covid-19, and the strict regulations of data sharing within the IAC, access to data was limited. During the initial stages of this thesis, a dataset was constructed, with all information that seemed relevant at that time. However, in later stages more insight in the processes was obtained, creating demand for new data. This data, however, could no longer be obtained.

Table of Contents

Preface	1
Executive Summary	2
1 Problem introduction	6
1.1 Company description	6
1.2 Component Maintenance and Availability Programs	7
1.3 Motivation for research	8
1.4 Problem statement.....	9
1.5 Research Questions & Approach	9
1.6 Scope.....	10
2 Present situation	11
2.1 Special situations in Forward Exchange program.....	11
2.2 Increasing demand of Performance Exchange Program	13
2.3 Situations causing disruptions.....	14
2.4 Performance Indicators	16
2.5 Available interventions	16
2.6 Means of receiving alerts	19
2.7 Availability of Data.....	20
2.8 Conclusion	21
3 Literature review	22
3.1 What is machine learning?.....	22
3.2 Data preparation	23
3.3 Model Selection	30
3.4 Training the model	34
3.5 Evaluation of the model	34
3.6 Hyper parameter tuning	36
3.7 Distribution Fitting.....	38
3.8 Function approximation	39
3.9 Conclusion	39
4 Methodology	41
4.1 Conceptual solution design	41
4.2 Data	43
4.3 Classification Model	47
4.4 Monte Carlo simulation	50
4.5 Conclusions.....	55
5 Results.....	56
5.1 Performance of the Artificial Neural Network.....	56

5.2	Feature evaluation	57
5.3	Validation of the alert generating model.....	61
5.4	Conclusions.....	66
6	Conclusions, limitations and recommendations.....	67
6.1	Conclusion	67
6.2	Recommendations	68
6.3	Discussion	69
6.4	User interface design.....	71
	References.....	74
	APPENDIX	77
	APPENDIX A: Detailed Description of Available Data	77
	APPENDIX B: Interview questions operational planners	78
	APPENDIX C: Box plots of categorical Data	80
	APPENDIX D: Data pre-processing	81
	APPENDIX E: Bayesian Optimization.....	85
	APPENDIX F: Hyper parameter tuning of Random Forest.....	87
	APPENDIX G: Distribution Fitting.....	89
	APPENDIX H: Drawing from conditional probability.....	90

1 Problem introduction

This Master Thesis encompasses research conducted for an Independent Aerospace Company. The main subject of this Thesis is the Component Maintenance & Availability (CMA) program. This chapter of the Thesis further introduces the company, its history, and its way of conducting business (section 1.1), the content of the CMA program (section 0), and the challenges concerning the CMA program, which motivated the IAC to initialize this project (section 1.3). In section 1.4 the problem statement is conducted, based on initial conversations with my company supervisor. Section 1.5 elaborates how the problem is approached and discusses the (sub) research questions. Finally, in section 1.6 the theoretical scope is explained.

1.1 Company description

This section is removed in the public version of this Thesis

1.2 Component Maintenance and Availability Programs

The CMA program concerns rotatable parts of aircraft, which are parts for which it is economically worthwhile to restore them to a 'as good as new' condition upon failure. The CMA program consists of three different subprograms which are: The Forward Exchange (FE) program, the Performance Repair (PR) Program, and the Performance Exchange (PE) program. In the remainder of this section of these programs is shortly discussed, furthermore in Figure 1 a schematic overview of the three programs is given. In this figure, the numbers represent the order in which the flows of goods or information occur.

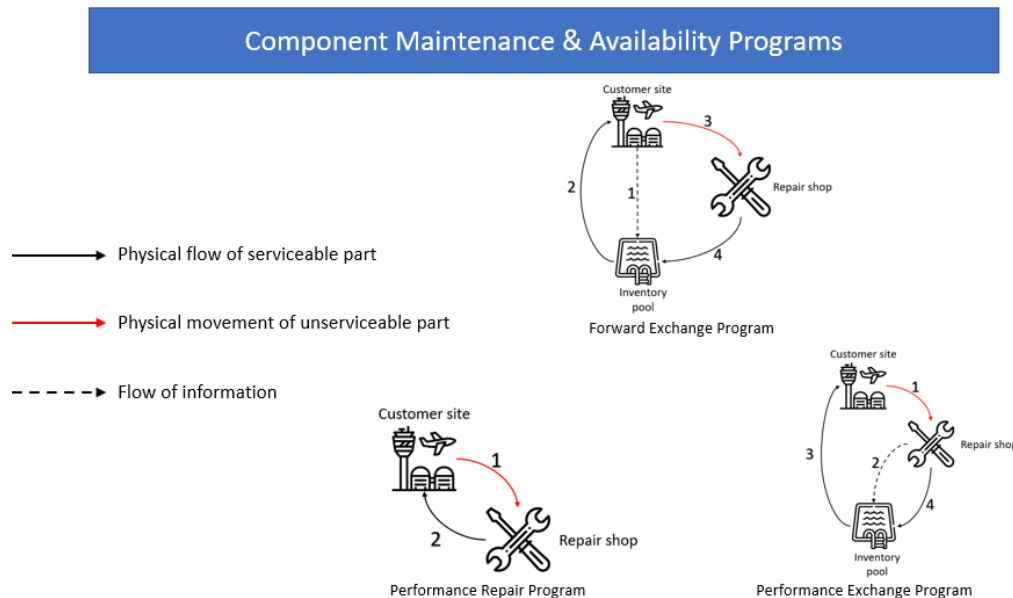


Figure 1 Component Maintenance & Availability Programs

1.2.1 Forward Exchange Program

The first program that is incorporated in the CMA program, is the Forward Exchange (FE) program. The FE program, as shown in Figure 1, is a basic version of the program, which will be further extended in section 2.1. The flow of goods and information is as follows: A customer orders a serviceable part from the inventory pool of the IAC, which is then sent to the customer. Then, once the part arrives at the customer, the part is replaced in the aircraft, upon which they send their unserviceable part to one of the repair shops that are part of the supply chain of the IAC. The part is then repaired and sent back to the inventory pool, where it will stay until a customer requires another part. Unfortunately, this is only a basic version of the FE Program, and the real supply network is a lot more complex.

1.2.2 Performance Repair Program

The Performance Repair (PR) program is a program where the IAC repairs part for external customers. On these repairs, maximum turnaround times are contracted. The PR program is backed by the Performance Exchange, so when the repair shops are unable to meet these turnaround times, the Performance Exchange program ensures intime deliveries. In Figure 1, the flow of goods is shown. For this program, the flow of goods is fairly simple, the customer sends an unserviceable part to the IAC repair shop, and the repaired part is sent back to the customer.

1.2.3 Performance Exchange Program

The Performance Exchange (PE) Program is used in two different situations. The first is as introduced in the PR program, where the IAC does not meet de agreed turnaround time, and an exchange part is

offered instead. So, in this case the PE program forms a buffer for problems occurring at the IAC repair shops. About 9.5% of the repairs over the last thirteen years have been filled by performance exchanges. A flow-diagram of how information and physical goods flow through the supply chain can be observed in Figure 1. First, the customer sends an unserviceable part to the repair shop of the IAC. Then, when the repair shop fails to repair the parts within the agreed turnaround time, the repair shop alerts the operational planners. They will offer a performance exchange to the respective customer, which comes from the inventory pool located at the IAC. The repair shop will continue the repair of the part, and when the part is finished, the inventory pool located at the IAC is replenished.

Another scenario when the PE program is used, is when the customer requires delivery before the agreed turnaround time is reached, for example in an Aircraft On Ground (AOG) situation. For an additional fee, the IAC might decide to help the customer and send an exchange part from the inventory pool. The flow of information and goods is the same as in the previous scenario, except for extra information coming from the customer to inform the IAC about its problems.

1.3 Motivation for research

The incentive for the IAC to initialize this Thesis, is a company review by Topan, Eruguz, Ma, van der Heijden, and Dekker (2019), who reviewed operational spare parts service logistics in service control towers. From this review, the importance of alignment between tactical and operational decision making became apparent, as it might lead to higher service levels (Topan et al., 2019). For the tactical planning, previous Master Thesis assignments have been used to find improvements. For the operational planning, on the other hand, this is not the case. Decision making is solely to resolve problems and based on experience of the operational planners.

As the parts involved in the CMA programs are of key importance for the IACs customers business, the IAC has strict service contracts with its customers. In these service contracts, several affairs are agreed upon, such as maximum turnaround times, service levels, fill-rates, and response times. To meet these agreements, the IAC currently has two full-time operational planners. These try to maintain component availability in the inventory pool and solve problems that arise in the supply network. Furthermore, they judge repair quotes by external repair shops, and make make/buy decisions. To assist in this process, the IAC has developed a Service Control Tower (SCT), which aims to provide insight into the current status of the supply chain and the performance concerning the customers in real-time. The SCT is specifically designed for the CMA programs, mainly for managing the inventory level of the pool.

This SCT currently is purely descriptive, meaning that it only is a visualization tool for the current state of the network. Furthermore, the SCT only considers information of the IAC, as customers and suppliers are reluctant to share data with the IAC. Because the service control tower and the physical stock are the only ways the supply chain is monitored, the operational planners are currently only notified once a stock-out occurs. This has two implications, the first being that as the stock-out already has occurred, the planner has to come up with a quick solution to meet the contract requirements. These interventions are often more expensive than the interventions that require earlier execution. The second implication is that, as the operational planners are always notified late, the service levels are lower than they can be with the invested capital. Another problem further amplifying the costs incurred by stockouts is that the operational planners have difficulty with seeing the impact of their decisions. This causes them to use interventions that are unnecessarily expensive or ineffective.

Furthermore, the components involved in the CMA programs are expensive. Therefore, management is reluctant to agree to the acquisition of new parts. The operational planners indicated that this is the main cause of problems, as often demand simply exceeds supply. However, they have to make the best out of this situation, with the means they have available. Thus, identifying potential problems in advance would enable the operational planners to do this.

1.4 Problem statement

I defined the problem as follows:

“The Independent Aerospace Company misses a proactive monitoring tool, causing them to incur higher operational costs than necessary and with a lower service level than achievable with the current capital investment.”

The focus of the IAC should be maximizing the availability of spare parts for the customers while minimizing the operational costs incurred to achieve this goal. The current control tower misses the key component which is the ability to send alerts (or warnings) when stock-outs, or other problematic situations, might be occurring in the near future.

1.5 Research Questions & Approach

The main research question of my thesis is:

“How can the Independent Aerospace Company construct a proactive alerting tool, which automatically recognizes and prioritizes potential problematic situations and notifies the operational planners?”

To answer this question, I developed several sub (research) questions, which need to be answered to answer the main research question and solve the problem of the Independent Aerospace Company. In the remainder of this chapter, I will present my five sub-questions and motivate their relevance.

“What is the current way of working at the Independent Aerospace Company and what are possible events leading to backorders? How are these events currently identified and what is their impact?”

The relevance of this question lies in the importance of understanding the current processes and way of working, to determine when backorders are likely to occur. When these situations are identified, I can focus my further research on these specific situations. I will do this by mapping out the network of the IAC to clarify the way of working and to understand the flow of goods through the supply chain. Furthermore, I will perform interviews with the operational planners to determine for which problems they receive alerts, how these alerts are received, and how relevant these alerts are. Finally, the current performance of the IAC is analysed, to further clarify the need for improvement.

“Which methods are described in literature for using artificial intelligence for classification. Which of those methods are the most suitable for the previously defined problems at the Independent Aerospace Company and which steps are described to apply them?”

When I have identified what information, I need to identify potential stock-outs, I need to come up with a decision model which can use this information to generate alerts and provide possible interventions. Furthermore, I will have to modify the data in such a way that it can be used as input for a model which generates operational alerts.

“How can machine learning algorithms, as described in the literature, be used to create a model which indicates parts most urgently require attention from the operational planners?”

By answering this research question, I will come up with a model which generates alerts, but also recognizes which alerts are more important than others. As Topan et al. (2019) found, many organizations have the generation of alerts in place, but do not use this system as too many unnecessary alerts are generated. Therefore, management of alerts is important, as people quickly lose their trust in a model if it is overly sensitive and generates too many alerts. On the other hand, if too many problematic situations are ignored, people lose trust in the model and will be reluctant to use it.

“How are the different components of the alert generating tool performing, and what is the performance of the alert generating tool as a whole?”

To validate the added value of making the alert generating tool pro-active, I need to compare the performance before and after my interventions. Based on these results, I can write a conclusion and provide recommendations to the Independent Aerospace Company and for further research.

“How can The Independent Aerospace Company leverage a pro-active alert generating model in practice and which additional steps are required to achieve this?”

The final chapter will answer the sixth question, and in this way provide advice to the Independent Aerospace Company. So, it helps them to implement the product I created, and use it to its full potential. A user-interface is conducted to allow the operational planners to use the alerting tool and furthermore I will provide recommendations to the IAC on how to further improve the alerting tool.

1.6 Scope

To limit the size of my thesis, and to make sure I can execute it within the given time-frame, I need to determine which things I will be focussing on, and which I will consider being given. Furthermore, some preferences have been indicated by the IAC. These decisions and preferences are briefly discussed in the remainder of this chapter.

1.6.1 Use of AI in Service Control Towers

Historically, the IAC has used several statistical methods to automate and optimize parts of the tactical and operational planning in the CMA programs. However, a large part of the daily operations requires human reasoning, making the use of basic algorithms ineffective. Therefore, the IAC would like to investigate the use of Artificial Intelligence in the operational planning. For this purpose, my company supervisor at the IAC has started his PhD to gain knowledge on this topic. His research investigates the use of artificial intelligence in a supply chain, more specifically in a Service Control Tower. Hence, the preferred solution uses artificial intelligence instead of the conventionally used statistical models. Thus, in my thesis, a solution incorporating artificial intelligence is preferred over statistical models, even though the performance might currently be underwhelming.

1.6.2 Operational level

In this thesis, I will only focus on the operational level of the Service Control Tower. The operational time scope embodies short term occurrences, with a maximum time scope of a few weeks. Furthermore, this means that the base stock levels are considered to be given and cannot be changed. So, if tactical decisions cause operational problems, I will discuss this in my recommendations, but in the solution, I will try to make the best out of a bad situation.

1.6.3 Fixed demand forecasting

The demand for the FE program has been intensively studied by the tactical planners at the IAC. These forecasting methods are based on historical removal/flight hour rates and have proven to be effective. Thus, these forecasts can be assumed reliable. So, if the FE demand is of importance for a proactive alert generating model, the existing forecasting methods will be used. If the IAC in the future decides that the forecasting methods have insufficient performance, the new forecasting methods need to be implemented in my final solution to ensure coherence between the operational and tactical decisions.

2 Present situation

In this chapter, I will provide a schematic overview of the business processes involved in the CMA programs. Moreover, I will provide the results of my data analysis, to see which disruptions occur in the supply chain and how often these occur. Finally, I will discuss the current activities of the operational planners, and what role the current SCT is playing.

In section 2.1 I will discuss several special situations in the Forward Exchange program. Then, in section 2.2, I will discuss the increasing demand for the Performance Exchange program. Next, in section 2.3 the situations that are the main causes of disruptions are elaborated. The main performance indicators for the performance of the CMA program, are discussed in section 2.4. Section 2.5, briefly discusses which methods the operational planners have at hand to resolve such situations and section 2.6 explains how alerts currently are received by the operational planners. Finally, in section 2.7 I will briefly discuss which data is available to base a solution on and in section 2.8 I will provide the main conclusions of this chapter.

2.1 Special situations in Forward Exchange program

In section 0, basic situations of the exchange programs have been described. In reality, there are a lot more variations to the CMA programs. These are ‘Lease parts at customer’, ‘Beyond economical repair’ and ‘Inaccurate Customer Diagnostics’, which will concisely be discussed in the remainder of this chapter.

2.1.1 Lease parts at customer

The first extension is that about 1% of the customers have some located on their site, called lease parts. When they require a spare part, they will take it from their pool, which is then restocked from the inventory pool managed by the IAC. Leasing of parts is mostly done for SKUs that are critical to the working of an aircraft, as it can result in a high reduction of lead times. For each part leased by the customer, an additional fee is paid, so the customer has to decide if they think it is worth it, which SKUs they want to lease, and how many they want to lease.

A schematic overview of the process is found in Figure 2. Once such a customer requires a part, they will obtain it from the inventory pool located at their site and inform the IAC on this. They will then swap the part with the broken part in one of their aircraft and send this broken part to a repair shop. the IAC replenishes the inventory located at the customer with an item from its own inventory pool.

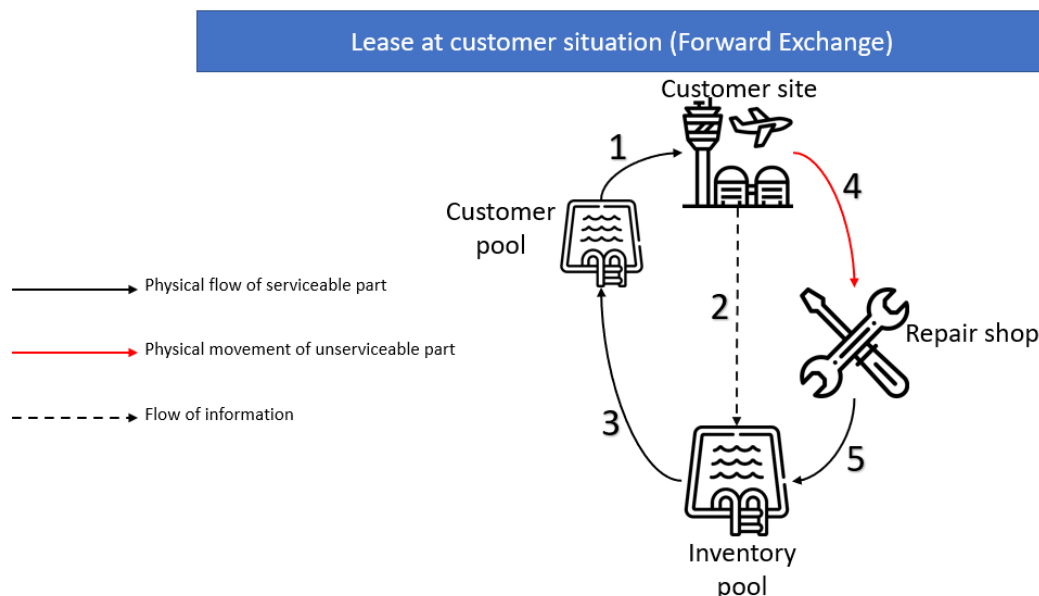


Figure 2 Forward Exchange with lease illustration

2.1.2 Irrecoverable parts

The second extension concerns parts that are irrecoverable. Sometimes, it is not economically worthwhile, or technically impossible to repair a part. Historically, this has happened for about 3.5% of all orders since 2007. When such a situation occurs, the part is sent back to the CMA inventory pool and the operational planners have three options. They can either cannibalize the part, scrap the part, or (temporarily) store the part. Unfortunately, no data is collected on this decision, so it remains unknown how often each option is used.

When an item is cannibalized, the item is torn apart in such a way that the smaller components from which the part is made can be used as scrap parts for other parts. If already a surplus of scrap parts is available, the planners will likely decide to scrap the parts, which requires less time from the repair shop, however it causes complete loss of the part. Finally, the operational planners might decide that the repair is currently economically unviable but might become viable in the future, in which case the part is temporarily stored.

Another decision the operational planners now have to make, is whether they want to replace the unit. Generally, this is the case, however the operational planners indicated that sometimes this way is also used to reduce the number of parts in inventory, for example when a type of aircraft is no longer used. The economic value of the part is then set to €0, and every returning repair is scrapped to reduce inventory. If the operational planners decide that the part should be replaced, the part is procured from an available supplier, through a tender offer. Figure 3 shows a schematic overview of this situation.

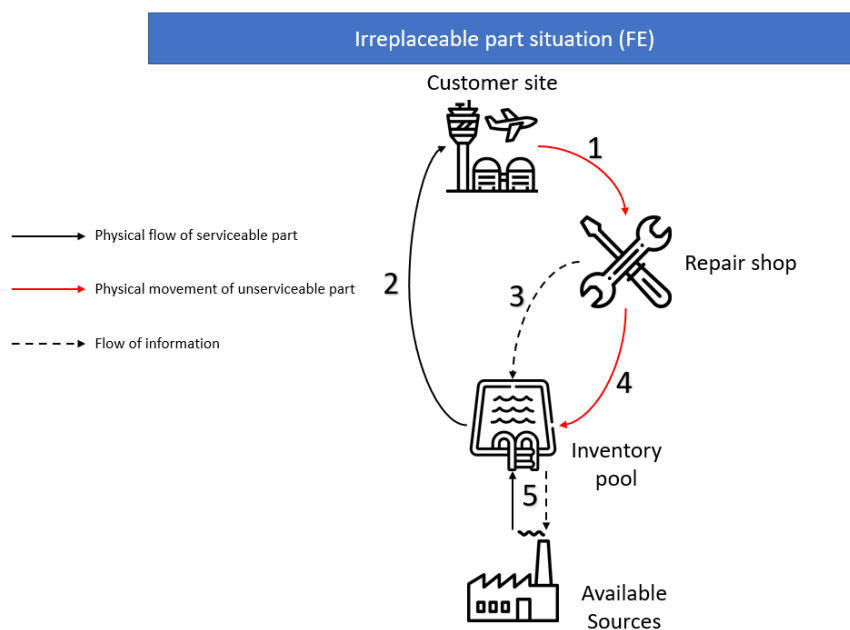


Figure 3 Beyond economical repair part (FE Program)

2.1.3 Inaccurate customer diagnostics

The final extension of the FE program is regarding the behaviour of the customer. Customers might wrongly diagnose a part as unserviceable. This can be caused by a mistake of the customers' diagnostic team but could also be caused by unclear reasons where a part seems to be faulty, while in other tests it appears to be working. Moreover, an aircraft might be grounded, and the responsible mechanic orders several parts, to ensure delivery of the required part. Due to this behaviour, occasionally serviceable parts are sent to the repair shop, where during the diagnosis no failures are found. Or customers might send serviceable parts back to the pool, as they are not required to repair their aircraft. About 3.5% of all orders is returned as is (RAI).

Figure 4 shows a schematic overview of this situation, again with the numbers representing the order in which the flows of either goods or information take place. The number ‘3’ is given to three different streams, as these are the possible flows due to inaccurate customer diagnostics. Which are:

- Customer sends an unserviceable part to the repair shop (red line to repair shop)
- Customers sends a serviceable part to the repair shop (black line to repair shop)
- Customer sends back a serviceable part to the CMA inventory pool

These three possibilities are all indicated with the number “3”, in, which shows a schematic overview of this situation.

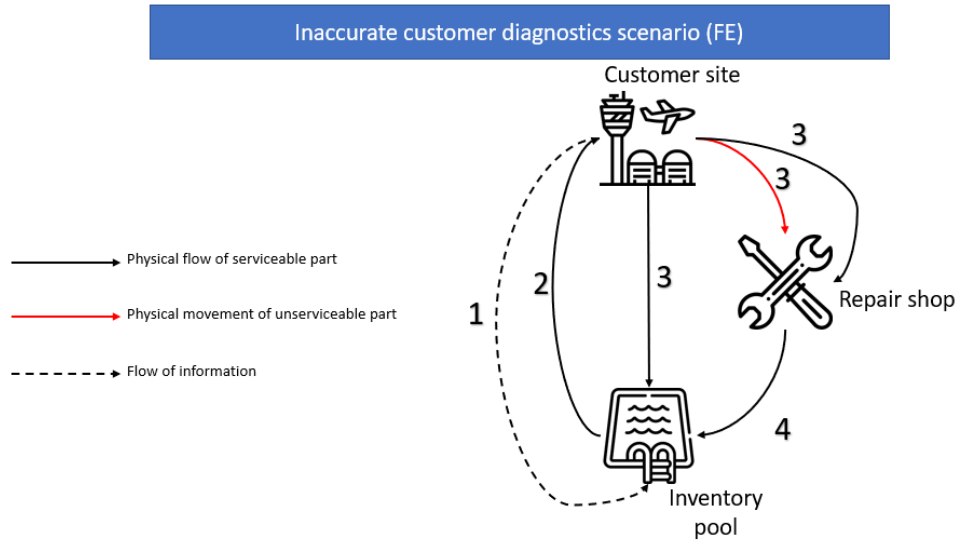


Figure 4 Inaccurate customer diagnostics (FE Program)

2.2 Increasing demand of Performance Exchange Program

While the CMA program originally only existed of the FE program, during the last few years, the number of performance exchanges has rapidly increased. Initially, the PE program was only used for exceptional cases, and only a fraction of the customers was willing to pay an additional fee to have a performance guarantee. However, since 2017, less customers make use of the FE program, and instead keep their own inventory while outsourcing the repairs, with performance guarantee, to the IAC.

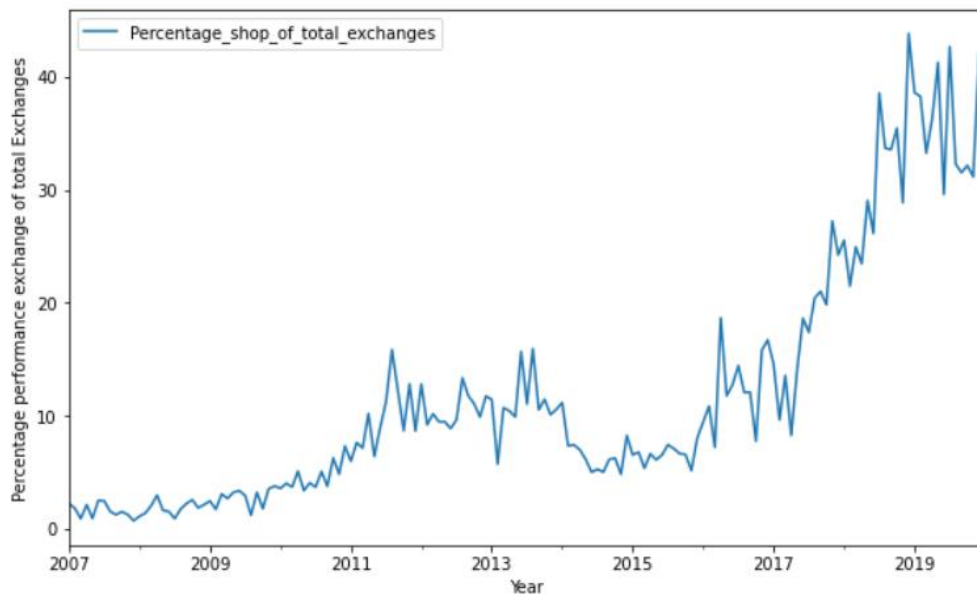


Figure 5 Percentage of performance exchanges of the total number of exchanges

Figure 5 shows the percentage of performance exchanges, with respect to the total number of exchanges. It is clear that this percentage has increased drastically, and thus, that the impact on the CMA inventory pool has been increasing. Currently, almost 40% of the exchanges performed monthly are exchanges for the PE program.

This increase in performance exchanges has two reasons. The first, is a changing business landscape. More carriers keep their own stock, and only outsource the repairing of parts, so the number of FE contracts has dropped. The second reason is the increasingly competitive business environment, resulting in more competitive agreements on turnaround times. The operational planners indicated that the sales department offer shorter turnaround times than are generally achieved by the repair shop. Which in turn decreases the likelihood that the repair shop will repair the part in time, and thus increases the likelihood on performance exchanges.

2.3 Situations causing disruptions

2.3.1 Direct Backorders

This situation is the most critical, as backorders have a direct impact on the performance of the CMA program. Whenever these occur, the operational planners will take extreme measures to resolve the problem as quickly as possible. This often means leasing an item from another supplier, or, if a part is being repaired internally, ask the repair shop to immediately repair that part.

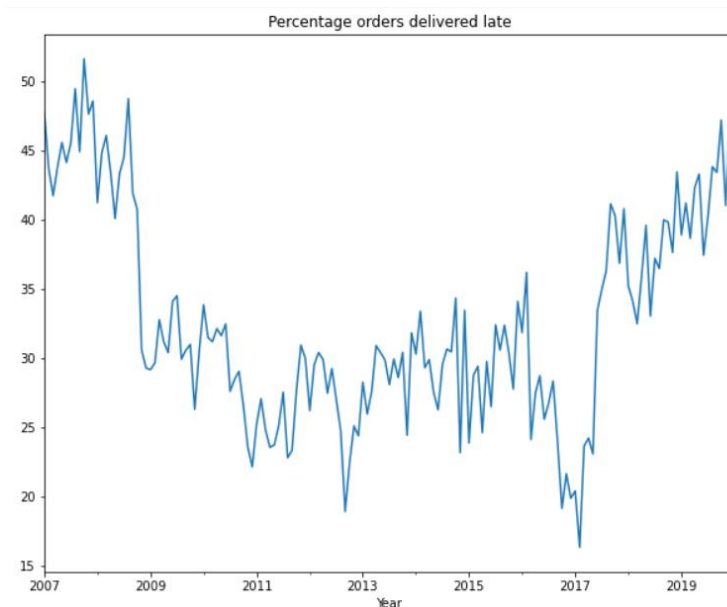


Figure 6 Percentage of orders delivered late (monthly)

Figure 6 shows the percentage of orders that is delivered late, aggregated per month. Although the performance has increased since 2007, it is still rather poor. Considering all orders since 2007, about 36% has been delivered late.

Contracts of the IAC are on an Ex Works basis, which means that the order is considered 'delivered' as soon as the IAC hand the order over to the carrier. Furthermore, no service differentiation is applied by the IAC, thus every situation in which the IAC has delivered late, can be considered a situation with backorders, as they were unable to deliver.

An important note to this section, is that these metrics are based on internal business rules. These business rules are set by the IAC, to aim for operational excellence, and thus are strict. From an external view, the performance is a lot better, as often the internal due date is several days before the external due date. Furthermore, late deliveries are always discussed with the customer, who regularly indicates that delivering a few days late is no problem.

2.3.2 No stock on hand

Although no stock on hand does not directly mean that there are performance problems, the likelihood that problems are occurring soon is high. In the interview, the operational planner mentioned that he considered no on hand stock almost as worse as having direct backorders. The only difference is how extreme the measures are that are taken to resolve the situation, as there is no immediate performance loss. Unfortunately, there is no documentation on how often no stock on hand occurs. However, the operational planners mentioned that much of their time is spent on resolving such situations.

2.3.3 High turnaround time (repair shop)

Currently, when repair shops are experiencing high turnaround times, it is only recognised when the operational planners are reviewing other problems. Upon further investigation, they find the sources of these problems to be the turnaround times of the repair shops. When repair shops take too long to repair parts of the CMA inventory pool, the on-hand inventory will gradually drop, eventually causing back orders.

Whenever this occurs, the operational planners will contact internal repair shops to prioritize parts for which the current on hand stock is low. However, external repair shops rarely want to do this, as they have more customers who also have contracted turnaround times and changing up the repairing order might cause them to breach other contracts.

The shops that are compared here, all have fulfilled at least 5% of the orders since 2007 and the last repair has been registered after 01-01-2018. This filter is applied as the IAC uses many shops, but some shops only fulfilled some incidental orders, making statistical assumptions very inaccurate, or have gotten out of business. Furthermore, I made a distinction between internal and external shops, as these shops have different ways of working and can be managed in different ways. From the pareto diagram presented in Figure 7, it can be observed that about 35 (10%) external repair shops performed 70% of the orders.

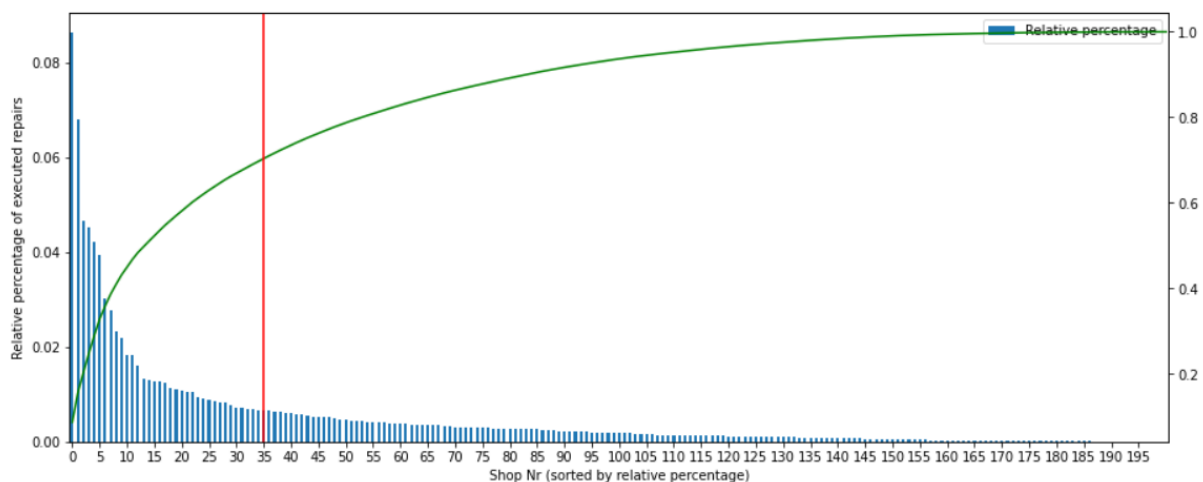


Figure 7 Pareto diagram of external repair shops

The performance of the 10% most commonly used shops varies a lot. The best performing shop only delivers 15% of the order too late. However, the worst performing shop delivered 85% of the orders too late and is 39 days overdue on average. These disruptions in the supply to inventory pool, lower the inventory availability. Therefore, the operational planners will have to perform more interventions to prevent drops in customer satisfaction or breaches of their contracts.

2.3.4 High customer return time

The final common situation which might cause problems in the supply chain, is that customers are late with returning the (broken) part, when they received an exchange part by the forward exchange program. Similar to the throughput time of the repair shops, when customers are late with returning their parts, the number of parts on hand is negatively affected. This problem is currently only identified when problems such as backorders are arising, and further analysis by the operational planners is conducted.

There are clauses in the contracts stating that the customer has to return the replaced part within a certain timeframe. However, many customers fail to meet these terms. The operational planners state that this is due to the IAC not issuing penalties for long return times, making the customers less eager to send their units back to the IAC. Currently, about half (49.1%) of the parts is returned late by customers, with an average return performance of five days after the due date. Some of the worst performing customers return the cores late in more than 75% of the cases and have an average return performance of over 40 days late.

2.4 Performance Indicators

As described in the previous section, there are four situations which are likely to cause disruptions in the supply chain of the IAC. However, while it is inconvenient when a repair shop has longer turnaround times, or a customer is late with the returning of a core unit, it is not directly problematic. Situations are only problematic for the performance of the IAC, if they cause the IAC to be unable to deliver parts in time, as late deliveries will eventually lead to breach of contracts.

While in most retail supply chains, sales are lost when no stock is available when an order is placed, as the customer can easily go to another retail store. The same situation does not hold for the IAC, as customers pay a fee to use the FE program, and demand (in time) delivery from the IAC. So, when the IAC fails to meet the initial order, the order will be added to the backlog, and is fulfilled as soon as a part becomes available.

As the time it takes the IAC to fulfil the order is of importance, the most suitable performance indicator is the number of backorders. Backorders are defined as ‘Orders for a good or service that cannot be filled at the current time due to a lack of available supply’ (Kenton, 2019). The aim of the operational planners is to minimize backorders, so this is the main performance indicator.

2.5 Available interventions

To prevent, or resolve, backorder situations, the operational planners have several available interventions. These interventions are ‘Doing nothing’, ‘Asking a repair shop to prioritize an order’, ‘Asking a repair shop to drop ship an order’, ‘Lateral transshipments’ and ‘Leasing a part’, or a combination of multiple interventions. No data is available on how much each intervention costs, how much time is won by performing the intervention, or how long performing an intervention takes. Furthermore, these properties are very item specific, as some parts require almost 40 man-hours to repair, while other only take a few hours. Therefore, the properties given for each intervention are only an indication.

2.5.1 Doing nothing

The simplest intervention is ‘Doing nothing’, which as the name implies means the operational planner will analyse the situation and decides that no intervention is required. Currently this happens only very rarely, as the alerts are generated reactive, meaning in almost every case an intervention is required due to the urgency of the alert. When the operational planner decides to do nothing, it is possible that at a later point in time, another intervention is used. For example, the operational planner expects a part to

return within several days, so demand can be met, but it turns out that the part is delayed. Now, an intervention is actually required to prevent backorders.

2.5.2 Asking a repair shop to prioritize an order

When an operational planner is 'Asking a repair shop to prioritize an order', the operational planner will ask the executing repair shop to move an order up in the repair sequence. For internal shops, this intervention is frequently used, for external shops however, the repair shops often do not want to do this. External shops have contracts defining the return times for repairs and changing the sequence in favour of the IAC might cause them to breach other contracts. For this reason, an extra fee is paid whenever the IAC wants an external repair shop to prioritize their order.

For internal repair shops, this intervention is free and (depending on the required repair time of a part) will often result in same-day delivery. For external repair shops a fee of around €500-€1000 is paid to prioritize the order. Moreover, it takes several days before the part is ready to be shipped to the IAC, after 2-3 days of shipping the part is finally available for the IAC to fulfil demand or restock their warehouse.

2.5.3 Asking a repair shop to drop ship an order

When there is no on hand stock, and an forward exchange order comes in for a part, the operational planners can ask a repair shop that is currently processing that part, to deliver it directly to the customer, instead of the pool of the IAC. This shortens the delivery time by a few days, as less shipping and processing is required. For this intervention, rarely any costs are incurred, or these are neglectable as the IAC does not have to ship the part by themselves anymore, resulting in cost savings.

2.5.4 Lateral transshipments

The next intervention that is discussed is the lateral transshipment. The operational planners can relocate a functioning component from one to another location. There are three different options for relocating components, which are from the commercial warehouse, from lease stock, and from the quarantine warehouse.

The commercial warehouse is a different warehouse than the warehouse where the CMA stock is located. As for the FE and PE programs, strict agreements are made, sometimes the commercial stock is used for fulfilling the demand of the CMA programs. Once a repair for that particular part is finished, the part is sent back to the commercial warehouse instead of the commercial warehouse.

The second option is the use of lease stock of customers. These customers are then asked to make their lease stock available for fulfilment of orders of other customers. Generally, this option is disliked by the customer that owns the lease stock, as they pay a large fee to have these lease parts. Furthermore, this option is only viable when the customer is located close to the customer with a malfunctioning part, as otherwise long lead times nullify the effect.

The final option is using the quarantine warehouse. The quarantine warehouse is the warehouse where unserviceable parts are stored, that yet have to be repaired, but the repair did not seem worthwhile yet. Once backorders occur, the operational planner can check if the part is available in the quarantine warehouse and decide if it is worthwhile to execute the repair for this part to fulfil the demand.

2.5.5 Vendor exchange

The final option for the operational planners is using the exchange programs of competitors. However, as the IAC is not under contract for such exchanges, the costs of this intervention is remarkably high (about 1/3-1/4 of purchasing price). Therefore, this intervention is only used as a last resort to meet contract requirements.

2.5.6 Other options

Next to the aforementioned interventions, operational planners have a few other options that are cannot be used to directly solve problems but can be used to prevent problems from happening in the future. Therefore, they are not considered interventions, as using them to resolve backorders is either impossible or impractical. The first option is asking a customer to send back core units. As seen in chapter 0, some customers are slow with returning their core units. Reminding them to send them back can increase the number of parts that is kept in stock (after they have been repaired). The second option is procuring an item from the market. As this is a process which requires quite a lot of time, resolving backorders with this option is impractical. However, if a part is often backordered, even though the supply chain is running smoothly, it might be that demand simply exceeds supply, and therefore an extra part should be acquired.

2.5.7 Combination of interventions

Finally, the operational planners indicated on a regular basis, combinations of interventions are used. Several reasons have been demonstrated by the operational planners why interventions are combined. For example, increasing the chance of success, by executing multiple interventions, the chance of resolving or preventing backorders is increased. Another example is asking a repair shop to prioritize an order and ask them to dropship the order directly to the customer.

2.5.8 Costs of interventions

As mentioned before, the costs for performing interventions, increase when the response time is shorter.

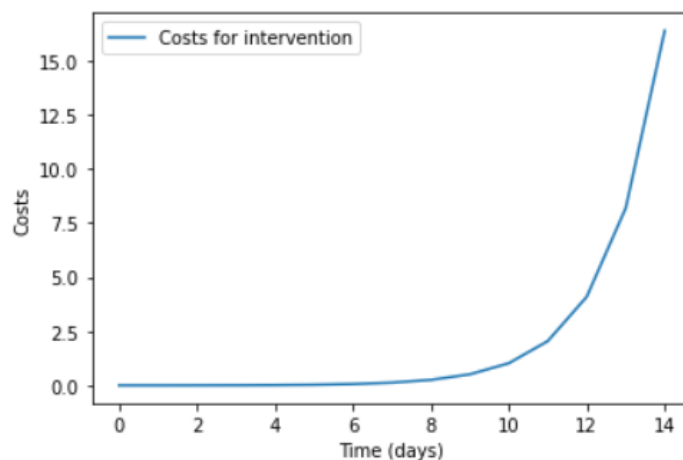


Figure 8 Exponentially rising costs for interventions

This is due to the limited number of interventions that remains available on a short term. When a problem is discovered late, only the more rigorous interventions remain available. Figure 8 shows a schematic overview of how costs exponentially increase from time (days) 0 till 14, where day 14 is the day on which an item is backordered. From this figure it is amazingly simple to observe that early executed interventions are cheaper than just in time interventions. But to be able to know that an intervention is required, the operational planner needs to be aware of the problem.

2.6 Means of receiving alerts

The operational planners currently receive alerts through the ERP system of the Independent Aerospace Company. This ERP system generates alerts when backorders occur. Furthermore, a reporting tool has been programmed to evaluate the current state of the supply chain. This tool considers the current (on hand) stock, the open requests, the criticality of a part and based on these criteria gives parts a priority rating. In this tool past performance is only incorporated by reporting the number of backorders for a specific part over the last three years. This tool is executed manually daily and produces about two alerts on a weekly basis.

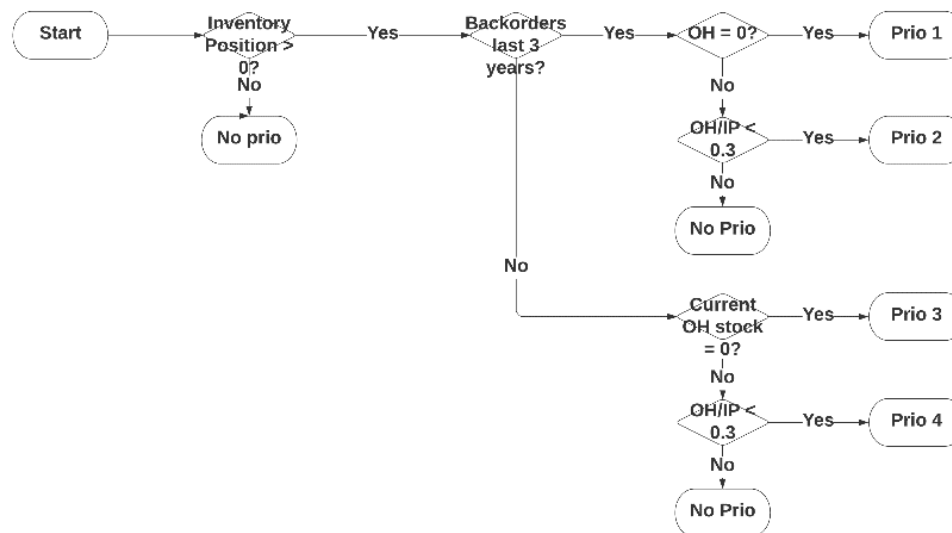


Figure 9 Flowchart priority selection

Figure 9 provides a flowchart in which the prioritization logic can be observed in more detail. Note that the tool only considers historical backorders and the current state, and no predictions are incorporated. The only way the chance of backorders is incorporated, is with the ratio between OH and IP that is calculated. However, this ratio completely ignores the forecasting methods the IAC has in place. Furthermore, the price, importance, and contracted delivery times are not considered. Although the alerts generated by this tool works correctly for a large portion of the different parts, the operational planners indicated that based on their experience, they often change the priority listing. These situations often incorporate problematic parts, which have high failure rates, unreliable customers, and specific knowledge of an aircraft (e.g. parts that are often replaced together). Furthermore, the operational planners indicated that the tool is not proactive, as it only lists parts that are currently encountering problems, and not parts that have a high probability to encounter problems in the near future.

Furthermore, the performance exchanges are not considered in this alerting tool. As the demand of performance exchanges is rapidly increasing, this leads to underestimation of the total pool demand. Finally, the returning repairs are not considered in this tool. When parts are returning, future demand might be covered with parts that are currently being repaired. Therefore, these should be considered in the alerting tool.

2.6.1 What-if scenarios

Currently the tool also lacks the flexibility to evaluate different scenarios. For example, the operational planner might have asked an internal repair shop to prioritize an order, for an item that has zero on hand stock and has experienced backorders in the past three years. The repair shop agreed to prioritize and promises delivery on the next day. According to the current alerting tool, the part should still receive a 'Priority 1', as the on-hand stock is still zero. However, the operational planner knows that a part will

be received the next day. Therefore, it is more relevant to evaluate the situation where the on-hand stock is one, to see if more interventions are required.

Moreover, now, and then an external customer wants to make use of the forward exchange program. As these customers often require these parts on a short notice, high premiums are paid if the IAC decides to deliver the part. In this situation, the operational planner would like to evaluate if temporarily reducing the on-hand stock is likely to cause trouble. Based on how likely the exchange will cause trouble, and the premium the customer is willing to pay, a decision can be made if the risk is worth the premium or not.

2.7 Availability of Data

In this chapter the available data to base the solution upon will be described. Most of the data comes from a data set drawn from the ERP-system of the IAC. This data set contains historical data on exchanges and repairs performed by the IAC and will be the main input for the tools created to predict problematic situations. The technical planner indicated that since 2007 the data has become a lot cleaner, as since then data entry rules were enforced more strongly. When only data entered after 2007 is included, the data set consists of 272,163 instances (order lines). The relevant columns are described in Table 1, a more detailed description of these fields can be found in APPENDIX A: Detailed Description of Available Data.

Column name	Feature name	Data type	% Missing values
CUSTNAME	Customer name	Categorical	0
PRIORITY	Priority indicator	Categorical	0
TRANS_TYPE	Transaction type	Categorical	0
EXCH_TYPE	Exchange type	Categorical	0
PARTNUMBER	Unique part number	Categorical	0
PRODUCT_ID	Unique product identifier	Categorical	0
KEYWORD	Product description keyword	Categorical	0
COND	Product Condition	Categorical	0
LINE_ADDED	Repair shop entry date	Datetime	0
DELIVERED	Delivery date	Datetime	0
DEL_DUE_DATE	Delivery due date	Datetime	0.14%
CORE_RCVD	Returning part receive date	Datetime	29.57%
STOCK_UPDATED	Warehouse stock update	Datetime	5.43%
WORK_PERF	Work performed	Categorical	0
SHOP	Repair shop number	Categorical	0
PROD_GROUP	Product group keyword	Categorical	0
MAINT_TAT_AGR	Agreed turnaround time	Numerical	89.56%

Table 1 Database overview

Upon the first exploration of the data, quite a lot of noise was experienced. First, quite some duplicates are found, so multiple lines in the database represent the same order. Next, extreme values were found for, for example, the realised TAT. For this metric, values over 700 are present, indicating that outliers are persistent in the data set. Furthermore, data entry errors were experienced, e.g. in the work performed column 'MOD' and 'MOR' were used interchangeably, while they both mean 'Repair plus modification'. So, before the data can be used, the data should be thoroughly cleaned.

2.7.1 Unavailability of Data

Unfortunately, there is also a lot of data unavailable, meaning that alternative ways must be found to gather this information. Firstly, the used interventions are not documented as such, so in the dataset it cannot be traced back for which orders an intervention was required. If this data were available, situations with high risk of backorders could be predicted by identifying which situations historically led to interventions. If a similar situation would occur, this could lead to an alert.

Furthermore, both suppliers and customers are reluctant in sharing data. So, data regarding the status of an external repair is unavailable. For internal repairs, the data is available, but of low value. For internal repairs, most of the total turnaround time comes from the queue of parts that are waiting for a repair. Once the diagnosis for a part is done, it is most of the time finished on the same day. Only in case of unavailability of scrap parts or other exceptions, the repair is not directly finished. So, the status updates within a repair shop is often meaningless and thus cannot be used for status updates of repairs.

2.8 Conclusion

In this chapter, the present way of working and the performance of the Component Maintenance & Availability (CMA) program at the Independent Aerospace Company (IAC) are discussed. The CMA program can be divided into two subprograms, which are the Forward Exchange (FE) program, and the Performance Exchange (PE) program. The inventory pool for the CMA program is shared by the PE and the FE program. At present, the IAC is experiencing a lot of backorders (36% of orders is delivered late), which causes the operational planners to use expensive interventions to resolve these backorder situations. So, proactively generating alerts for stock keeping units (SKUs) that will potentially face backorders would allow the operational planners to prevent backorders from occurring soon.

Currently, a reactive alert generating model is in place, which only considers historical backorders, the present inventory level, and the inventory position of each Stock Keeping Unit (SKU). If the inventory level for a SKU is too low with respect to the inventory position, an alert is generated. While previously most claims of inventory came from the FE program, nowadays an increasing fraction of the claims comes from performance exchanges. However, these are not incorporated in the existing alert generating model. Furthermore, the existing alert generating model does not incorporate parts that are in repair, which will return to the CMA inventory pool upon finishing.

So, to make the alert generating model more proactive, means must be found to incorporate the orders for the inventory pool caused by performance exchanges. Furthermore, a way should be found to incorporate returning repairs in the alert generating model, as neglecting these would lead to underestimation of the supply of the inventory pool. Finally, a way should be found to use these components to estimate the chance of backorders, as this is the most important performance indicator for the performance of the IAC.

Unfortunately, the available data only consists of historical performance on different repair and forward exchange orders. So, no data is available on the states of a repair, nor historical interventions which took place. The performance on the historical repair orders can be used to predict which repair will not be finished in time, and thus results in an order for the CMA inventory pool. Furthermore, historical turnaround times may provide information on the returning repairs, which will replenish the CMA inventory pool.

In the next chapter, the following topics are discussed, based on a review of the available literature. First, how a classification model can be constructed, which is able to identify which repairs will finish in time, and which will be finished late. Secondly, how historical turnaround times can be used to model the returning repairs process. Thirdly, how these different stochastic components can be incorporated in a model, which is able to approximate the chance of backorders for each SKU.

3 Literature review

In this chapter I will conduct a literature review to identify which steps are required to construct a more proactive alert generating model, which incorporates the performance exchange orders and the returning repairs. To achieve this, the following three sub-questions are defined:

- *What artificial intelligence algorithms are described in literature, which can be used to identify which repair will be finished in time, and which repair will be late?*
- *What methods are available to approximate the distribution of historical turnaround times?*
- *How can multiple stochastic variables be used to approximate the chance of backorders?*

The first question will be the most challenging, and the answer to this question forms the largest portion of this chapter. Section 3.1 gives an overview of available machine learning methods and elaborates what machine learning is. The remaining sections answering the first sub-questions follow the seven steps to successfully conduct a machine learning project, as defined by Chollet (2018). These steps are: data collection, data preparation (3.2), model selection (3.3), training the model (3.4), evaluation of the model (3.5), parameter tuning (3.6), making predictions based on the model. As data collection is already conducted prior to the start of this research, this step is not included in the literature review. The second question is addressed in section 3.7, and the third question is elaborated in section 3.8. Finally, in section 3.9 the main findings of this chapter are discussed.

3.1 What is machine learning?

The first question to answer, is what is machine learning. El Naqa and Murphy (2015) define Machine Learning as ‘an evolving branch of computational algorithms that are designed to emulate human intelligence by learning from the surrounding environment’. So, it be an application that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. Well-known examples of the use of ML are the recommendation systems used by Netflix or Bol.com. A figure created by MIT gives a nice visualisation on how Machine Learning differs from traditional programming.

Roughly said, there are three forms of Machine Learning, these are: Supervised Learning, Unsupervised Learning, and Reinforcement learning. Each of these types has its own requirements regarding the required nature of the problem you are trying to tackle, the amount of data that is available, and finally the nature of the data that is available.

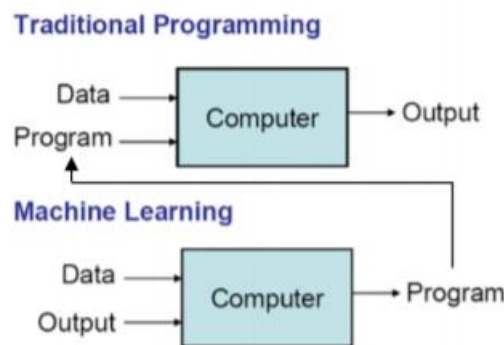


Figure 10 Traditional programming vs machine learning

3.1.1 Types of machine learning

3.1.1.1 Supervised Learning

The most prevalent kind of ML is Supervised learning, where data is labelled in order to tell the machine exactly what types of patterns it should look for (Fumo, 2017). In essence, the machine is learned to, based on input (X), predict the outcome (Y). To be able to learn the machine how to predict Y, we are required to have historical data for which we now know the outcome, so we have a means to provide

feedback to the model on how it did. In short, supervised learning is trying to ‘model relationships and dependencies between the target prediction output and the input features’ (Fumo, 2017).

3.1.1.2 Unsupervised Learning

Whereas in supervised learning, data is labelled, for unsupervised learning the data is presented to the machine as unlabelled. The machine will then classify groups of data, based on the data, without a direction on what the data means. This method of machine learning is most suitable for classifying groups, which are not easily identified by humans. Unsupervised learning has been used by the Dutch government to identify tax-fraud cases, as the model was able to identify ‘odd cases’ from the dataset it was provided.

3.1.1.3 Reinforcement Learning

Reinforcement learning is a form of learning in which trial and error plays a large role. The machine will perform an action and will receive feedback on how good this action was. For next time, when it is confronted with a similar situation, it will know if its previous action was good or not and whether it should be repeated. The goal of reinforcement learning is optimizing the outcome, based on a given state, by selecting the best decision for that state. This kind of machine learning is mostly used in the operations research, where a true optimal solution exists, but is unlikely to be found, due to the size of the problem. Reinforcement learning can be an effective way of finding a solution that is good, without fully exploring every option.

3.1.2 Conclusion

Both for identifying performance exchanges and predicting repair throughput times, supervised learning is the most suitable approach. We have a clear variable which we want to explain, with the data which is already available. Therefore, in the remainder of this literature review, I will only focus on supervised machine learning methods.

3.2 Data preparation

Data preparation forms the second step of the process. It is highly dependent of the problem at hand (Bohanec, Borštnar, & Robnik-Šikonja, 2016), but should be handled with care. If data preparation is not conducted properly, the model is likely to have bad performance (Garbage in, garbage out). Therefore, about 60% of the time for solving such a problem, is spent on data cleaning and organising (Press, 2016). During the data preparation, data is cleaned, transformed, and validated to form a high-quality training set.

3.2.1 Data cleaning

3.2.1.1 Missing data

One of the first steps in data pre-processing is determining what to do with missing values in the dataset. García, Luengo, and Herrera (2015) discuss several ways in which missing values can be treated. The first is discarding the features or instances for which missing values (MVs) are observed. This approach can only be used if the data is Missing Completely At Random (MCAR), as otherwise removing this feature will cause bias to be induced in the data. Fully discarding a feature is also an approach which can be used, when a lot of MVs are observed for one feature. A rule of thumb is that if about more than 60-70% of the values is missing, it is better to remove the feature.

The second approach is imputation by means of sampling. This approach is most suitable for non-categorical features. When applying this approach, maximum likelihood procedures are used to estimate

the parameters of the complete portion of the data. Using these parameters, the MVs are then filled by sampling from this estimated distribution.

The third approach is to keep the instances with MVs but encoding these. For example, for categorical features, an ‘other’ category can be created, or for (positive) numerical features, a -1 can be used to illustrate that the value is missing. However, not all models are able to deal with having a -1 in a positive numerical feature, as they might try to establish formulas based on these numerical values and the -1 skews the distribution.

The final approach discussed by García et al. (2015) is multiple imputation. Where relationships between features are identified and used to estimate the values of the MVs. When this approach is applied, it is important to note that multiple values are estimated to represent the same feature. This way the uncertainty induced by estimating the MVs is accounted for. So, multiple datasets are generated, in which the MVs are receiving a different estimate. The results of the training and validation of these datasets can then be pooled to obtain results.

3.2.1.2 Outlier detection

When performing outlier detection, we are trying to identify examples in the data which behaviours are quite different from the expected behaviour. These examples are called outliers, or anomalies. For outlier detection, unsupervised learning can be used. Outlier detection has a high relationship with cluster analysis, as clustering is the process of finding majority patterns in the data and organise it accordingly. Outlier detection, on the other hand, attempts to catch the exceptional cases present in the data, which have significant deviations from the majority patterns (García et al., 2015).

Furthermore, statistical methods are available for detection of outliers. For example, the Z-score can be used to identify observations that are outliers, by calculating how many standard deviations a data point is from the sample mean. As a rule of thumb, 2.5, 3, and 3.5 are threshold z-scores to drop instances if the z-score is above these values. This results in approximately 3.4%, 1%, 0.2% being dropped respectively (Santoyo, 2017).

3.2.1.3 Noise reduction

This chapter focuses on the noise imperfections of the data, and how to deal with them. Noise in data is almost unavoidable and can have several negative consequences in classification problems. Performance of classification and regression models is highly dependent of training data, especially in supervised problems, where noise alters the relationship between the features and the dependent variable. Noise hinders the knowledge extraction from the data and spoils the models obtained using that noisy data when they are compared to the models learned from clean data from the same problem, which represent the real implicit knowledge of the problem (Zhu & Wu, 2004).

Broadly, there are two types of noise in data; the first is attribute noise, the second is class noise. Attribute noise occurs when one or more attributes of an instance are corrupted (missing, unknown, incomplete). Class noise (or label noise) refers to incorrectly labelling of classes in the training dataset. Many reasons for this kind of noise are present, such as subjective labelling, data entry errors, and inadequacy of the information used to label. Class noise can be subdivided into two sub-categories (Catal, Alan, & Balkan, 2011), being contradictory examples and misclassifications. The first occurs when duplicate (or remarkably similar) features represent a different class label (e.g. round and green once representing an apple, and in the second occurrence a pear). The misclassifications occur when labels are assigned wrongly (e.g. apple labelled as pear).

Several ways have been identified for dealing with noisy data, the most relevant are:

Using robust learners, these learning models are designed to be less influenced by noisy data. Such models often implement a purifying strategy, which reduces the possibility of overfitting to noise that is present in the training data. Note that when too much noise is present in the data set, a robust learner may also have poor performance.

Implementing data polishing methods, which aim at correcting noisy instances, before the model is confronted with the training data. As this method is very time consuming, it is most suitable for small datasets. Teng (1999) found that, when data polishing is applied to the training set, and the test set still contains noise, the performance is higher than when no noise reduction is applied.

Applying noise filters, which aim at identifying noisy instances in the data, and removing them from the training set. Most models require such an approach, as they are not very capable of handling noisy data.

As for the situation at the IAC we are dealing with a lot of data, data polishing is not a viable method. Moreover, from the data description provided in chapter 2.7, there is a lot of noise in the data set. Therefore, the most suitable approach to deal with noise is by applying noise filters.

Applying noise filters to the data, before the data is presented to the model has the advantage that the model selection is not influenced by the presence of noise in the dataset (Gamberger, Lavrac, & Dzeroski, 2000). García et al. (2015) discuss the three most common methods of applying noise filters to data, these are ‘Ensemble Filter’, ‘Cross-validated Committees Filter’, and ‘Iterative-partitioning Filter’. All these three methods use a voting scheme to determine which instances should be eliminated from the dataset. For this voting, two possible schemes are available. The first is a consensus scheme, which removes an instance if it is misclassified by all classifiers. The second is a majority scheme, which removes the instance if more the half of the classifiers misclassify the instance. The consensus scheme is more conservative, and is less likely to reject good data, but might retain bad data. The majority filter, on the other hand, is better at detecting bad data, at the expense of having a higher probability of rejecting good data.

In the following section, each of the noise filters mentioned above is shortly discussed, for a more detailed explanation, please refer to García et al. (2015), chapter 5.

The *Ensemble Filter* attempts to improve the quality of the training data by detecting and eliminating mislabelled instances. It uses a set of learning algorithms to create classifiers in several subsets of the training data that serve as noise filters for the training set. This is done by dividing the data in Γ subsets. The identification of potentially noisy instances is carried out by performing an Γ – Fold Cross Validation (FCV) on the training data with μ classification algorithms, called filter algorithms.

The *Cross-Validated Committees Filter (CVCF)* makes use of ensemble methods to identify and exclude instances which have been mislabelled. CVCF is mainly based on performing an Γ - FCV to split the full training data and on building classifiers using decision trees in each training subset (Verbaeten & Van Assche, 2003).

The *Iterative-Partitioning Filter (IPF)* is based on the Partitioning Filter. The Partitioning Filter is a scheme developed by Zhu, Wu, and Chen (2003), and is very effective for identifying and removing mislabelled data from large data sets. Large data sets cannot be learned after one time, while this is an assumption for most noise filters. The IPF iteratively removes noisy instances, until a certain stopping criterion is met (e.g. less than a certain percentage of instances identified as noisy).

As IPF is the most suitable for handling large datasets, this approach seems most promising for the case of the IAC.

3.2.2 Data transformation

After filtering out noise of the dataset, the data might still not useful enough for a learning model. The attributes in the data are often raw, meaning that it comes directly from databases or from the system where the data is currently being used. Therefore, the attributes likely must be transformed into attributes which can be used by a learning model. What kind of transformation is required, depends on the nature of the data and the distribution of this data.

3.2.2.1 Types of data

Broadly speaking, there are four types of data, split in two categories; categorical and ‘numerical’ (Donges, 2018). Categorical data represents characteristics, like gender or language. Categorical data can be split into two types, which are ‘nominal’ and ordinal’. Nominal data represents discrete units and are used to label data that have no qualitative value. Nominal data, however, do not have an order. Ordinal data is similar to nominal data, as it also has no quantitative value. Nonetheless, for ordinal data, the order matters (e.g. highest level of education).

Two types of numerical data have also been identified, these are ‘interval’ and ‘ratio’. Interval values represent units that are ordered and have the same difference. So, a unit is interval data when it contains numeric values that are ordered, and the exact differences between values are known. For ratio data, this same rule holds, but in addition it also has an absolute zero (starting point).

3.2.2.2 Numerical data

For numeric columns, normalization is required when features in the dataset have different ranges (e.g. age and income). The goal of normalization is to change the values of numeric columns in the dataset, to a common scale, but without distorting the differences in the ranges of values (Jaitley, 2018). So, normalizing data does not create new attributes, it just transforms the existing attributes into a new set of values which have the desired properties (García et al., 2015). The tree mostly adopted normalization techniques are ‘Min-Max Normalization’, ‘Z-score Normalization’, and ‘Decimal Scaling’.

Min-Max Normalization is the process of scaling the values of an attribute with respect to the minimal and maximal value observed in the attributes. The new range which is most widely used is [0,1], however, this is not necessary. To transform a value v of column i (v_i) by applying Min-Max Normalization, the following formula is applied: $v_i' = \frac{v_i - \min_i}{\max_i - \min_i}$. One of the downsides of Min-Max Normalization, is that it cannot handle outliers in a column very well. Therefore, it should always be checked if these are present in the column that needs to be normalized. Another downside is that the minimum and maximum value might be unknown.

The second form of normalization, *Z-score Normalization*, is better at handling outliers. Z-score normalization is based on the mean and standard deviation of the column and will therefore be more robust if outliers are present in the column. It is applied by calculating the sample mean (μ_i) and sample standard deviation (σ_i), and applying the following formula: $v_i' = \frac{v_i - \mu_i}{\sigma_i}$. By applying this, the new μ_i will become 0 and the σ_i will become 1, which are properties of the standard normal distribution.

The final form of scaling is *decimal scaling*, which is a simple way for reducing absolute numerical values. It normalizes values, by shifting the decimal point by dividing the values by a power of ten. So, $v_i' = \frac{v_i}{10^j}$ for which j is increased until the absolute max value in $i < 1$.

3.2.2.3 Categorical data

For categorical data, the type of transformation depends on the type of data, as different approaches need to be used for both types. Ordinal data is mostly encoded by assigning integers to each unique value that is present in the attribute (Brownlee, 2020). Therefore, it can only be used for attributes that have a (weak) ordinal relationship (e.g. bad = 0, normal = 1, good = 2). Whenever this method is used for variables that have no ordinal relationship, the variable might be misleading to the model. Instead, one-hot encoding should be used when no ordinal relationship exists between the values. This is where the integer encoded variable is removed and one new binary variable is added for each unique integer value in the variable. For example, if we have a colour attribute, with three values (red, green, blue), this will result in three columns. For the value ‘red’, the values in columns green and blue would be 0, the value in the column red would be 1. The only downside to this kind of encoding, is the increased memory usage.

3.2.3 Data reduction

Nowadays, it is not difficult to imagine the disposal of a data warehouse for an analysis which contains millions of samples, thousands of attributes, and complex domains. Data sets will likely be huge, thus the data analysis and mining would take a long time to give a respond, making such analysis infeasible and even impossible (García et al., 2015). A high dimensionality in a data set increases the computational complexity, as there are many possible predictor variables. Fukunaga (2013) argues that there is a linear relationship between the number of training samples required and the dimensionality for obtaining high quality models. Therefore, the size of the data set increases exponentially with the dimensionality, as a higher dimensionality also implies more training samples (Hwang, Lay, & Lippman, 1994). In the original data set provided by the IAC, there also is a high level of dimensionality. Therefore, data reduction might be required to reduce the memory usage, and improve the efficiency of the learning model (Han, Pei, & Kamber, 2011). García et al. (2015) define three categories of data reduction, which are ‘Dimensionality Reduction (DR)’, ‘sample numerosity reduction’, and ‘cardinality reduction’.

DR aims to reduce the number of attributes or random variables in the dataset, so it reduces the number of columns in the dataset. It incorporates methods such as Feature Selection and Feature extraction/construction, which detect, remove, and combine irrelevant dimensions. This is done by creating parametric models (such as regression) which predicts an attribute, based on lower dimensional, for which only the parameters need to be stored, instead of the actual data. Transformation of the original data into a smaller space can be done by Principal Components Analysis (PCA), factor analysis, Multi-Dimensional Scaling (MDS) and Locally Linear Embedding (LLE), which are the most relevant techniques currently present (García et al., 2015). The most commonly used a described method is the PCA, thus, this method will be used to reduce the number of attributes and random variables in the dataset. A more extensive explanation of PCA is given in chapter 3.2.3.1..

Sample numerosity reduction (SNR) is a method which aims at replacing the original data by a smaller data representation. While DR can only use parametric methods, requiring a model estimation that fits the original data, sample numerosity reduction can also be non-parametric. Non-parametric models work with the original data itself, and return other representations of the data, while maintaining similar structures. Methods for sample numerosity reduction are data sampling, data grouping, data sampling, and data clustering. An example of sample numerosity reduction is instead of using zip codes in the dataset, aggregate the zip codes to regions or states, reducing the number of unique values in the attribute.

Cardinality reduction contains the transformations applied to access a reduced depiction of the original data. An example of cardinality reduction is introducing an ‘other’ value, which is assigned to values which represent less than a certain threshold in the attribute (Casas, 2019). This is especially useful when many values only occur very few times. The transformation of variables, as discussed in the previous chapter, are also forms of cardinality reduction.

3.2.3.1 Principle components analysis (PCA)

PCA is a technique for reducing the dimensionality of large datasets. It increases interpretability while minimizing information loss. This is done by creating new uncorrelated variables that successively maximize variance (Jolliffe & Cadima, 2016). The basic idea is to find a set of linear transformations of the original variables which could describe most of the variance using a relatively fewer number of variables. This is done by figuring out patterns and correlations among various features in the data set. When high correlations are found, a decision needs to be made on reducing the dimensions of the data, such that the significant data is still retained. Principle components become a new set of variables, derived from the original set of variables. They are computed as such, that the newly generated variables are highly independent from each other. In their values, the most useful information that was scattered among the initial values is included. The whole process of PCA consists of five steps, which are briefly discussed below.

Step 1: normalization of the data. This step is often already conducted, before the PCA is started. As normalization is a highly effective way of improving the quality of the data. Normalization is executed by following one of the procedures described in chapter 3.2.2.2.

Step 2: creating covariance matrix. A covariance matrix is used to express the correlation between the different variables in the data set. Identifying highly correlated variables is essential, as they are likely to contain biased and redundant information, which reduces the overall performance of the model. A dataset with a dimensionality of p , will result in a matrix with size $p \times p$, as it expresses the correlation between every variable in the data set.

Step 3: Calculating Eigenvectors and Eigenvalues. These need to be calculated, that are required to be computed from the covariance matrix, to determine the principle components of the data set. For every dimension in the data set, an Eigenvector-Eigenvalue set needs to be calculated. The Eigenvectors show where the data, the most variance is present. As more variance in the data indicates more information in the data, Eigenvectors can be used to identify and compute the Principal Components.

Step 4: Computation of Principal Components. Once the Eigenvectors and Eigenvalues are computed, they need to be ordered in descending order based on the Eigenvalues. The Eigenvector with the highest Eigenvalue is the most significant, and thus forms the first principal component. Based on this order, the Eigenvectors with the lowest Eigenvalues have the lowest significance and can thus be removed to reduce the dimensionality of the data.

Step 5: Reducing Dimensions. Finally, we can re-arrange the original data, with the final principal components representing the most significant information of the dataset. This is done by transposing the original data and multiplying it with the obtained feature vector.

3.2.4 Data sampling

To ease the analysis and modelling of large data sets, sampling methods are used. García et al. (2015) identify four purposes of data sampling. These are ‘Reducing the number of instances submitted to the DM algorithm’, ‘Supporting the selection of only those cases in which the response is relatively homogeneous’, ‘Assisting with respect to the balance of data and occurrence of rare events’, and finally ‘Dividing a data set into three data sets to carry out the subsequent analysis of DM algorithms’. Thus, it is particularly useful for reducing the memory usage of the data set, and run analytical models more quickly, while retaining accurate predictions. Furthermore, sampling is also used for validating the model. This is done by creating a training set and a test set. The model is then trained on the training set, then its performance is analysed by predicting the test set and validating the output with the correct output.

The five most used ways for sampling are discussed in the following section:

Simple random sample, without replacement (SRSWOR) is generated by drawing s instances from the entire data set, with an equal probability for each instance to be drawn. As the instance is not replaced, the same instance cannot be drawn twice.

Simple random sample, with replacement (SRSWR) is very comparable to the SRSWOR method, however, this time instances are replaced after being drawn. Meaning, that these instances can be drawn more than once.

Balanced sample is a sampling method designed based on the target value. When the target value contains imbalanced classes (one class occurs way more than other classes), simple random sampling will likely result in a sample containing many instances for that class. This hinders learning, as the model will get limited information on the other classes. Balanced sampling overcomes this problem, by incorporating a similar number of instances for every class. This type of sampling is especially useful for imbalanced learning (García et al., 2015).

Cluster sample can be applied if the data set can be grouped into several disjointed groups (clusters). From each cluster a random number of instances can be drawn by using SRSWOR or SRSWR.

Stratified sample divides the entire data set into mutually disjoint parts (strata). From each of these strata, an SRS is drawn, to obtain a sample. This method is similar to the balanced sample; however, the predefined composition of the final results now depends on the distribution of the dependent variable.

To remove the probability of observing good results by chance, a k-fold cross validation can be executed. This process is illustrated in Figure 11 (Niu, Li, Wang, & Han, 2018), where each iteration $testsize = \frac{1}{nr\ of\ iterations}$, and the performance is the average performance over the ten iterations.

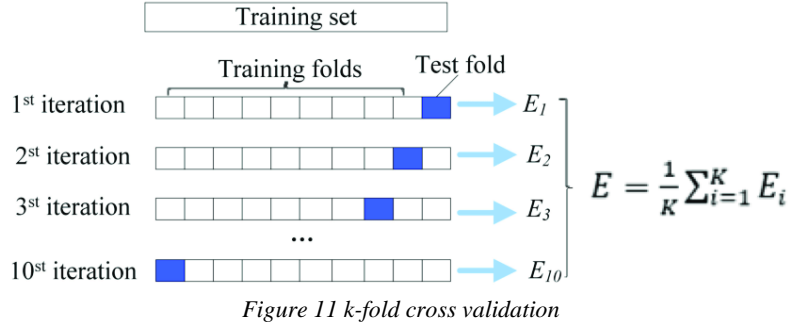


Figure 11 k-fold cross validation

3.2.5 Conclusion

The first step to take, for cleaning the data set, will be identifying missing values. If features are showing a high level of missing values (>70%), it is better to drop those features, as they are unlikely to provide useful information to the model. Furthermore, we must think about how the missing values can be filled with either interpolation or business rules. For example, in the Agreed TAT towards the customer, the missing values can be filled with a 'default contract length'. Next, the outliers should be identified, as they are likely to distort the data. For this process I will make use of statistical methods to filter outliers, as the unsupervised learning version will take too much time. The drawback of using these statistical methods, is that they are only capable of identifying univariate outliers, while the unsupervised learning methods would also be capable of identifying multivariate outliers.

The models that are selected in chapter 3.3, all are capable of handling noise well. So, the choice for using robust learners is implied. However, as mentioned in section 3.2.1.3, even robust learners lose performance if too much noise is present in the data. Therefore, some form of noise reduction still must be applied. The most promising form of noise reduction found, due to the large data set involved in this problem, was the IPF, which will be used to reduce noise. The numerical columns will be transformed with either a min-max scaler or a z-score scalar, depending on the distribution of the data. For the categorical features, one-hot coding is the most suitable, as for most of the categorical features, no ordinal relationship exists. As discussed in chapter 3.2.3, for the reducing of the dimensionality of the data a Principle Component Analysis will be conducted.

As the performance exchange classification is dealing with an imbalanced dataset, the balanced sampling method is the most suitable. Furthermore, in both cases the k-fold cross validation will be used to improve the validation quality.

3.3 Model Selection

After the data has been fully processed and prepared for the training, the model which is used for learning and predicting should be selected. The selection of the model depends on several factors, such as the size, nature and quality of the data, the available computation time, the urgency of the task, and what is the desired goal with the data. The best way of determining which model is most suitable, is by trial and error, as it is nearly impossible which model will perform best without trying first. However, the following cheat-sheet (Figure 12) is developed by Li (2017), gives a nice overview of available supervised learning methods, and simple questions to determine which models are most suitable.

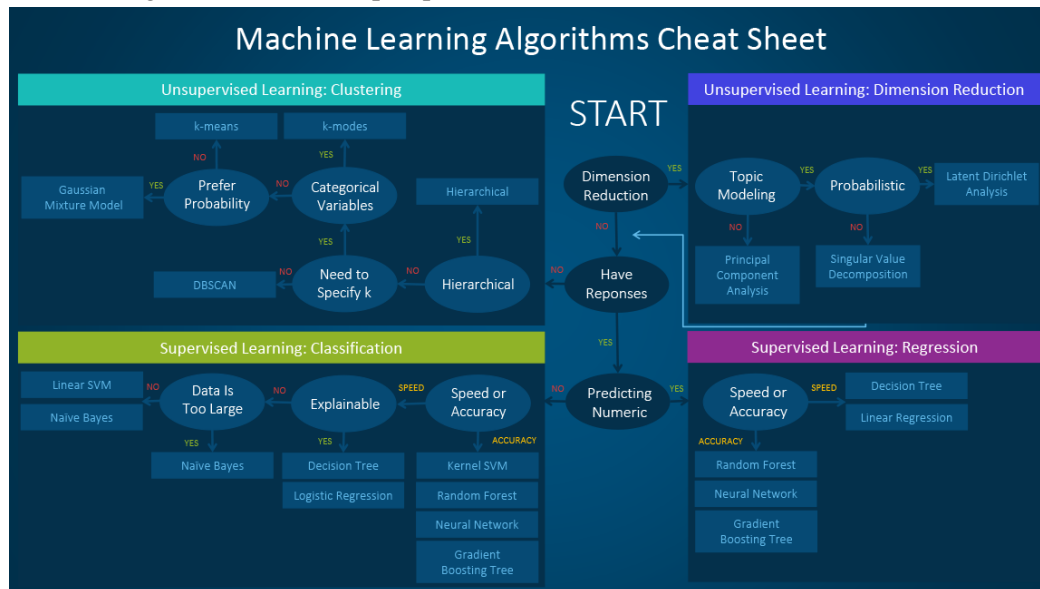


Figure 12 Machine Learning Algorithms Cheat Sheet

Following this cheat sheet, the first question we must answer is if the problem is regarding dimension reduction. Meaning, we would like to identify clusters of similar data points. As we want to predict a target value, this is not the case, we move downward to the second question. The second question is whether the target value is known or not (if responses are known). As this is the case, we move downward to the third question, which is if we want to predict numeric data or not. This answer to this question is no, as for the IAC the problem at hand is a classification problem. The final question is whether we prefer speed or accuracy. Because the model is not continuously running, but is ran at weekly intervals, we are more interested in accuracy than speed. This results in four models that are suitable for our situation. However, Kernel SVM training time increases quadratically, making the approach inappropriate for large datasets. Therefore, the 'Kernel SVM' method will not be further elaborated. The other three models ('Random Forest', 'Neural Network', 'Gradient Boosting Tree') are discussed in the following section.

3.3.1 Random Forest (RF)

A RF is built from a larger collection of decision trees, which are well-known for their ability to classify classes based on their features. It consists of many nodes, which all have a decision rule to divide classes. In each node, the model will try to ask a question, such that the resulting groups are as different from each other as possible (while the members of each group are as similar as possible). These nodes are connected by branches, which form the logical flow of the tree (Yiu, 2019). Decision trees are known for their easy interpretability, as it is quite easy to follow the logic by which the decision tree performs its classification.

Implied from its name, a RF is a collection of these decision trees. These decision trees all spit out a classification, and the class that is spit out most often, is the class that is selected. According to (Yiu, 2019) a RF uses 'A large number of relatively uncorrelated models (trees) operating as a committee will outperform any of the individual constituent models.' For the model to be effective, it is important

that the individual trees are uncorrelated. If the trees are highly correlated, they are likely to make the same mistakes, which causes the wisdom of the cloud principle to vanish.

To ensure the trees have a low correlation, two procedures are used. The first is called ‘Bagging’ (or Bootstrap Allocation), which makes use of the fact that decision trees are overly sensitive to the data which they are trained on. When applying bootstrap allocation, we allow trees to overfit the data on which they are trained, so they can grow very deep and are not pruned. The resulting trees will have high variance and low bias. Small differences in the used data, can lead to quite different tree structures. By drawing random samples (with replacement) from the data, many different trees can be created. Each of these trees is then used to predict the outcome, given certain input features, and the output with the most votes is selected.

The second process is called feature randomness. In a normal decision tree, which is not part of a RF, every feature is considered when a new node is created. For creating RF trees, when a node is created, a random subset of features is drawn, which that node can use to divide the groups. This ensures that a higher variation between trees is achieved, which in turn results in lower correlation between trees.

The use of RF has many advantages, and a few disadvantages. The first advantage is the versatility of the model, as it can be used for both classification and regression problems. Furthermore, they can be used without many pre-processing of the data, as they are very well at handling outliers, missing values, and noise. Furthermore, rescaling and transformation of data is often not required. Downsides to R the IAC are memory usage when a large dataset is used, overfitting when hyperparameters are incorrectly tuned, and interpretability problems, as RFs become a black box for classification (Kho, 2018).

3.3.2 Artificial Neural Network (ANN)

Artificial Neural Networks are part of a machine learning stream called deep learning. They are inspired by the structure and functioning of the brain. So, it tries to simulate a network of neurons like in the human brain, such that a computer will be able to learn things and make decisions in a humanlike manner (Marr, 2018). In essence, neural networks are multi-layer networks of neurons that are used to classify things or make predictions. In this section I will first briefly discuss how (forward feeding) artificial neural networks work, and then how they can be leveraged in practice. Finally, I will present some (dis)advantages of using neural networks.

In general, a neural network consists of at least one input layer, at least one hidden layer, and at least one output layer. In Figure 13 (Vieira, Pinaya, & Mechelli, 2017) a simplistic overview is given of the layout of a neural network. On the far left, the input layer is shown. It consists of data presented by the user, on which the model should base its decisions. Neural networks are known to require a lot of data to learn, so a large data set should be used as input.

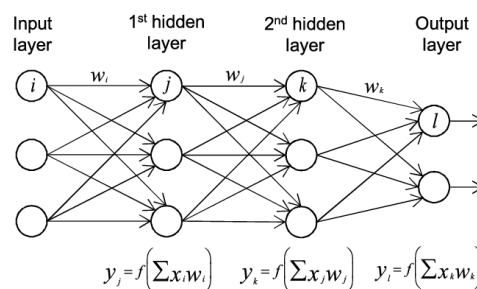


Figure 13 Simple Forward Feeding Artificial Neural Network

The arrows between the nodes describe how all neurons (nodes) are interconnected, and how data flows from the input layer, through the hidden layers, to the output layer. To each of those connections a weight is assigned, which indicates how much influence a one node has on another, depicted as w_i in Figure 13. Each connection of neurons has its own weight, and these are the only values that are modified while the model is learning. As more data is processed by the model, the network is learning more about the data, with each hidden layer adding some form of ‘understanding’ of the data.

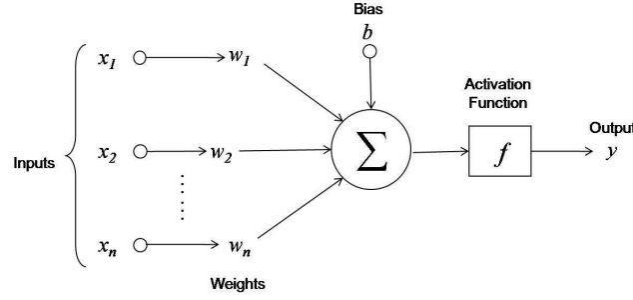


Figure 14 Working of a Neuron in a ANN (Arnx, 2019)

Each node of the ANN works in the following way. First, all values of every neuron from the previous layer that are connected to this neuron are summed. In Figure 14 there are three inputs (x_1, x_2, x_3), which means three neurons from the previous layer are connected to this example neuron. Before these inputs are added, they are multiplied with the connection weight (w_1, w_2, w_3). Furthermore, a Bias term is added to the total calculated value. This value is chosen before the learning phase and can give the model information on the learning data (e.g. if the sample is unbalanced, a bias term can be added). After performing the summation, an activation function is applied to the calculated value. The goal of this function usually is to transform the calculated value to a number on the range $[0,1]$. This process is executed for every neuron in a layer, and when the calculations are finished, the output forms the input for the next layer.

The learning of the ANN is done by minimization of a loss function, depending on the problem at hand. As we are dealing with a binary classification problem, the most appropriate loss function is a cross-entropy function, also known as logarithmic loss. The loss function is as follows:

$$Loss = -(y * \log(p) + (1 - y) * \log(1 - p))$$

Where y is the correct class label, p is the predicted probability that the target variable belongs in a class, and \log is the natural logarithm. By minimizing this loss function, the distance between the correct labels and the predicted labels is minimized (Brownlee, 2019).

The forward feeding model learns through backpropagation. This process incorporates comparing the predicted value with the actual value. If the model gave a wrong prediction, it would adjust weights and thresholds to minimize the difference between the actual and predicted value. These weights and threshold values are usually adjusted very gradually, therefore training takes so long for neural networks. The speed of learning can be adjusted, by adjusting the learning rate parameter. This parameter determines how it will modify a weight, either very gradually or by bigger steps. Correctly setting this parameter is important, as too high might make the model less robust against outliers, while too low might mean the model is not able to learn (Arnx, 2019).

An advantage of used ANNs over other machine learning algorithms, is the self-organisation that they possess. Which means that an ANN can create its own organisation or representation while learning about the data. Another advantage is that neural networks can be trained in parallel, which allows for distributed computation (Maind & Wankar, 2014). Another advantage is that ANNs can automatically approximate whatever functional form is best characterizing the data. For simple forms (e.g. linear), this property is of little value. But for more complex forms it allows ANNs to extract more signal from complex underlying functional forms (Hill, Marquez, O'Connor, & Remus, 1994).

Although ANNs have been proved very promising, there are several drawbacks to using ANNs. The first is the volatility of the development of ANNs. Compared to established, statistical methods, ANNs are very novel and undergoing a lot of changes (Hill et al., 1994). So, current best practices might be discontinued within a relatively short period of time. Another drawback is the difficulty to interpret ANNs, as they perform lots of computations and transformations to the input data, leaving almost no way to trace the way decisions are made. Furthermore, ANNs contain a lot of parameters that need to be tuned, such as number of layers, number of nodes, learning rates, activation functions and biases.

Due to this high amount of parameter tuning required, ANNs are very susceptible to overfitting to the training data. The final drawback is that ANNs are very computational heavy, and dedicated hardware is required to successfully leverage them.

3.3.3 Gradient Boosting Tree (GBT)

The final method for high accuracy, supervised machine learning is the use of Gradient Boosting Trees. These are often used due to their efficiency, accuracy, and interpretability. In this section, I will briefly discuss how GBTs work, and what their (dis) advantages are.

A GBT is an ensemble model of decision trees (see chapter 3.3.1), which are trained in sequence. For each training iteration, a GBT learns the decisions trees by fitting the negative gradients (residual errors) (Ke et al., 2017). In general, Gradient Boosting involves three key elements. The first is a loss function, which is the optimization target. The second is a weak learner (simple decision tree), which is used to make predictions. The final element is an additive model to add weak learners and minimize the loss function.

The Loss Function is depending on the type of problem that needs to be solved. The key is that it should be differentiable, as the differentiation of the loss function allows the determination of the gradient. The gradient boosting framework is very generic, so every loss function can be used, as long as it has a derivative (Brownlee, 2016a). For the weak learner, generally a decision tree is used. More specifically, regression trees are used, which output real values for splits, which can be added together. This allows subsequent models outputs to be added, which corrects the residuals in the predictions. The generation of trees is a greedy procedure, where split points are selected based on minimization of the loss. It is important that the weak learners are very limited to their size and depth, as increasing these too much, will result in too much computing power required to generate the weak learners, reducing the efficiency of the model. Finally, an additive model can be generated, where trees are added one at a time, and existing trees in the model remain unchanged. A gradient descend method is used to minimize the loss while adding trees. However, instead of a set of parameters (like in weight in ANNs), the GBT uses decision trees. After calculating the loss, a new tree is added to the model, to perform the gradient descend procedure. To achieve this, the tree is parameterized, and these parameters are modified in such a way that the residual loss is reduced. When this process is finished, a sequence of trees that which can be used to produce predictions for the problem at hand (Brownlee, 2016a).

An advantage of using GBT is that the model is often able reach extremely high predictive accuracy, higher than many other models. Furthermore, it allows for a lot of flexibility, as it can optimize many different loss functions without many alterations. Finally, the model requires little pre-processing, as it can handle categorical, numerical values, and missing values, as they come. This last property makes the GBT a particularly good model for initial exploration of the predictability of a problem, as it is able to produce results without much time investment (apart from computing times).

Disadvantages of GBTs are that the model will continue to improve to minimize all errors. This will result in a high probability of overfitting if incorrect validation methods are used. Furthermore, the GBT will require a lot of computation power, as many trees need to be generated before reasonable performance is achieved. These trees will consume a lot of computing power, time, and memory, making hardware requirements rather steep. The final disadvantage comes from the flexibility advantage, as the flexibility results in many parameters (number of iterations, regularization parameters, tree depth) that interact and influence the behaviour of the approach. Therefore, tuning of this parameter to improve learning efficiency and performance requires a lot of time.

3.3.4 Conclusion

All three models seem very promising for solving the problem at hand. They are all selected for their accuracy over speed, so this disadvantage reported for every model can be neglected. Although GBTs and RFs will likely outperform a ANN in a very basic model, a correctly hyper tuned ANN will perform significantly better (Haldar et al., 2019). Higher accuracy will reduce the chance of missing parts for

which alert were required, and thus is the main aim in for the model. Therefore, in the remainder of this thesis, I will focus on the use of ANNs.

3.4 Training the model

Now we have selected the model, it is time to trains these models. Training of the models is the process of incrementally improve the ability of the model to predict the target variable, by feeding it the historical data. At first, the model will predict fully at random, as it has no idea what the data is, and how this data can explain the target variable. For a linear regression model, the formula the model generates is in the form $y = a * x + b$, where the model should predict a and b in such a way that the mean square error is minimized. For every x , the model will predict y and based on the difference between the predicted and actual value a and b are adjusted. This is the process is called training.

For machine learning, there are many more parameters involved in the model that can be adjusted. For example, the number of weights in an ANN, which is the product of the number of nodes in every column. Additionally, every node also has its own bias term. These terms, usually are formed into matrices, which W indicating the weights matrix, and B for the bias terms (as shown in matrices below).

$$\begin{matrix} w_{1,1} & w_{1,2} & b_{1,1} & b_{1,2} \\ w_{2,1} & w_{2,2} & b_{2,1} & b_{2,2} \\ w_{3,1} & w_{3,2} & b_{3,1} & b_{3,2} \end{matrix}$$

The training process involves initializing some random values for W and B and using these random values to predict the output. As expected, this holds a terrible performance, as randomly filling these matrixes provides no information of the correct target values. Now, these predictions are compared to the correct output values (that the model should have produced). Finally, the values stored in W and B are updated, to improve the quality of the next predictions. The way these values are updated, is different for each kind of machine learning model. This process is repeated for a predetermined amount of cycles (called steps), with each step updating the values stored in W and B (Yufeng, 2017).

As discussed in chapter 3.2.4, the full data set needs to be split into a training set and a test set. The training set is fed to the model, such as described above, and the test set is used to validate the model, which is described in chapter 3.5.

3.5 Evaluation of the model

After the model has finished training, the performance should be evaluated. Evaluation of a model is performed by feeding the model a dataset that is has not seen yet. This ensures that the model is not solely relying on memory of the correct responses but is actually using generalizable rules to predict the correct outcome. Therefore, the outcome of the evaluation gives a particularly good indication on how the model will be performing in the real world, when it is confronted with real data.

As stated previously, in classification problems, the goal is to predict the correct class, given a set of features. To evaluate the effectiveness of the model, several evaluation techniques are available. Which one is the most suitable, depends on the underlying question that is answered using the model. In the case of this thesis, for the classification model, the goal is to identify which repairs are likely to result in a performance exchange. First, some of the most prevailing evaluation methods are discussed and compared, after which the most suitable one is selected. A confusion matrix is a two-dimensional matrix, of which the number of rows and columns is equal to the number of available classes in the target variable. In our case, there are only two classes, which are ‘performance exchange’ or ‘no performance exchange’. Generally, the predicted classes are presented in the rows, and the actual classes in the columns, an example of a confusion matrix is shown in Table 2.

N = sample size	Predicted: No	Predicted: Yes
Actual: No	True negative (TN)	False positive (FP)
Actual: Yes	False negative (FN)	True positive (TP)

Table 2 Example Confusion Matrix

The confusion matrix shows the performance of the model, by indicating how often classes are predicted (in)correctly. The cells marked green are correctly identified classes, while the those marked in red are incorrect. Based on these metrics, performance indicators (PIs) can be calculated.

The simplest PI to measure the performance of the model, is the *accuracy*. The accuracy is calculated as follows $Accuracy = \frac{TP+TN}{TP+TN+FN+FP} * 100\%$, so the correctly identified classes over the total sample size. The main drawback to using accuracy as performance indicator, is that it can be very skewed if we have an imbalanced sample. For example, if we try to predict a rare form of cancer that only present in 1/1000 cases, in a sample of 10,000 cases. The model might predict that zero of the patients has cancer, which has $Accuracy = \frac{9990}{10000} * 100\% = 99.9\%$, indicating the model is performing very well. However, the model did not predict one positive case correctly, meaning it has no practical use at all. For this purpose, recall, precision, and F1-score have been developed (Brownlee, 2014).

Recall (or True Positive Rate) is calculated as $Recall = \frac{TP}{TP+FN}$, thus, it measures how many of the actual positive cases is correctly identified by the model. Therefore, this PI is especially useful if we want to identify the minority cases present in the total data set.

The *precision* (or Positive Predicted Value), is calculated by $Precision = \frac{TP}{TP+FP}$, so, it measures how many of the positively predicted cases is actually positive. Often, precision is called the classifier exactness.

Most of the time, a trade-off needs to be made between the recall and the precision. If we rather identify as many of the positive cases (high recall), we need to accept that this will likely increase the number of false positive cases, and thus lowers the precision. A more balanced approach is by using the F1-score, the F1-score is calculated by $F1 - score = 2 * \frac{Precision * Recall}{Precision + Recall}$, which conveys the balance between the recall and the precision.

Another PI is the ‘Receiver Operating Characteristics’ (ROC) Curve, which is the ratio of the True Positive Rate and the False Positive Rate. The ROC curve is visualised in Figure 15. The larger the area under the curve is, the higher the accuracy of the model. If the area is 1, it means that the model fits the data perfectly, making it likely that overfitting took place. On the contrary, if the area is less than 0.5, the model is too inaccurate to be used reliably.

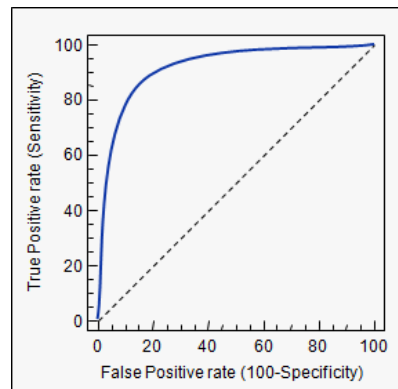


Figure 15 Receiver Operating Characteristics Curve (Bhattacharya, 2018)

From this ROC curve, the area under the curve (AUC) can be calculated. A theoretically perfect model will have an AUC of 1, as it will be a vertical line on FPR = 0, and TPR = 1, thus the area under this curve is 1. A receiver operating curve is more likely to look similar to Figure 15, as shown above. The ROC provides an indication of the balance between the true positive rate and the false positive rate. Selecting a model based on the ROC is produces, will prevent the class imbalance to highly impact the performance of the model.

3.6 Hyper parameter tuning

As mentioned in section 3.3.2, ANNs contain a lot of parameters that need to be tuned, to achieve good performance. Without proper parameter tuning, ANNs are likely to be outperformed by other, less extensive models. The process of tuning these parameters is called (hyper) parameter tuning, and involves an iterative, time consuming process in which parameters are altered and the results for different settings are compared (Steward, 2019). The parameters that need to be tuned are: 'Nr of layers', 'Size of layers', 'Activation Function', 'Learning rate', 'Dropout size', 'Batch size', 'Nr of epochs', 'Batch size'. The ANNs are compared on the following four metrics, when parameter tuning is applied: 'Converge speed' and 'Performance'.

3.6.1 Description of parameters

3.6.1.1 *Nr of layers*

The number of layers simply refers to the total number of hidden layers that is present in the model. More layers allow the model to approximate more complex functions, at the cost of increasing computation time. As discussed in 3.3.2, the layers are fully connected, and these connections are the weights that need to be updated. Thus, the number of layers has a high influence on the number of weights that need to be updated. Having more weights in the model, will allow the model to better understand the data, but increases the computation costs and the likelihood of overfitting.

3.6.1.2 *Size of layers*

The size of the layers refers to the number of nodes present in each layer. This size can either be fixed, for each layer, or varying per layer. When a layer consists of too little nodes, the model will not be able to incorporate all available information of the data into the model. However, when the model contains too many nodes, the model will start to 'remember' the training set, instead of developing generalizable rules. Generally, for tabular datasets, the size of the layers decreases with every layer. So, the first hidden layer is the largest, and the last hidden layer is the smallest.

3.6.1.3 *Activation function*

As mentioned before, an artificial neuron simply calculates a weighted sum of its input, adds a bias term, and decides if it should activate or not. This last part, deciding if it should activate or not, is based on the activation function. This activation function activates based on the weighted sum of the input of the artificial neuron. Choosing the right activation function is important, as not every function can be used in every situation. Furthermore, some activation functions are more computationally heavy than other functions (Sharma, 2017).

3.6.1.4 *Dropout size*

Adding a dropout layer to the ANN is a way to prevent overfitting of the model. A dropout layer works by randomly setting the outgoing edges of hidden units (neurons that make up hidden layers) to 0 at each update of the training phase. This ensures that the model must find a way to generalize, in order to achieve high accuracy, instead of 'remembering' the training set. The size of the dropout indicates how many nodes are dropped. As with the number of layers and the number of nodes, again we have a trade-off between over and underfitting. Dropping too little nodes in the dropout layer will allow the model to overfit, while dropping too many nodes can cause the loss of useful information. Generally, the dropout size is argued to be on the interval (0, 0.5) (Srivastava, Hinton, Krizhevsky, Sutskever, & Salakhutdinov, 2014).

3.6.1.5 Optimizer

The optimizer is the collection of algorithms that are used to change the attributes of the ANN, such as weights and learning rate to reduce the loss of the model, and thus allows the model to ‘learn’. The optimizer defines how weights and learning rates should be altered to reduce the loss of the model. Selection of the optimizer is mostly based on the availability of time and data. Some optimizers require large amounts of data, while others can achieve good results, while data is sparse. As this is the case for the IAC, the according to Ruder (2016) the best optimizers to use are adaptive learning-rate methods. For these methods, the learning rate decays, based on the number of times the model has seen the training set. Furthermore, the ADAM optimizer is argued to be the overall best choice, because of its wide usability and lack of extra parameters that require tuning.

3.6.1.6 Learning rate

For each optimizer, a learning rate must be determined. This learning rate determines how steep the alterations are that are made to the model, based on the loss gradient. Choosing a value that is too small, leads to very long learning times and a model that gets stuck in a local optimum, while a too large learning rate leads to a sub-optimal set of weights caused by too fast learning. Another option is letting the learning rate change based on a learning rate schedule, during the training. The effect of the learning rate is illustrated by Zulkifli (2018), shown in Figure 16. While eventually the model with the low learning rate might converge to the loss of the good learning rate, it will take way too long, using up too much GPU power and time.

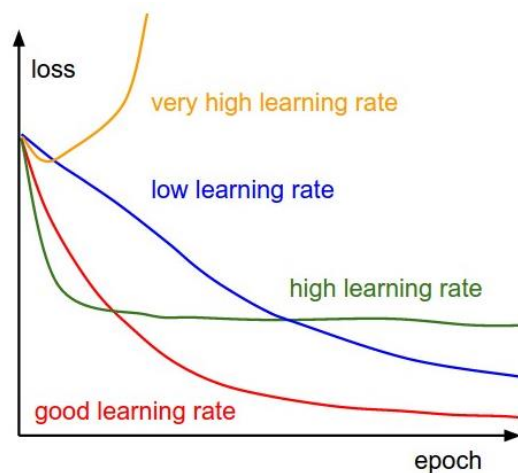


Figure 16 Effect of various learning rates on convergence

One learning rate schedule is the time-based decay, where the learning rate decreases based on the number of epochs the model has trained on. The learning rate is then determined by the following formula: $Learning\ rate = \frac{1}{(1+decay*epochs)}$. This is the most used learning rate schedule, due to its generalizability and ease of use. Another schedule is the step decay, where the learning rate is decreased by a factor every few epochs. Common uses are halving the learning rate every ten epochs.

3.6.1.7 Batch size

The batch size refers to the size of the sample that is fed to the model, before the weights are updated. The samples fed to the model are used to approximate the cost function, which is then minimized to improve the model. If too few samples are fed to the model, the model might update the weights based on noisy instances, reducing its generalizability. However, using a too large batch size will require a lot of processing power, as more predictions must be made before the model weights are updated and the effect of the updated weights can be observed. Furthermore, the hardware requirements of using a large batch are a lot higher, as the entire batch must be loaded into memory before it can be processed. The

most common setting for batch size is 32 (Bengio, 2012), as this setting generally achieves good results, while having low hardware requirements.

3.6.1.8 *Number of epochs*

The number of epochs refers to the number of times the model sees the entire training set. As seen in Figure 16, the number of epochs required to learning is highly dependent on the setting of the other hyperparameters. Therefore, often an ‘early stopping’ decision rule is used. This early stopping rule stops the training process, if the model has not improved for a x number of epochs. Stopping the model before the full number of available epochs is used, allows us to set the number of epochs to a large number, and stop the training once the model is no longer reducing the loss value.

3.6.2 *Tuning of parameters*

The large number of parameters that need to be tuned, with the large set of states the parameter can take, in combination with the difficulty of training ANNs, hyper parameter tuning is a computationally heavy. The most commonly used approach for parameter tuning is Grid search (Brownlee, 2016b), where the grid is formed by possible states for each of the parameters. Due to the sheer size of the grid (product of all states of all parameters), not every option can be explored. Therefore, the end user has to decide which options are present in the grid (e.g. 100, 200, 300, or 400 nodes for each layer), this way the solution space is limited and interesting areas can be found (for example when 300 nodes always outperforms the other settings). Then, the grid search can be performed again, but this time with 250, 275, 300, 325, and 350 nodes, to explore this area in more detail.

Another proposed method for hyper parameter tuning, is by using random search. Bergstra and Bengio (2012) found that randomly selecting hyper parameters outperforms grid search. So, instead of systematically exploring promising areas, randomly drawing values from the entire solution space is preferred. This approach is especially dominant for problems that have a high dimensionality. The drawback of using this approach is that results are unintuitive and thus difficult to understand why certain hyperparameters are selected.

The final method to select hyper parameters is proposed by (Shahriari, Swersky, Wang, Adams, & De Freitas, 2015), who argue that Bayesian optimization is the most promising method. Simply put, the Bayesian optimization method trains the model with different hyper parameter settings and observes the function that is generated by the model. This process is repeated many times, while each time the newly selected hyperparameters are only slightly different and help plot the next relevant segment of the problem space. By executing this process, the goal is to approximate the optimal function across the entire problem set. Shahriari et al. (2015) found that this approach yields significantly better results than random search, while using the same number of trails.

As the results from Bayesian optimization are the most promising, this method will be used for hyper parameter tuning in this thesis.

3.7 *Distribution Fitting*

Distribution fitting is the process of finding a probability distribution with specific parameters, which allows summarization of a large amount of data. One of the main advantages of fitting a probability distribution to historical data, is the usability in simulation studies (Ramberg, Dudewicz, Tadikamalla, & Mykytka, 1979). With a fitted distribution, instead of drawing randomly from historical values, values can be drawn from the fitted distribution. This limits the memory requirement and is often faster than drawing from historical values.

The maximum likelihood estimation (MLE) method is the most popular method for estimating distribution parameters from an empirical data sample. The MLE method is based on the philosophy of finding the distribution, that has the highest likelihood of producing the empirical sample (Myung, 2003). With every probability distribution, parameters are associated. As these parameters change in

value, different probability distributions are constructed. Myung (2003) defines the problem as follows: 'Given the observed data and a model of interest, find the one Probability Density Function (PDF), among all the probability densities that the model prescribes, that is most likely to have produced the data.'. A likelihood function, is a function of the parameter, given a particular set of data, defined by the parameter scale. So, the likelihood function, is a function which can be maximized, to obtain the parameter settings that have most likely provided the observed data. For this process, fully automated Python packages exist, and for the practical implementation these will be used. However, I still wanted to briefly elaborate the technical background of the MLE approach.

3.8 Function approximation

Due to the high number of stochastic variables persistent in the supply chain of the IAC, finding a true optimal solution is impossible. Furthermore, optimization models often lack an estimation of the variability or robustness of a solution in a stochastic environment. Metrics such as lead-time variability, percentage of on-time delivery and so on, are hard to obtain, let alone incorporate, when using an optimization model (El-Aal, A El-Sharief, Ezz El-Deen, & Nassr, 2008). Simulation in supply chain management can offer a complement to the more prevailing modelling using optimization models, since simulation is more suited for representing random effects. Furthermore, the purpose of simulation is to shed light on the underlying mechanisms that control the behaviour of a supply chain. Which is key for obtaining acceptance of the tool by the operational planners. Finally, a simulation can be used to predict the way in which the supply chain will behave when underlying factors are changed, allowing for effective ways to evaluate what-if scenarios.

Although there are many strongpoints for using simulations, there are also drawbacks. For example, a simulation is very dependent on the quality of the input data. So, again the garbage in leads to garbage out principle holds. Additionally, simulations are incapable of providing easy answers to complex problems. Finally, simulations are incapable of solving problems by itself. It only is a means for predicting outcomes, given certain settings (Cagliano, Rafele, & management, 2008).

A powerful simulation type is the Monte Carlo simulation, which involves assigning multiple values to an uncertain variable to achieve multiple results, which can then be averaged to obtain an estimate. Monte Carlo simulation can be viewed as a brute force approach, which can be used when exact solutions are unavailable. The advantage of using random variates instead of a single average number, is that using random variates allows to incorporate the uncertainty of the random variables. Another advantage of using Monte Carlo simulations, is the ability for evaluating what-if scenarios. This allows the operational planners to quickly analyse the impact of for example, delivering a part to an external party, thus reducing the on-hand stock by one. Furthermore, Monte Carlo simulations allow for easy visualisation, which aids in the acceptance of the tool by the operational planners, as alterations of underlying parameters visually impact the outcome of the simulation.

3.9 Conclusion

In the first section of this literature review, the different types of machine learning are discussed. As we want to determine if a repair will be finished or not, based on the available features, supervised machine learning is the most suitable approach.

The first step in data preparation is identifying and dealing with missing values. First, missing values are, if possible, filled by either business rules or interpolation. If after this process a high level of missing values persists (>70%), the feature is dropped. Next, outliers should be identified and removed, to prevent them from distorting the data. Identifying outliers using unsupervised learning is a time consuming and requires a lot of additional research to be executed correctly. Therefore, simpler statistical methods will be used to identify outliers. To reduce the class noise persistent in the dataset, a IPF is used, a IPF iteratively removes noisy instances, until a certain stopping criterion is met. To ensure that all data is in the same range, a min-max or a z-score transformation is used, depending on the statistical distribution of the data. For the categorical features, one-hot coding is the most suitable, as

for most of the categorical features, no ordinal relationship exists. Finally, for reducing of the dimensionality of the data a Principle Component Analysis will be conducted. As the performance exchange classification is dealing with an imbalanced dataset, the balanced sampling method is the most suitable. Furthermore, in both cases the k-fold cross validation will be used to improve the validation quality.

The selected machine learning model is the Artificial Neural Network, due to academic interest in the topic and the novelty of the method. However, for validation purposes, the Random Forest will also be used, albeit with less parameter tuning, to limit the time required to validate the performance of the ANN. The models will be validated on the F1-score, which is a score for the balance between the recall and the precision. Furthermore, the Area Under the Curve is used to validate the model, which gives insight on how well the model can distinguish different classes. The final step before the model can be used to make predictions is tuning of the hyper parameters. This will be done using a Bayesian optimization, as the number of hyper parameters persistent in the model is huge and Bayesian optimization is an efficient way to find good parameters.

To approximate the statistical distribution of turnaround times of different SKUs, distribution fitting based on the Maximum Likelihood Estimation will be used. This statistical distribution can then be used to simulate turnaround times of returning repairs, which form the supply of the inventory pool of the CMA program. The prediction of the performance exchange orders, the turnaround times of historical repairs, and the demand from the FE program are subsequently used as input for a Monte Carlo simulation, which is can be used to approximate the chance of backorders. Due to the stochastic nature of the performance exchange orders, the turnaround times, and the demand from the FE program, an exact solution is not available, which generates the need for an approximation by the Monte Carlo simulation.

4 Methodology

In this chapter the question: ‘How can machine learning algorithms, as described in the literature, be used to create a model which indicates parts most urgently require attention from the operational planners?’ is answered. Furthermore, design decisions and methods used to retrieve the results presented in this research are described, as this will enhance the reproducibility of the research.

In section 4.1, the conceptual solution is discussed, together with the choices and decisions which led to this solution. In section 4.2 the available data, and the processing required to be able to use this data is described. In section 4.3 the methods used to predict the demand coming from the performance exchange program are discussed. Section 0 provides information on how these methods can be used to approximate the chance of backorders for the CMA inventory pool. Finally, in section 4.5 the main conclusions of this chapter are provided.

4.1 Conceptual solution design

4.1.1 Overview

As discussed in chapter 2, the existing alert generating model does not incorporate the following components ‘demand from the performance exchange program’ and ‘repair pipeline’. The lack of inclusion of those two components makes the existing alert generating model merely reactive. Furthermore, the increase in PE makes the incompatible with the current way of working at the IAC. To include the performance exchange orders, and the repair pipeline, the following conceptual design has been composed, a schematic overview is found in Figure 17.

As derived from Figure 17, the first steps are selecting which features from the available data are of importance, and checking the quality of the data, to evaluate if pre-processing is required. This step is mainly of importance for the classification model, as the historical turnaround times used to model the repair shop pipelines are directly available in the (cleaned) dataset. Once the data is sufficiently clean, the data is used to assemble an artificial neural network, which has as main goal, identifying which orders will not finish in time, and thus need to be fulfilled by an SKU from the inventory pool. This identification, together with the repair shop pipeline and the FE demand, are used to model the chance of backorders for the CMA pool for a specific SKU, by means of a Monte Carlo simulation. Finally, the SKUs with the highest chance of facing backorders are presented to the operational planners.

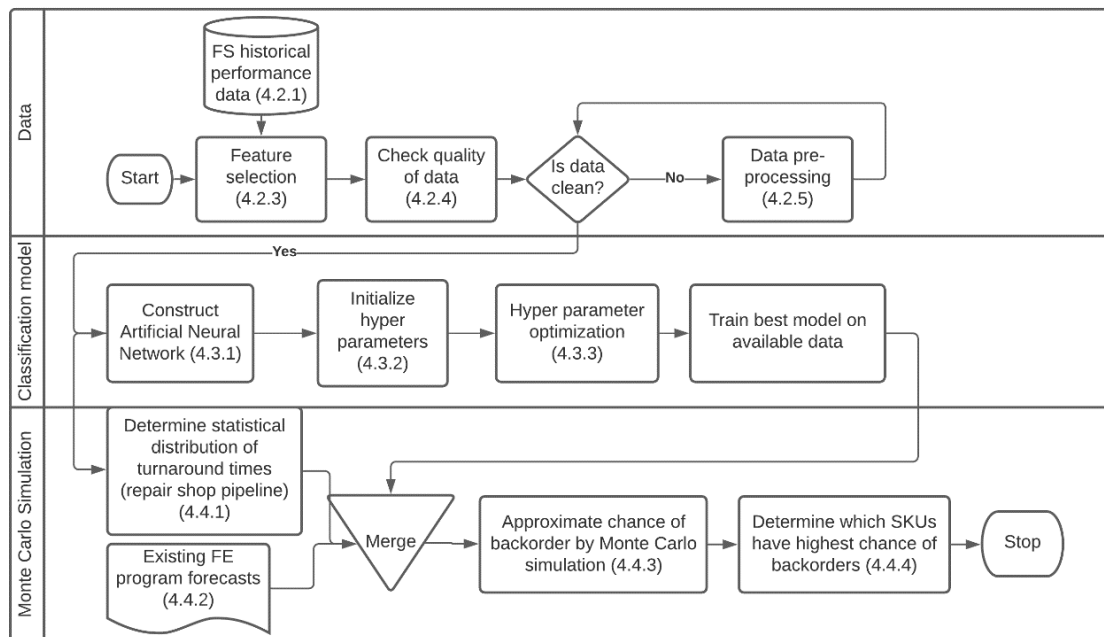


Figure 17 Conceptual model for alert generation

4.1.2 Design choices

4.1.2.1 Planning horizon

In agreement with the operational planners, the scope of the tool is determined to be two weeks. This is equal to the operational planning horizon. Furthermore, the planners indicated, that more than two weeks in advance, interventions are very rarely required. The first intervention that is used (as it is the cheapest) is asking the repair shop if a repair order for a SKU can be prioritized. But due to the planning horizon of the repair shops, more than two weeks in advance, this is of no use

Furthermore the scope of two weeks is selected based on the time between activating an intervention and this intervention being effective, the cost savings per unit time as defined in chapter 2.5.8, and the time available for the operational planners. When the scope is set too short, say shorter than one week, the interventions would still be costly, however, if the scope is set too large, too many alerts would be generated, taking up too much time of the operational planners.

Another factor which limits the scope at two weeks are the returning core units from forward exchanges. These returning core units are not incorporated in the model, as the process of returning and repairing these parts usually takes longer than two weeks. Whenever a forward exchange is performed, the customer generally has five working days before to send the unit back to the IAC. Additionally, lead times and time for the inbound warehouse further increase the time before the repair can start. In total, this process takes about eight days, if the customer adheres to the contracted return times, which is definitely not always the case, as seen in section 0. So, if the scope is less than two weeks, these returning repairs are not likely to skew the outcome of the simulation if the scope is limited to two weeks. After this period however, parts might be in repair while they are not incorporated in the model. Meaning that the model becomes oversensitive, as the modelled supply is lower than the actual supply.

4.1.2.2 Contracted repairs exceeding TAT

The output of the ANN is a continuous number on the range [0, 1]. On this range, 0 means the model is 100% sure the agreed TAT will not be exceeded, while 1 means the model is 100% that the agreed TAT will be exceeded. For every repair that has a due date within the considered scope, the model will predict whether the due date will be met, and thus, if a performance exchange is required. The model always assumes that the performance exchange is only used when the due date is not met. However, historically performance exchanges have been sent out weeks before the due date. Furthermore, sometimes performance exchanges are offered to the customer, but are rejected, as the customer is not in direct need for the part. As the data, nor the operational planners, are able to pinpoint situations in which these cases occur, the following assumption is developed:

'If the IAC is unable to meet the agreed TAT, a Performance Exchange is obligatory, and the customer will always accept this exchange. When the OH-stock is ≤ 0 at a moment a Performance exchange is required, it is considered a backorder situation.'

Moreover, the agreed due date, is the date on which the demand comes in, if a performance exchange is required. E.g. if a repair was due on the 5th of January and the model predicts that a performance exchange is required, the demand for the 5th of January increases with 1.

For simulation of the performance exchange demand, every contracted open repair is evaluated by the classification model, to predict the probability that a performance exchange from the inventory pool is required. This probability can then be used to randomly select with success probability p (coming from the classification model), to simulate whether a repair is finished in time or not.

4.1.2.3 *New buy*

Two situations have been identified, for when parts are acquired. The first is when a part is scrapped in the repair shop, but the operational planners want to keep the inventory position at the previous level. A part is then acquired from available sources (market, OEM, competitor etc.), to fill the position previously taken by the scrapped part. The second situation is when the tactical planners decide that the inventory level should be increased. Again, a part is then procured from available sources.

Due to the declining demand on the CMA program, a lower inventory position is required to maintain performance levels. Hence, when a part is scrapped, the inventory position is not necessarily recovered. Furthermore, the management of the IAC is reluctant with investing in acquisition of new parts to increase the inventory level, as parts are of high value and require large investments, even though the tactical calculations might have shown that the inventory position for a SKU needs to be increased. Thus, the management prefers incurring larger operating costs over a long-term investment, resulting in short term cash reduction.

Finally, when the operational planners decide to procure a SKU, and this decision is approved by management, a tender offer is opened. According to the tactical planner, these tender offers can be a long-lasting process, and historical data on arrival times for procurement parts are unknown. So, pipeline stock not visible in the planning horizon, are not included in this thesis, as they will never influence the outcome during the planning horizon.

To not completely neglect the effect of the acquisition of parts the operational planners can manually increase the inventory by using the what-if analysis. Now the acquisition of parts can still be considered in the model, without having to be specifically modelled. So, if they know about a part that is arriving soon, it can still be considered in the simulation. The full implementation of these what-if scenarios are discussed in section 6.4.3, where different uses are discussed.

4.1.2.4 *Approximation of repair shop pipeline*

As described in section 2.7.1, no accurate data on the status of repairs is available, so the pipeline needs to be approximated. This will be done by using historical turnaround times, and the present duration of a repair. A detailed description and the execution of this approach are found in section 4.4.1.

4.2 Data

4.2.1 Original dataset

The original data set, as described in section 2.7 is a dataset containing 198 columns and 285,561 rows (historical orders). This dataset is used for making tactical decisions and is based on queries from the ERP system of the IAC. To reduce the load on the ERP system, this dataset will also be used as basis for this thesis, even though some information might be missing. The tactical planner at the IAC argues that the data before 2007 is of low value, and upon inspection this seems true, as it has a very high percentage of missing values (about 20% of all values is missing). Removing all data from before 2007 reduces the dataset to 271,163 rows.

4.2.2 Dependent variable

Originally, the goal was to predict the total turnaround time for each part that has a PE contract. However, during initial exploration of the machine learning models, it turned out that there is too much variance in the total turnaround time. The average means absolute deviation observed was 19.61 days, which means that on average the model predicted the TAT 19.61 days off. Many of the contracts have an agreed TAT of 30 days, meaning that the model is on average way too inaccurate to determine if a repair will be finished in time or not.

So, a better dependent variable is, if the model predicts if the repair will be finished in time or not. If the repair is not finished in time a PE is required, according to the assumption made in section 4.1.2.2. Thus, the PE demand can be predicted using the ANN, by letting the model prediction how likely the agreed TAT will be exceeded.

4.2.3 Feature selection

As the data used in this research, as introduced in section 2.7, is not specifically gathered with this research in mind, a lot of features persistent in the data set are of no use (e.g. customer name, account number, project etc.). An overview of the entire available dataset, with a short description of each column is found in APPENDIX A: Detailed Description of Available Data. Based on discussions with the two operational planners and a tactical planner, potentially useful features have been identified or extracted. These features, together with some key characteristics are presented in Table 3.

Feature name	Description	Data type	Number of categories
Shop name	The name of the shop that is executing the order	Categorical (nominal)	495
Work performed	The work scope of the repair (e.g. Repair, modification, overhaul)	Categorical (nominal)	6
Condition	The condition of the SKU, indicated by the customer (e.g. serviceable, overhaul, inspect)	Categorical (nominal)	9
Priority	Priority as indicated by the customer (e.g. Aircraft on ground, critical, routine)	Categorical (ordinal)	4
Internal shop	Is the executing shop internal (1) or external (0)	Categorical (nominal)	2
Percentage in time	Percentage of historical orders for the executing repair shop that has been finished in time (at the arrival of the respective part), measured over the entire dataset	Numerical	-
Agreed TAT customer	The agreed total turnaround time as defined in the contract with the customer	Numerical	-
Now in shop	The number of the IAC SKUs in the executing repair shop, at the moment the respective SKU arrives	Numerical	-
Weekly aggregated TAT	The average achieved turnaround time of the past week, for the executing repair shop, at the moment the respective SKU arrives	Numerical	-
Three months aggregated TAT	The average achieved turnaround time of the past three months, for the executing repair shop, at the moment the respective SKU arrives	Numerical	-
Half year aggregated TAT	The average achieved turnaround time of the past half year, for the executing repair shop, at the moment the respective SKU arrives	Numerical	-
Year aggregated TAT	The average achieved turnaround time of the past year, for the executing repair shop, at the moment the respective SKU arrives	Numerical	-

4.2.3.1 Exploration of selected features

In this section, relationship between the dependent variable and the selected features is explored. As the dependent variable is a binary variable, exploration of the explaining power of the selected features can be challenging. Therefore, the explaining power of the features is evaluated against the total achieved turnaround time, from which the dependent variable is derived.

Numerical data

For the numerical data, a correlation matrix is created. This matrix shows the correlation between the different independent, numerical features and the dependent feature (in this case the TAT). The correlation matrix is found in Table 4.

	Weekly aggregated TAT	Three months aggregated TAT	Half year aggregated TAT	Year aggregated TAT	Agreed TAT customer	Now in shop	Total achieved TAT
Weekly aggregated TAT	1	0.6716	0.6206	0.5528	0.0497	-0.0866	0.2554
Three months aggregated TAT	0.6716	1	0.9231	0.8199	0.0547	-0.1421	0.2652
Half year aggregated TAT	0.6206	0.9231	1	0.9081	0.0555	-0.1621	0.2492
Year aggregated TAT	0.5528	0.8199	0.9081	1	0.0619	-0.1918	0.2297
Agreed TAT customer	0.0497	0.0547	0.0555	0.0619	1	0.1981	-0.0051
Now in shop	-0.0866	-0.1421	-0.1621	-0.1918	0.1981	1	-0.1634
Total achieved TAT	0.2554	0.2652	0.2492	0.2297	-0.0051	-0.1634	1

Table 4 Correlation matrix numerical values

The first observation is that the correlation between the aggregated TATs over different time periods is high, which is expected, as it is the average turnaround time of a shop, averaged out over different timespans. So, during the evaluation of the model, it should be evaluated of one (or more) off these features can be dropped, or if they all provide useful information to the model. The second observation is the low correlation between the Total achieved TAT and the Agreed TAT customer. This can be explained by the nature of the agreed TAT variable. This variable contains contracted times, which are thus terribly similar, making the variable more categorical than continuous. If it is incorporated as a categorical value, the following boxplot (Figure 18) is obtained, showing that the different categorical values have different distributions.

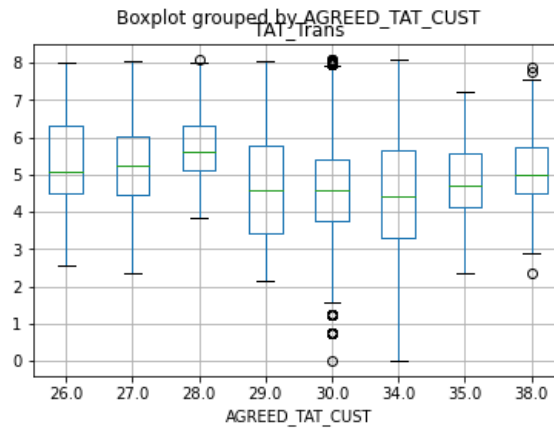


Figure 18 Boxplot of Agreed TAT customer with normalized achieved TAT

Categorical features

For evaluation of the categorical data, a two-way Anova (Analysis of Variance) test is conducted on the achieved Turn Around Time (TAT). This test identifies if particular instances of the independent variable are significantly different from other instances of that independent variable. This is done by observing the total variation in TAT in the sample, and then doing the same for one of the categories. The results of the Anova-test are presented in Table 5. As the F-values are large, with a high level of significance, we can conclude that for every variable, at least one category is significantly different from the other categories. Furthermore, I created boxplots for all the categorical values, with respect to the TAT, these can be found in

feature	df	sum_sq	mean_sq	F	PR(>F)
Internal or external (binary)	1	1643.824	1643.824	825.833399	<0.0001
Condition	5	541.5597	108.3119	54.4143664	<0.0001
Work performed	12	771.8677	64.32231	32.3146021	<0.0001
Shop name	270	3170.274	11.74176	5.89888944	<0.0001
Priority	3	391.1979	130.3993	65.5107408	<0.0001
Residual	12286	24455.32	1.990503		

Table 5 Two-way ANOVA test of categorical variables

APPENDIX C: Box plots of categorical Data.

Moreover, a large portion of the variance in the Total achieved TAT is explained by the Internal or External variable. On the other hand, which shop has executed the repair seems less significant. For now, each of the categorical features will be included in the model. After optimization of the model, the features will be evaluated, to observe the information that is obtained, by each feature.

4.2.4 Check data quality

Upon exploration of the status quo at the IAC, problems were already arising with the quality of the data. Many duplicate rows, missing values in features required to obtain the features as selected above (e.g. many of the orders did not have a date of entry). Furthermore, in the categorical features, very similar occurrences were found such as for the Work performed feature, were 'Repair', 'REP', 'RE', 'REPF' were all used interchangeably, while all these indicated the same value. So thorough cleaning of the dataset is definitely required.

4.2.5 Data pre-processing

In this section a brief overview is provided of the steps executed to pre-process the data and what the results of these steps are. For more details, please refer to APPENDIX D: Data pre-processing.

The first step in the data pre-processing is reducing missing data. First by the use of business rules or interpolation, missing values are filled. If these methods were deemed inappropriate, or unsuccessful and too many missing values remained in either a feature, or an instance, that respective feature or instance was removed. Luckily, by means of business rules, all features could be maintained in the dataset. However, some instances missed values for key instances (such as arrival date, making determination of the dependent variable impossible), and thus were removed. After removing these instances, 256,452 instances remained, and all 12 features are included in the dataset.

Next, noise reduction was required, as duplicated rows and confusing categorical values were persistent in the dataset. Furthermore, the majority of the orders was not under a contracted TAT, so for these orders turnaround times are less strict. This unfortunately resulted in a major loss of data because the dependent could either not be determined or was inappropriate for those cases. After removing all this noise, only 54,656 instances remain.

After removing the noise from the dataset, outlier detection took place. Due to the non-normality of the features, a regular z-score outlier detection is inappropriate. To overcome this problem, Leys, Ley, Klein, Bernard, and Licata (2013) propose a method where the median and the mean absolute deviation (MAD) are used. An outlier is identified as a value x_i that lays outside the interval $\frac{x_i - M}{MAD} \leq |\mp 4|$. In Table 6, below the threshold values and the number of dropped rows per feature is shown. After removing these outliers, a dataset with 48,854 instances remained.

Feature Name	Median	MAD	Lower Bound	Upper Bound	Excluded
AGREED_TAT_CUST	34	5.525586	11.89766	56.10234	679
NOW_IN_SHOP	57	269.9549	-1022.82	1136.82	742
Weekly	37.28571	17.98765	-34.6649	109.2363	1431
half_year	40.06222	13.86099	-15.3817	95.50618	1348
three_months	38.67201	11.61771	-7.79882	85.14284	676
yearly	40.77676	12.75443	-10.241	91.7945	926

Table 6 Outlier detection and removal

Moreover, data transformation took place, as all features should be on the same scale, and categorical features without ordinal relationships should be one-hot encoded. This means that every shop, every work performed, and every condition category got its own column. Therefore, the number of columns

in the dataset was highly increased, but the number of features remained the same (12 features). Furthermore, all the numerical values were scaled to a $[0, 1]$ interval.

The final step is data reduction, which is aimed at reducing the number of features, based on their explaining power of the dependent variable. Based on discussions with the tactical planner at the IAC, it was recognised that some repair shops are rarely used or are currently no longer being used. To overcome this problem, all repair shops that have not been used for over two years and have less than 0.05% contribution to the total number of repairs (~140 repairs), are placed in the ‘Other’ shop category. This reduction resulted in 143 shops remaining in the dataset. Furthermore, a principle component analysis (PCA) has been performed, to identify if numerical features could be removed. However, including all numerical features holds the best results, so for now all features are included in the final dataset.

The resulting dataset contains 176 columns (12 features) and 48,854 rows, which all values lay on the interval $[0,1]$. The high number of columns is mostly caused by the high number of different repair shops, different types of work performed, and conditions, which are now one-hot encoded.

4.3 Classification Model

The goal of the classification model, is to identify which orders with contracted turnaround times, will exceed this turnaround time. If such an order exceeds the agreed TAT, a performance exchange is delivered from the inventory pool. Therefore, less SKUs remain available for fulfilling regular (forward exchange) demand. In this section, the construction, optimization, and the performance of this classification model are discussed.

4.3.1 Construction Artificial Neural Network

The ANN is created in TensorFlow, which is a free and open-source software library for machine learning applications. TensorFlow is used, as it provides a high-level API for creating ANNs, so instead of explicitly programming every different activation function or optimizer, these can be called from the packages provided by TensorFlow. This decreases the complexity and the time required of programming the ANNs, while ensuring the functions are correct as TensorFlow is developed and used by professional Machine Learning developers. The programming language that is used for the implementation is Python, as this is a programming language supported by both TensorFlow and the development server of the IAC.

For determining and comparing the performance of different settings, the most suitable way as described by literature is using a k-fold cross validation method. As a large number of models will be validated in the parameter tuning phase, k cannot be too high, as that would lead in too many computing times required to calculate the different options. However, selecting k too low would increase the variation of the results. In general $k = 10$ is considered the most appropriate cross validation level (Kohavi, 1995)). However, training of the ANN is time consuming and training the same model ten times would take too long (one training & validation cycle takes 15 minutes) and would allow me to explore less potentially good settings. Therefore, a k of three would be sufficient, as it reduces the variation, while not creating exceedingly long runtimes.

4.3.2 Initialization of hyper parameters

The original model consist of an input layer with 175 nodes (equal to the number input columns i.e. features), which is fed the dataset as described in the previous chapter, a dense (hidden) layer of 175 nodes with a rectified linear activation function, a dropout layer, which drops 20% of the existing nodes, and an output layer with sigmoid activation function. The optimizer that is used is the ‘Adam’ optimizer, which is the most commonly used optimizer for tabular data. The model is trained for a maximum of 1000 epochs (recall that an epoch means that the model sees the entire training set), with a default batch size of 32. The training is stopped if for 10 consecutive epochs no improvements in the loss function are found. The original model has an average accuracy ($\frac{TP+TN}{FP+TP+FN+FP} * 100\%$) of 73.54%, an average precision ($\frac{TP}{TP+FP} * 100\%$) of 73.34%, an average recall ($\frac{TP}{TP+FN} * 100\%$) of 58.11%, and an average AUC of 0.804, which are promising results for an initial model. The ROC curve and the confusion matrix of the last model can be found in Figure 20 and Figure 19.

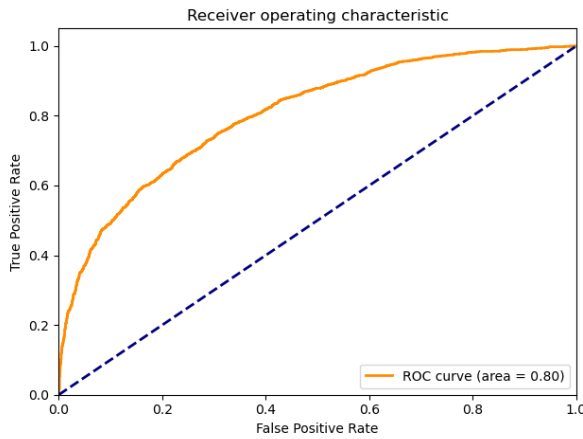


Figure 20 ROC curve initial model

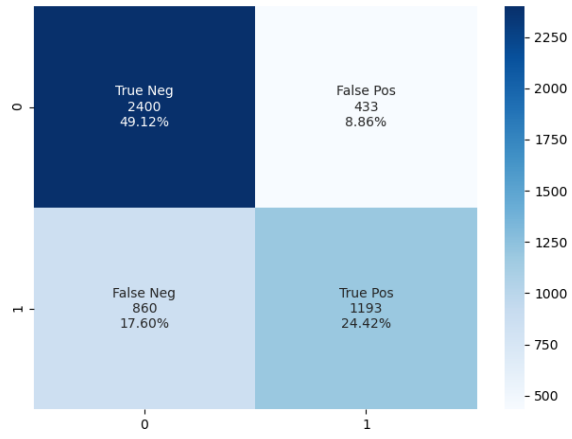


Figure 19 Confusion matrix initial model

4.3.3 Hyper parameter tuning

As presented in chapter 3.6, there are seven different parameters that require tuning. To limit the state space, following (Guillemot et al., 2019), the first step is to set some of the parameters to theoretical optimal settings. Next, a random exploration is conducted, to explore promising areas for parameters settings. Finally, a Bayesian optimization is executed to find the best hyper parameters, resulting in the most accurate model.

4.3.3.1 Selection of hyper parameters

The first hyper parameter that can be fixed is the optimizer. In general, the Adam (Adaptive Moment estimation), outperforms other optimizers, while having only the learning rate to tune (Wierenga, 2019), (Choi et al., 2019). It might be that with specific settings, other optimizers have better performance, but this does not outweigh the generally good performance provided by the Adam optimizer, which allows for better comparison of the other hyper parameter settings.

The second hyper parameter that can be fixed is the activation function of the output layer of the model. As we are dealing with a binary classification, the most suitable activation function for the output layer is the ‘Sigmoid’ function. This activation function provides a number on the interval [0, 1] depending on its guess if the TAT of a given order will exceed the agreed TAT or not, and how certain the model is. A value close to 0.5 means that the model is unsure regarding the class, a value close to one or zero indicates more confidence. For the evaluation, every output value is rounded to either zero or one and is compared to the actual class.

The third hyper parameter that is fixed, is the use of the ‘ReLU’ activation function for the other layers. ‘ReLU’ stands for Rectified Linear Unit and is the most commonly used activation function in forward feeding neural networks. Advantages of using this activation function are fast computation, simplicity in use, lack of vanishing gradient problem, and robustness. The fast computation is due to the underlying mathematics, involving only simple multiplications, whereas other functions work with natural logarithms, powers, and factors of e , increasing the computational difficulty. The simplicity in use is caused by the limited transformation incurred by the function, furthermore, the rectified linear activation function requires less transformation of the input, as the input is less bounded than with other activation functions. Finally, there is no vanishing gradient problem, as is a frequently occurring problem for other activation functions. This is due to the multiplication of layers and the output of the activation functions, the sigmoid activation function produces a value on the interval $[0, 1]$, when these are continuously multiplied, they converge to 0, making the model unable to learn.

4.3.3.2 Bayesian optimization

For the Bayesian optimization, several other parameters are fixed before the optimization starts. First, to limit the size of the model due to technical constraints, the maximum number of nodes persistent in the model is set to 1000. When this number is exceeded, the hardware requirements to retrain the model become too high for implementation at the IAC. Furthermore, the shape of the ANN is somewhat fixed, meaning that the commended triangular shape, as discussed in section 3.6.1.2 is used. Meaning that the first hidden layer is the largest, and the size decreases with each following layer. Finally, the maximum number of layers is set at 10, again to reduce the maximal hardware requirements to train the model. To evaluate the performance of settings, the model is running three times, and the average performance in terms of binary accuracy is reported. The total Bayesian optimization took 18 hours, for evaluating 110 models three times. Detailed results for each iteration of the Bayesian optimization are found in Appendix D. The best settings found result in an accuracy of 74.99%, a precision of 74.28%, a recall of 58.61%, and an AUC of 0.81, by using the following settings: dropout of 6.2%, learning rate of 0.024, total number of neurons used of 208, and a shrinkage per layer of 33%. The resulting confusion matrix and ROC Curve are presented in Figure 21 and Figure 22 respectively. On each of the provided performance indicators, this model achieves a better score. So, this model is objectively better than the previous, non-optimized model.

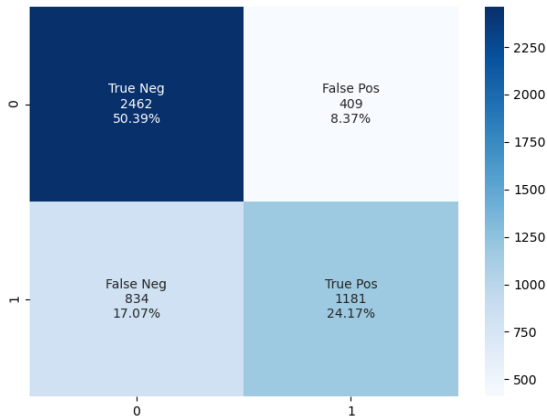


Figure 21 Confusion matrix after hyperparameter tuning

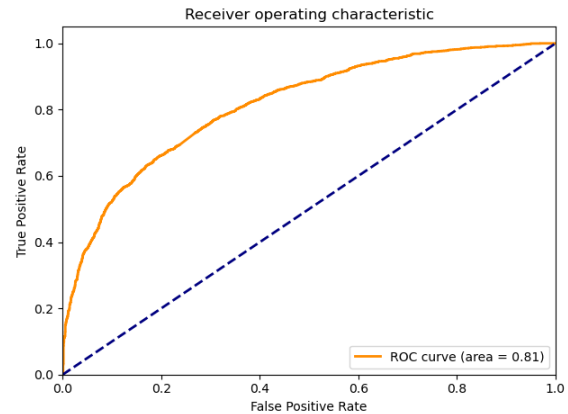


Figure 22 ROC curve after parameter tuning

4.4 Monte Carlo simulation

In this section, the alert generating model is described. The goal of this model is to identify which of the SKUs have the highest chance of facing backorders, as this is the main performance indicator for the IAC. This section is divided into four sub-sections, in which each of the components of the alert generating model is discussed and elaborated.

4.4.1 Statistical distribution turnaround times

As described in chapter 0, when the IAC delivers a forward exchange to the customer, the customer sends back a damaged part (or core unit). This core unit is then repaired, either in house, or by an external party if the IAC lacks the capabilities to perform the repair in house. After the core unit has been successfully repaired, it is sent back to the CMA inventory pool. For contracted repairs, which have exceeded the agreed turnaround time, and thus have been delivered in the form of a performance exchange, the repair will continue as a regular repair order. Therefore, these can be modelled in the same way as the returning core units are modelled.

To be able to incorporate the repair shop pipeline, the historical turnaround times of the repair shops are used. To these historical turnaround times, a statistical distribution is fitted, to model the pipeline as accurately as possible. As the distribution describes the total throughput time, it has to be slightly altered for repairs that have already been in the repair shop for, e.g. 30 days. Instead of the entire distribution, values should then only be drawn from the conditional probability distribution of $P(X=x \mid X \geq 30)$.

To simulate the turnaround time of the returning repairs, historical data is used to estimate the distribution of shop throughput times. A histogram of the distribution is found in Figure 23.

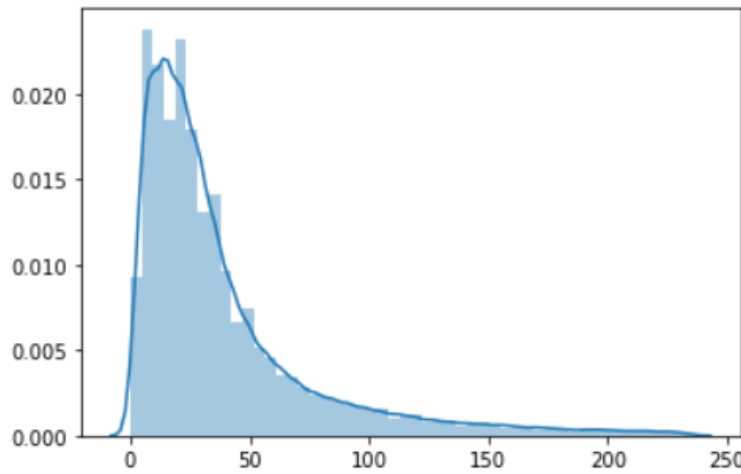


Figure 23 Histogram of historical Turn Around Times

By using the pypi fitter package, common distributions have been fitted to the historical data. These common distributions are defined by the fitter package and are: ‘cauchy’, ‘chi²’, ‘expon’, ‘exponpow’, ‘gamma’, ‘lognorm’, ‘norm’, ‘powerlaw’, ‘rayleigh’, and ‘uniform’. These are the most commonly found distributions, and thus are most likely to fit the historical data of the IAC. For each distribution, the fitter package tries to find the best combination of parameters, minimizing the sum of the square errors between the data and the fitted distribution ($\sum_i (Y_i - pdf(X_i))^2$). The results of each distribution with the best parameters are found in APPENDIX G: Distribution Fitting, in ascending order. From this table, we can conclude that the three-parameter lognormal distribution with parameters (sigma = 0.892, mean = -1.22, location = 27.0) fits the historical data the best. So, the lognormal distribution in this case is not a standardized lognormal distribution, but a lognormal distribution which has a shifted location.

4.4.1.1 Conditional probability

To ensure that turnaround times are not shorter than the already passed time, numbers drawn from the distribution are rejected if the drawn turnaround time is smaller than time the part currently has spent in repair. For repairs that have only ran for a short amount of time, this approach causes no problems. However, for repairs that are in the tail of the distribution, a lot of numbers have to be drawn before a suitable value is found, which might lead to too long runtimes of the simulation. To test this behaviour the procedure is ran for 1000 iterations for every multiple of 50 days, until turnaround time of 1000 is encountered. The probability of a repair having such high turnaround time is $P(X > 1000) = 0.025\%$. The results of this experiment are found in Figure 24, which shows the runtime in seconds of successfully drawing 1000 values for the given minimal turnaround time. Furthermore, it shows the probability of observing the corresponding minimal turnaround time.

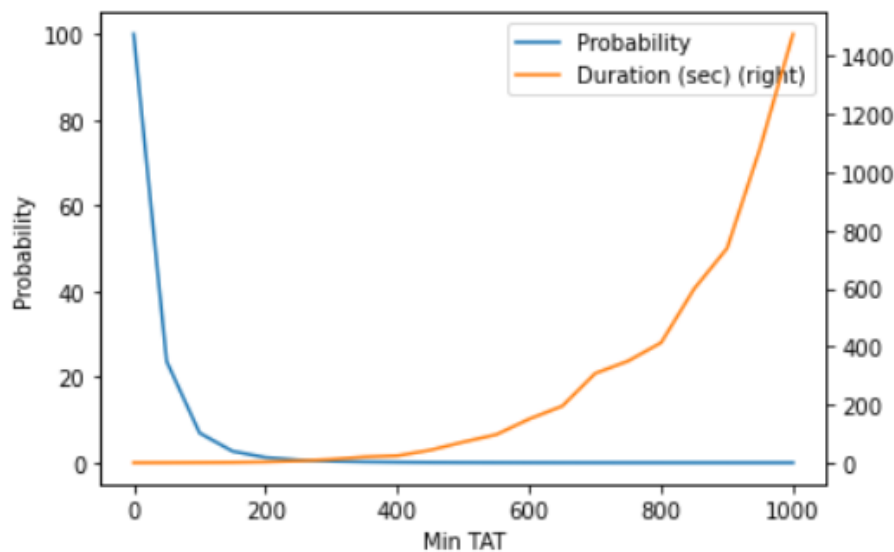


Figure 24 Runtime vs minimal turnaround time

Although the runtimes are getting rather long, when the turnaround times are getting large, the runtimes are not problematic. The tool will be running once per day, probably during the night, so slightly longer runtimes do not form a problem. Furthermore, historically, such long turnaround times have been observed mostly due to administrative negligence. Where a part was already ordered to be scrapped, but was not registered as such, resulting in seemingly long turnaround times. So, although this method of drawing conditional values might not be optimal, it is not likely to form any problems with runtimes of the tool. However, if in the future problems arise with too long runtimes, in

APPENDIX H: Drawing from conditional probability, a more complicated, yet faster approach is discussed.

4.4.2 Existing FE program demand forecasts

For the tactical planning, extensive models have already been created in former projects executed by the IAC. These models also form the input for the tactical planning of the CMA program, so using these as input ensures a well-formed alignment between the operational and tactical planning. The forecasting methods are based on the historical removal rate of components per flight hour, multiplied with the planned flight hours and the float size reported by the customers. Contractually, the customers are obliged to provide accurate predictions of their flight hours, so it can be assumed that the forecasting for the tactical planning is accurate. By using the available data on the customer flight hours, customer float size, and historical removal rates, daily demand rates can be calculated. the IAC has determined that the FE demand follows a Poisson arrival process, which means that the daily demand rate can be used to model the demand for the FE program. This can be achieved by drawing random variates from the Poisson distribution with λ = demand per day.

4.4.3 Approximate chance of backorders by Monte Carlo simulation

As an exact solution for the chance of backorders is unavailable, due to the high level of stochasticity, a Monte Carlo simulation will be used to approximate the chance of backorder for a SKU. This is achieved by simulating the inventory level over the scope of two weeks and observing in how many replications backorders occur. In this section, this approach is described in more detail and finally three example replications of a fictional scenario are discussed.

4.4.3.1 *Technical*

To approximate the chance of backorders for a SKU, a Monte Carlo simulation is executed. The input for this simulation are: ‘The current inventory level’, ‘The SKUs in the repair shop pipeline, with their current repair duration’, ‘The contracted repairs, with their due data, and the likelihood of exceeding this due date based on the classification model’, and ‘The forward exchange demand rate’. The output of the classification model, is a chance between 0 and 100%, based on the confidence of the model. The more confident the model is, the higher its output.

For 1000 iterations, for every SKU the procedure as schematically displayed in Figure 25. So, every replication the inventory level is set to the inventory level as found in the ERP system. Then the two weeks loop starts, where t represents the days.

The first step is determining if FE demand arrives, as discussed this is done by existing forecasting models, and using the Poisson distribution to draw random variates.

The second step is determining if contracted repairs have exceeded the agreed TAT, and thus have to be delivered from inventory. This is done by checking if one of the open repairs has a due date equal to t , and randomly drawing with success probability p , where p comes from the classification model.

The third step is determining whether a repair is finished, and the inventory is replenished. This is done by for every repair in pipeline, drawing a total turnaround time from the fitted distribution (section 4.4.1), and deducting the current duration of that repair. If the outcome of this calculation is equal to t , the part arrives.

Finally, the inventory level at the end of day t is calculated. This is done by taking the inventory level of the previous day, deducting the FE and PE demand, and adding the returning repairs.

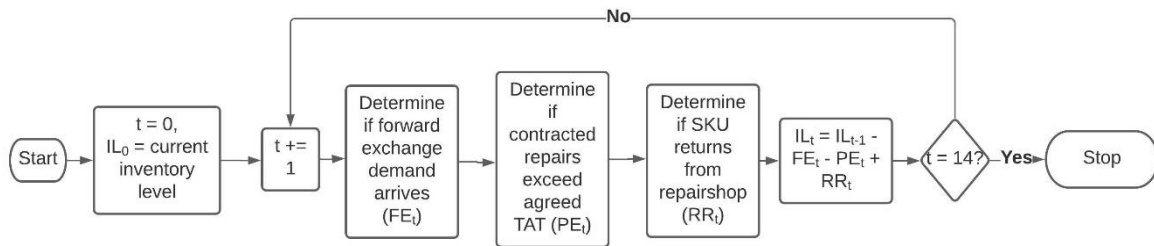


Figure 25 Replication overview Monte Carlo simulation

4.4.3.2 Examples of simulation replications

In this chapter the application of the simulation is elaborated by the use of three replications based on an example scenario.

The example scenario is as follows:

Current inventory level = 1

Parts in repair shop pipeline = 2, with the following characteristics:

Repair_1: Time in repair = 20,

Repair_2: Time in repair = 40

Nr of contracted repairs = 2, with the following characteristics:

Contracted_Repair_1: Due date $t = 5$, prediction model = 0.7

Contracted_Repair_1: Due date $t = 10$, prediction model = 0.3

Forward exchange demand = 0.07 per day

Based on these inputs, the simulation is ran for three replications, and the inventory levels over time are shown in Figure 26 Figure 26.

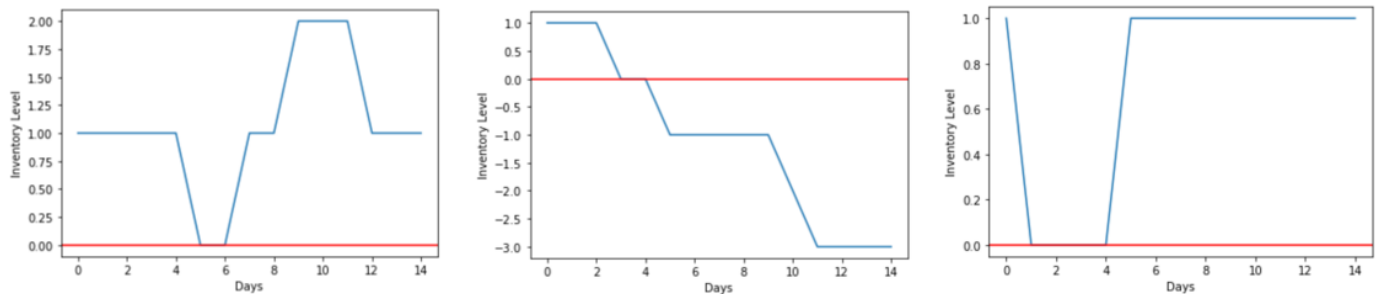


Figure 26 Three example replications of fictional scenario

In the replication, the following mutations took place during the scope of the simulation. On day 5, a performance exchange was required to meet the agreed TAT for the first contracted repair, reducing the inventory level to 0. Then, on day 7 and day 9, repairs returned from the shop and increased the inventory level to 2. Finally, on day 12 a forward exchange took place, again reducing the inventory level to 1. Thus, in this iteration, no backorders took place.

In the second replication, on day 3 forward exchange demand occurred, reducing the inventory level to 0. Then, on day 5, a performance exchange was required to meet the agreed TAT for the first contracted repair, further reducing the inventory level to -1, thus causing backorders. To make matters worse, on day 10 another performance exchange was required to meet the agreed TAT of the second contracted repair, and on day 11 another forward exchange took place. The final inventory level is -3, which means serious trouble for the operational planners.

The third replication only has two events, first, a forward exchange on day 1, reducing the inventory level to 0. Then, on day 5, a repair returns from the shop, and the inventory level again is one. Which means that in the third scenario, no problems arise.

After running this scenario for 1000 replications, the average inventory levels as shown in Figure 27 are obtained. Furthermore, in 42.7% of the iterations backorders are experienced, which indicates that problematic situations might occur in the (near) future. This is also observed in the figure, as the average inventory level is below 0 from day 14.

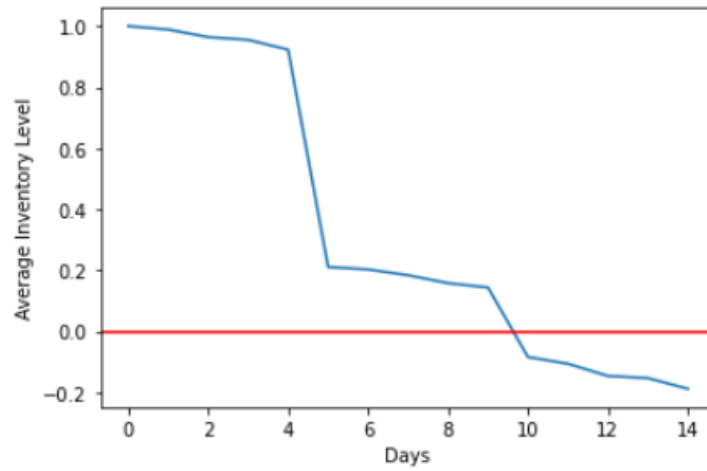


Figure 27 Average inventory level development over time

4.4.4 Determine SKUs with highest backorder chance

After running the Monte Carlo simulation for every SKU, we can determine which SKU has the highest probability of facing backorders in the coming two weeks. The operational planners are most interested, in the SKUs that have the highest probability of facing backorders. Therefore, the final output of the model will be the list of SKUs, sorted in descending order on the chance of backorders, according to the Monte Carlo simulation.

4.5 Conclusions

In this chapter, several steps are described to answer the sub-research question: *“How can machine learning algorithms, as described in the literature, be used to create a model which indicates parts most urgently require attention from the operational planners?”*. The proposed tool, as described in section 4.1, approximates the chance of backorders by simulating the development of the inventory level of an SKU, over a two week period. The mutations in the inventory level that are caused by the repair orders exceeding the agreed TAT, are predicted by a machine learning classifier. From the literature review, the most promising machine learning algorithm appeared to be an artificial neural network.

For such an algorithm to work effectively, extensive data pre-processing is required. The steps required, according to literature, are executed in section 4.2. These steps include, feature selection, reduction of missing data, noise reduction, outlier detection, data transformation and finally data reduction. After the data is cleaned, the ANN is constructed and optimized. This optimization is done by a Bayesian optimization with a three-fold cross validation. The ANN with the best performance achieved an accuracy of 75.0% and an AUC of 0.81.

To simulate the supply of the inventory pool, distribution fitting is used to approximate the turnaround time of returning repairs. This fitted distribution, together with the classification algorithm and existing forward exchange demand forecasts form the input of a simulation model. This simulation is executed to determine the chance of backorder, by determining in how many replications of the simulation backorders are experienced. Parts that have a high chance of backorders more urgently require attention of the operational planners than parts with low (or no) chance of backorders.

Furthermore, additional information on the outcome of the classification tool can be provided. The repair that has a high chance of exceeding the agreed TAT, is interesting to the operational planners, because an intervention might prevent this from happening. Furthermore, the classification model can be consulted to indicate why the model predicts that the agreed TAT might be exceeded.

5 Results

In this section, the numerical results of the research are presented. First, in section 5.1 the performance of the artificial neural network is analysed, by comparing it to a traditional random forest. In section 5.2 the features persistent in the classification model are evaluated, to see which are of greater importance, which could lead to useful insights on why the model makes certain predictions. In section 5.3 the model as a whole is validated against input of the operational planners. In section 5.4 the main conclusions of this chapter are provided.

5.1 Performance of the Artificial Neural Network

To analyse the performance of the Artificial Neural Network, it is compared to a Random Forest. The selection for the ANN was mainly based on academic interest in the topic. However, no convincing evidence was found that the ANN would outperform other classifiers, when using tabular data. As the RF algorithm is specifically designed for analysing tabular data, the RF might even outperform the neural network.

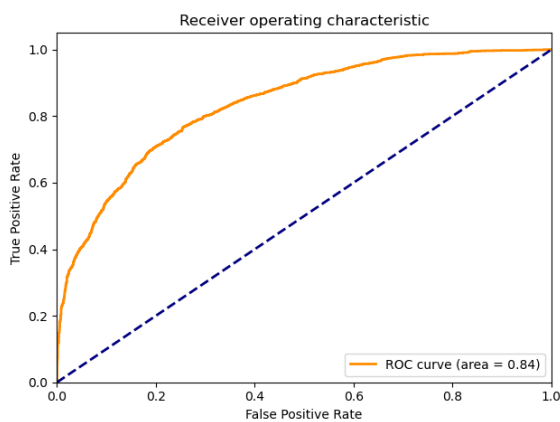


Figure 28 ROC curve optimized RF classifier

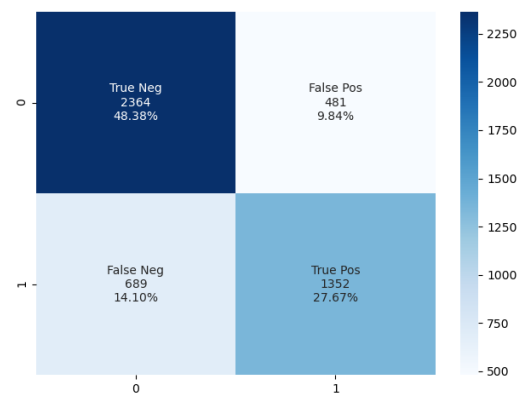


Figure 29 Confusion Matrix optimized RF classifier

As the Bayesian optimization executed for the ANN takes a lot of time, a simpler approach is selected for hyper optimization of the RF. Although this might lead to a worse RF than achievable, it still provides a sufficiently strong performance indication for the performance against the ANN. The hyper optimization approach used to optimize the RF, is a simple random exploration which is executed 100 times. Just as with the ANN, the performance for each combination of settings is cross validated three times. The results of this optimization are found in APPENDIX F: Hyper parameter tuning of Random Forest. The best RF achieved an accuracy of 75.5%, an AUC 0.84, and the ROC curve is found in Figure 28. Furthermore, in Figure 29 the confusion matrix for the best performing RF classifier is provided. The results of the best RF classifier are better than the performance of the optimized ANN. This indicates that the use of an ANN in this case might have been excessive, as similar results can be achieved with a Random Forest. Moreover, the Random Forest requires less resources to be trained and its results are better interpretable than those of an Artificial Neural Network. The hyper-parameter tuning of the RF is carried out by a random grid search, with a three-fold cross validation. The grid is described in Table 7.

Hyper parameter	Explanation	Grid values	Step size
Nr of estimators	The number of trees in the forest.	200, 1000	100
Max features	The number of features to consider when looking for the best split.	'auto', 'sqrt'	nan
Max depth	The maximum depth of a single tree.	10, 50	5
Min_samples_split	The minimum number of samples required to split a node	2, 5, 10	nan
Min samples_leaf	The minimum number of samples required to be at a leaf node	1, 2, 4	nan
Bootstrap	Whether bootstrap samples are used when building trees.	True, False	nan

Table 7 Grid optimization of random forest

Two reasons for the RF outperforming the ANN can be identified. These are the limited availability of data, and the numerical nature of the data. ANNs are known for their high requirement of data availability, to achieve a good performance. While initially a lot of data was available, during the pre-processing steps a lot of instances were dropped, as they were different kinds of orders than the contracted repairs we are investigating. Furthermore, RFs are known to achieve the best results when data are numerical, which is the case for this research. While ANNs have a broader application, RFs are likely to perform better when numerical data is used. In an empirical study by Fernández-Delgado, Cernadas, Barro, and Amorim (2014) RFs achieved the best results, when numerical data is used. So, the result that the ANN is outperformed by a RF is not surprising.

5.2 Feature evaluation

As the random forest Classifier achieved the best results and is easier to evaluate, this classifier will be used to evaluate the features. For evaluation of the feature importance, a standard feature evaluation tool pack constructed by scikit-learn (a python package) is used. This tool pack calculates feature importance as the decrease in node impurity weighted by the probability of reaching that node. The node probability can be calculated by the number of samples that reach the node, divided by the total number of samples. The higher the value the more important the feature (Ronoghan, 2018).

In Figure 30, the feature importance, with the standard deviation of this feature importance, according to the best performing RF classifier are plotted. From this figure, it can be concluded that the short-term performance and the workload (now in shop) are the most important features, providing the model with the most information. Next are the achieved turnaround times over a longer period, together with the agreed turnaround time. The final feature that is somewhat important is the long-term performance of the shop in terms of percentage of repairs finished in time. Finally, the type of repair provides some information to the model.

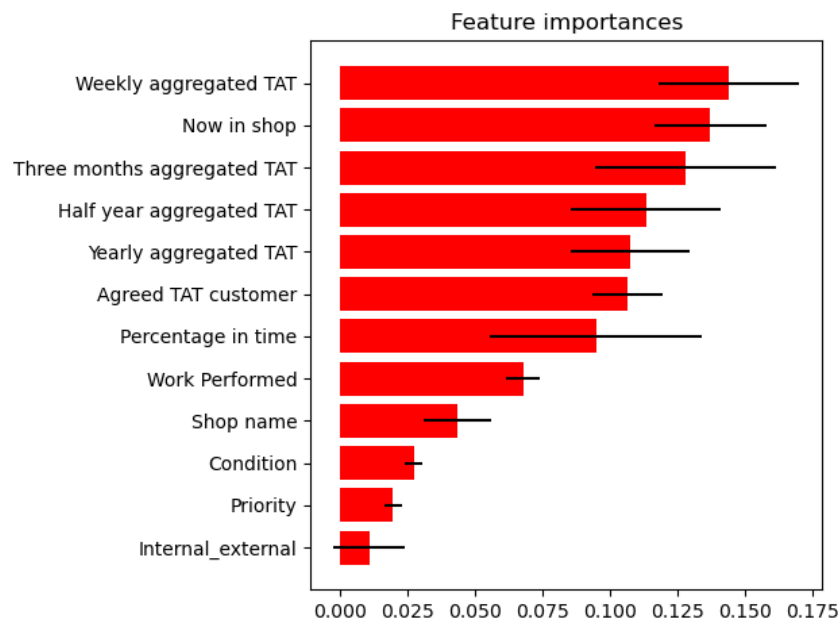


Figure 30 Feature importance Random Forest Classifier

As indicated in section 4.2.3, the individual aggregated TATs are highly correlated with the other aggregated TATs. As high correlation might cause confusion to the model, the Random Forest is retrained, but this time without the ‘Three months’, ‘Half year’, and ‘Year’ time scopes. This alteration holds slight performance increases, as the accuracy is increased to 76.68%, the precision to 68.02% and the recall to 73.04%, finally the AUC stays the same, at 0.84. The confusion matrix and the ROC curve are presented in Figure 32 and Figure 33 respectively. Moreover, the feature importance is again presented in Figure 31.

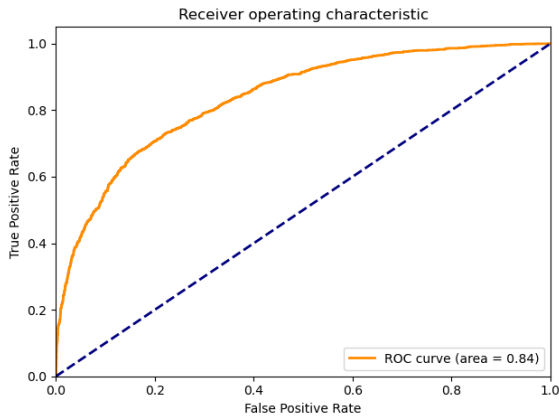


Figure 33 ROC curve optimized RF classifier, with selected features

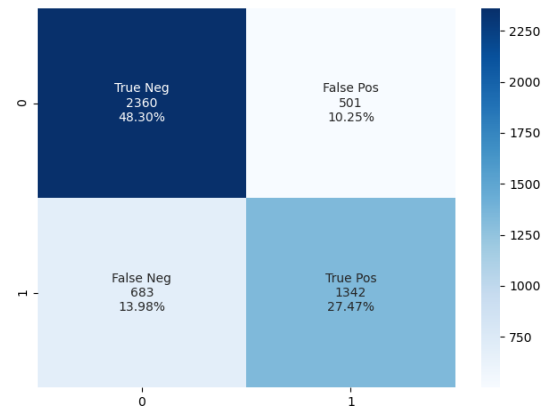


Figure 32 Confusion Matrix optimized RF classifier, with selected features

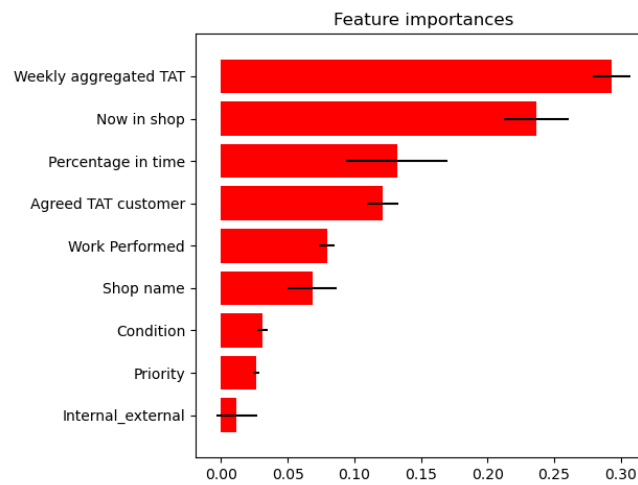


Figure 31 Feature importance Random Forest Classifier, with selected features

While the priority of the order and if the repair shop is internal or external are considered unimportant features, removing these from the model reduces the performance. When these features are removed, the accuracy drops to 74.5%, the precision to 65.2% and the recall to 71.7%, so it is objectively better to keep these features in the model. The low feature importance is likely to be caused by the method that is used to evaluate the feature importance. This method is known for prioritizing continuous variables, or categorical variables with high cardinality (i.e. high number of categories).

Finally, a random tree from the forest, in which the max depth = 3 is presented in Figure 34. A decision tree should be interpreted in the following way. In the three top rows, a decision criterion is defined. This criterion shows based on which feature, and the value of this feature, whether the evaluated instance moves to the left (criteria is true) or right (criteria is false). The next value, the 'Gini' (Gini impurity) is a metric that measures the probability from a randomly chosen element to be incorrectly classified (i.e. the probability of choosing an element times the probability of begin misclassified). Next, the samples value indicates how many instances were (during the training stage) evaluated at that particular node. The 'value' row provides an array, which describes how many of each class are in the sample. Finally, in the bottom row, the final guesses for the class are provided.

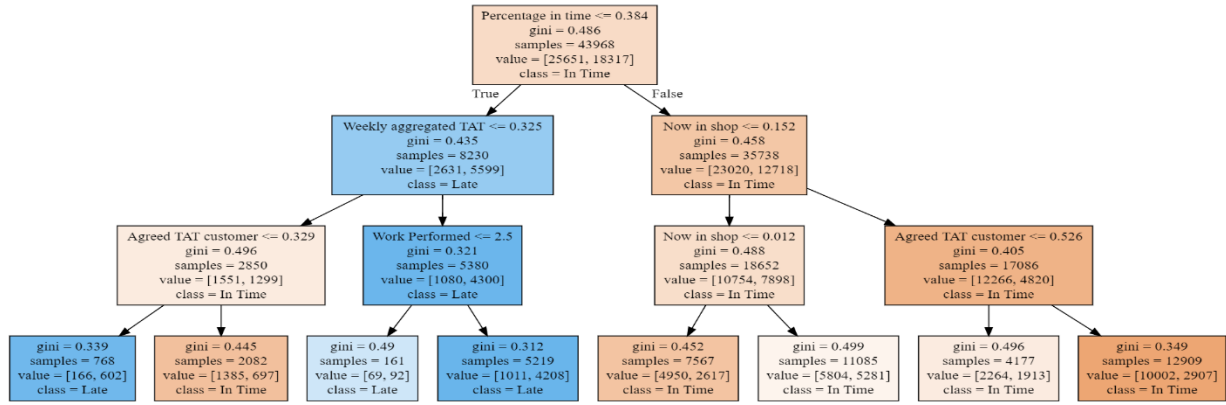


Figure 34 Example Tree with fixed depth (max_depth = 3)

5.2.1 Weekly Aggregated TAT

The most important feature, for decision making of the classification model, is the short-term aggregated TAT (over a week). In Figure 35 this relationship is visualised, and it is clear that there is a relationship between the short term performance of the repair shop, and the likelihood that the order will be finished in time.

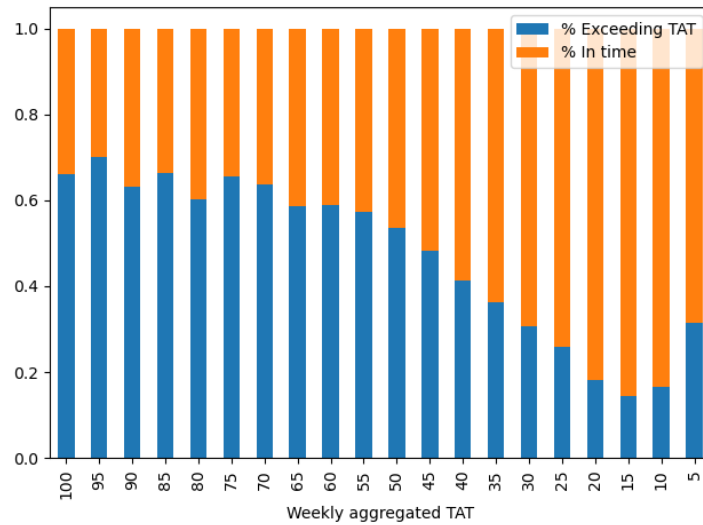


Figure 35 % of orders exceeding agreed TAT, based on short term shop performance

5.2.2 Agreed TAT customer

The second most important feature is the Agreed TAT with the customer. From Figure 36 it is easily observed that a shorter agreed TAT, leads to a higher percentage of orders that exceeds this TAT. This is an intuitive result, as with shorter agreed TATs, less room for error persists. On the other hand, this is an important result, as sales keeps setting out contracts with shorter agreed TATs. Therefore, the number of orders exceeding the agreed TAT is likely to increase, increasing the demand on the CMA inventory pool.

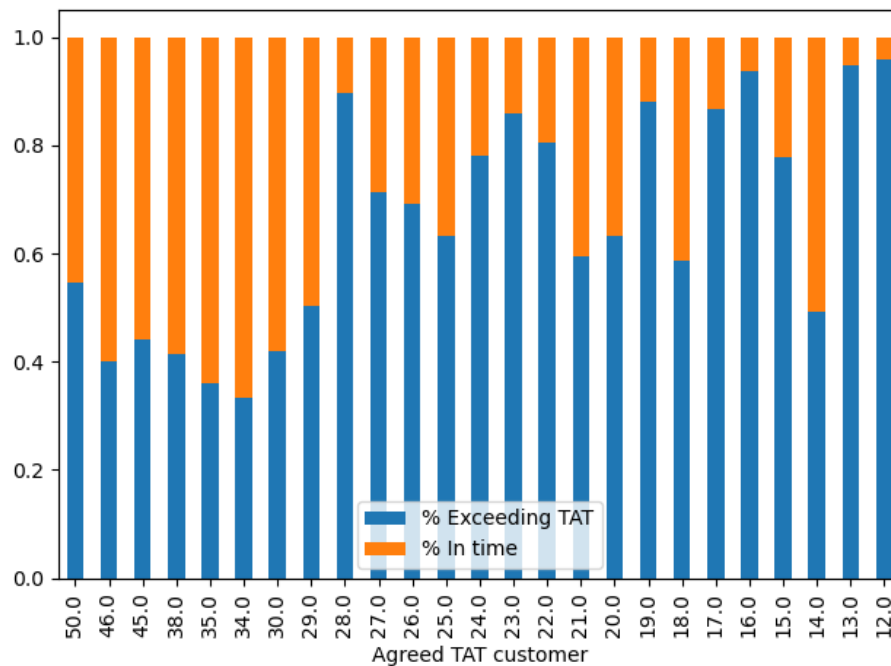


Figure 36 Percentage of orders exceeding agreed TAT by Agreed TAT

5.3 Validation of the alert generating model

5.3.1 Original validation approach

The original intention for validation of the model, was to bring the model live and let the operational planners use the tool in parallel with their normal way of working. For this to work, they would first use their usual approach to find potentially problematic situations and document these situations. Then they would run the simulation and identify which situations are highlighted by the model. Now they can document which situations are selected by the model, and classify them into the following categories:

- I. The model correctly identified a potentially problematic situation, which was not identified by the operational planners.
- II. The model correctly identified a potentially problematic situation, which was also identified by the operational planners.
- III. The model identified a potentially problematic situation, which upon investigation appeared to be incorrect.
- IV. The model missed a situation that was marked as potentially problematic by the operational planners.

These categories provide information on how accurately the model can predict problematic situations, and where improvements lay (i.e. is the model under- or oversensitive?). Next to verifying the accuracy of the model, this approach would also be able to identify what kind of situations cause the model to fail. This knowledge would be valuable for providing recommendations for future research and improvements of the model.

Moreover, this approach obliges the operational planner to use the tool. So, they could also provide feedback on the usability, dashboard design, and design of the tool as a whole. Although the operational planners were still able to provide feedback, based on the proposed dashboard design. Some design flaws only become apparent once the tool is used, not when the dashboard is evaluated.

5.3.2 Challenges in validation of the model

Unfortunately, a few setbacks were faced during the validation of the model. These challenges and their impact on the validation model are discussed in this section.

5.3.2.1 Covid-19

The largest impact, which left a mark on the entire Thesis, was the Covid-19 virus. This virus impacted the entire way of working for the IAC and required everyone to work from home. This made discussions and receiving feedback more difficult. Furthermore, during the writing of the first chapter, the operational planners had limited time available for interviews. At that time, they were busy with adjusting the CMA program to comply with the newly created regulations and ensuring the continuation of business for the IAC.

Many sectors are impacted by the virus, but the aviation industry is one of the industries that is hit the hardest. Therefore, the current way of doing business for the IAC is nowhere near the pre-Covid way of working. The repair shops have nothing to do, and the airlines that do use a small portion of their float try to delay repairs as much as possible. So, the revenue of the CMA program is currently near zero. Therefore, a large part of the CMA inventory is currently being sold, as the IAC requires cash so they can stay in business.

As no repairs are being executed, and the FE demand is currently (near) zero, validation against the present situation is of no use. For every part, the situation will be similar i.e. no open repairs with contracted throughput times, no returning parts, and a relatively high on hand stock. So, even though the model would perform perfectly as no potential problematic situations exists, and no situations are identified by the model, this result is useless.

Thus, we have to come up with another way to validate the quality of the model. A proposed way is bringing the tool live on the development server of the IAC, and instead of using present data, use data from a year ago. Now the same approach as discussed in section 5.3 can be used. The only drawback is that more time from the operational planners is used as instead of assisting them in doing their job, they now actively have to validate the model. However, this approach also faces challenges which were insurmountable. These challenges are discussed in the next section.

5.3.2.2 Bringing the tool live

For implementation for tools, such as the one constructed in this Thesis, the IAC has a development platform. This is a virtual server used for advanced analytics projects. On this platform, Application Programming Interfaces (APIs), Web applications, Interactive Notebooks and Static Notebooks can be hosted. The data hosted on this server is obtained by constructing custom data pipelines from the ERP system to the virtual server.

The data required for the model, is not yet available on the server and thus the pipelines still have to be built. As the advanced analytics team is busy with other projects, they indicated that several weeks, maybe months were required before the data is fully available. Only once the data is available on the server, the development of the tool on the server can start. Otherwise there is no way to debug or verify any sub-stages of the project. When the tool is finally live, the operational planners can start with validation of the tool using historical data. However, this entire process will take way too long to be part of this Thesis.

So, another way should be constructed to validate the model. The newly proposed method is using the available dataset and trying to find for which orders interventions took place. If the model identifies problematic situations for parts which belong to orders for which interventions took place, the model is working correctly. Unfortunately, this approach also appeared inappropriate, as discussed in the next section.

5.3.2.3 No data on historical interventions

As the title indicates, identifying for which orders interventions took place turned out to be impossible. The first problem is that interventions are not documented, which is mainly the case with cheaper interventions such as prioritizing of orders in internal shops. More expensive interventions are documented but happen to rarely to validate the model to. Secondly, interventions are not always successful, so sometimes turnaround times can be longer than usual, even though the operational planner asked to prioritize the order. On the other hand, sometimes orders are finished very quickly, without any intervention by the operational planner. Finally, data on the historical inventory levels are unavailable. So, observing when backorders occurred, and using this identification of problematic situations is also not possible. In conclusion, this approach is too inaccurate to provide a good validation of the alerting model.

5.3.3 Description of the validation approach

In this section the selected validation approach is discussed. As mentioned, this way of validation was definitely not the first choice, but one of the only possible ways. The selected approach is creating hypothetical scenarios based on historical data. These scenarios are then evaluated by both the alerting model and the operational planners and the tactical planner. The results of these evaluations are then compared, to see if the model acts similar to the operational planners. Each of these steps is discussed in more detail in the following subchapters.

5.3.3.1 Creation of scenarios

For validation of the model, a ‘present’ situation artificially created. To do this, for 50 SKUs random situations are constructed. These situations are based historical information but are not real historical

events. Again, the historical inventory levels are unavailable, so therefore alternative means have to be found. Each scenario consists of four parts, which are ‘Contracted Repairs’, ‘Returning Repairs’, ‘On hand stock’, and ‘Forward Exchange Demand Rate’.

5.3.3.1.1 Contracted Repairs

The contracted repairs are randomly drawn from the validation dataset, which was used to analyse the performance of the ANN in chapter 5.1. The use of data from the validation set, ensured that the model did not ‘remember’ a repair, and thus would be able to perfectly predict the outcome. For each part, uniform a number on the interval [1, 4] was drawn, which represents the number of contracted repairs that was currently in the IAC repair shop. These repairs are then randomly selected and removed from the sample, so no identical cases could exist. Next, for these each of these repairs a number on the interval [1, 14] is uniformly drawn. This number represents the due date (given $t = 0$).

Furthermore, the operational planners received the same information as the model was given to predict if the repair would be finished in time or not. So they received the following metrics ‘Shop Name’, ‘% in time delivery’, ‘The agreed turnaround time (with the customer)’, ‘Workload at shop’, ‘The work scope’ and the four aggregated historical turnaround times achieved by the relevant shop.

5.3.3.1.2 Returning repairs

For the returning repairs, the fitted distribution as discussed in section 4.4.1 is used. From this distribution, times are drawn, which represent the current duration of the returning repair. This metric is mostly of value to the model, as the operational planners, until execution of this project, had no idea of the lognormal distribution that the turnaround times followed. How long a repair has been in the repair shop, gives an indication for the expected return time, but due to the stochasticity, absolutely no certainty.

5.3.3.1.3 On hand stock

The on-hand stock is simply the stock at the start of the two-week period. The on-hand stock is an important metric, as the operational planners are very risk averse. Seeing that the on-hand stock is zero or one in their eyes is a guaranteed ‘prio 1’, except when the FE demand for that part is little. On the other hand, if the on-hand stock is too high, the scenario is of little interest, as chance of backorders is neglectable. So, to ensure that only somewhat interesting scenarios are created, the on-hand stock is drawn uniform on the interval [0, 5].

5.3.3.1.4 Forward Exchange demand

The final metric provided to the model and the operational planners is the forward exchange demand (in demand rate/two weeks). This way of presenting the FE demand is selected to ensure understanding of the metric. They stated that the demand rate per day confused them, causing them to make misguided decisions.

5.3.3.2 Evaluation by operational planners

Each of the 50 scenarios is evaluated by the two operational planners and a tactical planner. To reduce the bias, at first three scenarios were discussed plenary, to ensure that everyone correctly interpreted the provided metrics. Then they were provided with 50 newly constructed scenarios, which they had to analyse individually. For each scenario, the planners had to give an urgency score on the interval [1, 3]. Where one means extremely urgent, and five means perfectly fine.

5.3.3.3 Compare the outcome of the operational planners with the outcome of the model

The main output of the model is the percentage of iterations in which backorders were experienced. The planners, on the other hand, provided a score between one and three to indicate the urgency of the situation. Although the percentage of iterations in which backorders occur provides an indication of the urgency of the situation, they are not directly comparable. So, a translation function has to be determined to allow a fair comparison between the model and the output of the planners.

5.3.3.3.1 Transformation of model output

To match the output of the questionnaire, the output of the model also has to be transformed. As the model is created to assist the operational planners, their preferences have to be incorporated in the scoring. They indicated that a situation with >10% chance of backorders in their opinion is urgent. On the other hand, situations with <1% backorders are not relevant. Based on these thresholds, the following transformation is applied to the output of the model:

% chance of backorders > 10% → 1

1% < % chance of backorders < 10% → 0.5

% chance of backorders < 1% → 0

5.3.3.3.2 Key performance indicators

The performance of the model is measured in mean absolute deviation between the average score provided by the planners and the output of the model. A low mean absolute deviation shows that the model and the planners agree on which situations require an alert, a notification, or neither. Moreover, a correlation matrix is constructed to measure the correlation between the model and the decisions of the planners, as well as the correlation between the different planners. A higher correlation coefficient suggest that the model produces similar results to evaluation by the planners.

5.3.4 Validation results

In Figure 38 a high correlation between the model (Score Model), the scores of the operational planners (Score OP1 & Score OP2), and the score of the tactical planner (Score TP) can be observed.

	Score OP1	Score OP2	Score TP	Score Average	Score Model
Score OP1	1				
Score OP2	0.789708641	1			
Score TP	0.803885241	0.816804216	1		
Score Average	0.935799821	0.920826352	0.938602988	1	
Score Model	0.740489074	0.700134042	0.725649221	0.77630591	1

Figure 38 Correlation Matrix Validation Results

Furthermore, the correlations between the different planners are also high, indicating that none of the planners had difficulties with understanding the models, or made significantly different decisions than the others. As the correlation between the model, and the decisions of the planners is high, it is likely that the model identifies the same situations as urgent as the planners. The t-statistic for the correlation coefficient between the average score of the planners and the model is $t = 8.53$, which suggest strong signification correlation between the output of the planners and the model. In Figure 37, the deviation between the output of the model and the planners per scenario can be observed.

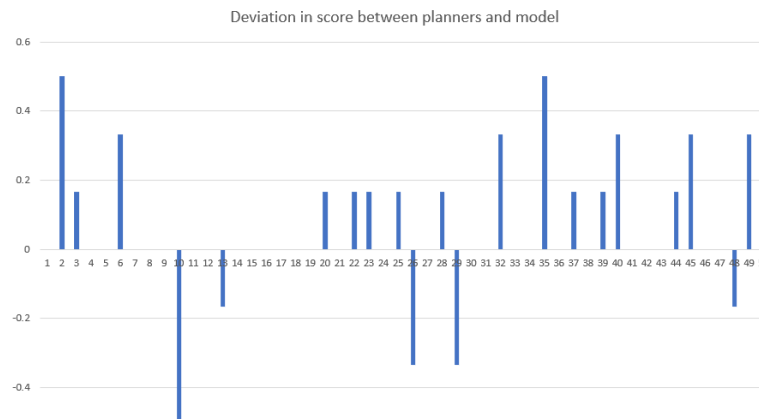


Figure 37 Deviation in score between planners and model

The deviation is calculated as follows $\Delta = \text{Avg score planners} - \text{score model}$. The delta between the two is mostly positive, meaning that the model is more likely to underestimate the urgency than overestimate. The mean absolute deviation is 0.113, which demonstrates that the model in general has a good performance, as its decisions are in line with the decisions of the planners.

To compare this outcome with the existing alert generating model, the output of the planners is also compared to the outcome of the existing alert generating model. The existing alerting model is a quite simple model based on decision rules. A detailed description of the tool is given in the description of the current situation (section 2.6). To compare the performance of the newly constructed model and the existing model, the existing model is used to generate alerts for the 50 scenarios that are used for the validation. As the scenarios provide no information on the historical backorders for the part number, every product is assumed to have experienced backorders in the last few years. This assumption is only invalid for few parts, as about $\frac{1}{3}$ of the orders has been delivered late.

To compare the performance of the existing alerting model with the questionnaires and the new model, the outcome of the existing alert generating model needs to be transformed. As nearly every part has experienced backorders in the last three years, only three priorities are given by the model. These are transformed as follows: Prio 1 \rightarrow 1, Prio 2 \rightarrow 0.5, No Prio \rightarrow 0. When these results are compared to the average urgency provided by the planners, the mean absolute deviation (MAD) is 0.207, which means that the MAD of the new model is compared to the old model $\frac{0.113-0.207}{0.207} * 100 = -45.4\%$. So, the new model is seemingly more in line with the judgement of the planners.

Finally, the scenarios where the highest deviation occurred between the evaluation by the planners, and the alert generating model, are briefly reviewed to see if potential points of improvement can be identified. The scenarios (as shown in Figure 37) that will be discussed are 2, 10, and 35. The main findings from analysing these scenarios are:

While the model only evaluates *if* a stockout is likely to occur, the planners also consider *when* a stockout will occur. So, when problems arise at the end of the planning horizon, they receive lower priority, than when a stockout might occur in the near future.

Furthermore, the pipeline for parts that have spent a long time in the repair shop is considered differently by the model and the planners. For example, when a part has spent more than 100 days in the planner's reason something is wrong with the part and assume that it will not return within the planning horizon. However, the model uses the conditional probability, as described in section 4.4.1.1, for approximating the pipeline. Although the long tail of the distribution incorporates the assumption by the planners that a part with a long repair duration is expected to take even longer, it is not a hard constraint.

5.3.5 Limitations to validation approach

As discussed, the used form of validation was not the first choice, but one of the few available options. This is due to the large number of limitations that this form of validation has to cope with. These limitations are elaborated in the following sections.

5.3.5.1 Scenarios an approximation

The first limitation is that the used scenarios are an approximation, and not real historical situations. Therefore, sometimes unrealistic scenarios were created, which would never actually be faced. For example, the IAC currently tries to send all parts with the same part number to the same repair shop. However, as repair shops sometimes go out of business or new repair shops are started, historically not every part was repaired by the same repair shop. As the validation model draws from all historical repairs for a certain part, the scenario that three parts with the same part numbers are repaired by three different shops. In reality the IAC would send all these repairs to the same shop, and probably make an agreement such that all parts are shipped in the same batch. This deviation from reality might have caused confusion with the planners, causing them to overestimate the urgency for a situation

5.3.5.2 *Limited number of scenarios*

The evaluation of scenarios by the planners was quite time consuming. Therefore, only 50 scenarios have been created and evaluated, providing a rather small sample to compare the model with. Having a smaller sample size, reduces the statistical power of the validation, as the obtained results are more likely to be observed by chance. So, although signals have been found that the model provides useful alerts and notifications, and ignores the correct situations, hard conclusions cannot be drawn.

5.3.5.3 *Limited time available*

The final limitation is that planners were busy, and thus had limited time available. As they did not want to wait too long with providing their results, they might sometimes have been too quick with drawing conclusions. This would mean that for some of the scenarios they made incorrect decisions. Although the impact of this limitation is limited by taking the average score of the three planners, it might still have an effect on the final score of the model.

5.4 Conclusions

For validating the performance of the artificial neural network, a random forest has been constructed. This random forest outclassed the ANN by a small margin, which is surprising as a lot more advanced and time-consuming hyper parameter optimization methods have been used for the ANN. Furthermore, the input features of the classification model have been evaluated, and the most important features seemed to be the current workload, together with the agreed turnaround time with the customer. Furthermore, the highly correlated aggregated turnaround times over different time periods were found to be confusing to the model, and only incorporating the short-term performance of the repair shop yields better classification results.

Next to the more proactive outlook of the newly constructed model, several other benefits have been identified. The first benefit comes from time savings for the operational planners. Currently, reasonable time is spent by the operational planners to diagnose situations that require their attention. However, as the alert generating model is able to identify a large portion of the urgent matters, less time might be required by the operational planners. Although fully relying on the model is currently too premature, as the model is not tested in a live situation, the results of the validation are promising. If the model is able to correctly classify potentially problematic situations, and has an extremely low false negative rate, operational planners only might have to deal with the situations provided by the model.

Furthermore, the model is able to identify problems that conceivably occur in the future. In chapter 2 we have seen that intervention costs increase with later discovery of problems, so early pinpointing of probable issues could potentially save the IAC operational costs. If problems are discovered in an earlier stage, the range of interventions that can be used is larger, and thus cheaper alternatives can be used. Unfortunately, no data is available on the cost of interventions, so quantification of the benefits is not possible.

To evaluate the quality of the alert generating model, two operational planners and one tactical planner were asked to evaluate 50 randomly generated scenarios. Their evaluation was then compared to the existing alert generating model, and the newly constructed alert generating model. The mean absolute deviation between the output of the planners, and the new model was 45% lower than the old, reactive model. Showing that the new model is more in line with the estimates of the planners, and thus is more suitable for providing alerts to the operational planners.

6 Conclusions, limitations, and recommendations

In this chapter, in the conclusion section, the main research question “*How can the Independent Aerospace Company construct a proactive alerting tool, which automatically recognizes and prioritizes potential problematic situations and notifies the operational planners?*” is answered. Furthermore, by discussing the limitations to my research, and the recommendations to future research and projects by the IAC, I will answer the final sub-research question “*How can the Independent Aerospace Company leverage a pro-active alert generation model in practice and which additional steps are required to achieve this?*”.

6.1 Conclusion

6.1.1 Current situation

The need for a proactive alerting tool, mainly comes forward out of the discrepancy between the existing alerting tool and the current situation. The existing alerting tool is old and has been developed during times when the majority of demand came from forward exchanges. However, due to shifts in the market, the number of forward exchanges is decreasing, while the number of performance exchanges is increasing. These performance exchanges are not incorporated in the existing alerting tool, meaning that demand is generally underestimated. In turn, this underestimation of demand, results in late alerts, mostly when things already are going wrong. Due to this late notification, the operational planners have to resort to expensive interventions to resolve the problematic situation.

6.1.2 Prediction of Performance Exchange demand

Incorporation of the performance exchange demand is one of the most important improvements that need to be applied to the alerting tool. Based on interest by the IAC, my company supervisor, and myself, the desired way to predict performance exchanges is by use of machine learning algorithms. From the literature study, a binary classification model, specifically an Artificial Neural Network or a Random Forest, seemed the most promising. These classification models will be used to predict if the IAC will manage to repair a part within the agreed TAT, because if the IAC does not manage to do so, a performance exchange is required to meet contracted agreements. Both the Artificial Neural Network and the Random Forest achieve an accuracy of 75%, but the RF performs slightly better on the receiver operating characteristic (ROC) curve, which is an indication for the trade-off between the precision and the recall.

6.1.3 Simulation of demand and supply

The proactive alerting tool itself is based on a simulation of the demand and the supply of the inventory pool. The demand consists of the forward exchange demand, forecasted by existing tactical planning models, and the repair orders resulting in a performance exchange, predicted by the machine learning algorithm. For the supply, a shifted-lognormal distribution is fitted to historical turnaround times, to simulate the total turnaround time, given the current repair time, of repairs that are returning to the IAC inventory pool. This simulation simulates the inventory level during the coming two weeks, to determine the chance of backorder, by determining in how many replications of the simulation backorders are experienced. Parts that have a high chance of backorders more urgently require attention of the operational planners than parts with low (or no) chance of backorders.

6.1.4 Results

To evaluate the quality of the new alert generation model, 50 scenarios are created based on historical transactions and orders. These were analysed by two operational planners and a tactical planner, to determine how urgent the situations were. The average judgement of the planners was then compared with the output of the new model, and the output of the old model. The mean absolute deviation between the output of the planners and the new model was 45% lower than the mean absolute deviation between

the planners and the old reactive model. Showing that the new model is more in line with the estimates of the planners, and thus is more suitable for providing alerts to the operational planners.

The main advantage of the new alerting model arises from situations that are currently fine, due to potential requirements for performance exchange, face potential problems in the future. The existing model would use the current inventory level combined with the inventory position and decide that everything is fine. The new model, on the other hand, predicts if the agreed TAT of the contracted repairs will be achieved, and uses this as additional input to make decisions. While the old model would only provide an alert once the performance exchange has been performed, and the inventory level drops below a certain threshold. The new model would already provide an alert upfront, allowing the operational planners more time to use interventions, and thus use a wider scale of available interventions.

6.2 Recommendations

The recommendations section is divided into two parts, the first are recommendations for further research or to further improve the model. The second are recommendations to the IAC, based things that caught my attention during my internship.

6.2.1 New directions

6.2.1.1 *Inclusion of core receive times*

The first recommendation for further research and improvement, is the inclusion of the core return times. These are completely neglected in my model, limiting the time scope of the simulation. The returning core units, sent by a reliable customer, take approximately 10 working days before they arrive at the repair shop. Therefore, as long as the time scope is limited at two weeks, the model is accurate. After this time, the core could have entered the repair shop and the repair could even have started. Leading the model to underestimate the supply of the inventory pool, thus causing the model to be conservative.

6.2.1.2 *Use of repair shop data*

Another recommendation is the use of repair shop data. Although external shops are reluctant to share data, the internal data should be available. From this data, important determinants for turnaround times can be deducted, such as:

- I. Availability of shop replaceable units
- II. Number of items in queue
- III. Scheduled date for repair

This data can all be used to determine if a repair will be finished in time, and thus if a performance exchange is likely to happen or not.

6.2.1.3 *Customer requirement of Performance Exchange*

Currently, the model assumes that every contracted repair that is exceeding the agreed TAT, will result in a performance exchange. In reality this is not the case, as customers get the option to obtain a performance exchange, but do not have to accept this. Several reasons can be found for customers rejecting performance exchanges. For example, the customer wants his own part back, and not an exchange part. Another reason might be that the customer wants to create goodwill, such that in the future the IAC might return the favour.

The performance exchange program is based on the idea that the customer always has line replaceable units available, as long as the repair is registered in time. So, if customers are willing to share data on their current inventory level, the IAC could use this information to determine if performance exchange is required or not. Less performance exchanges lead to lower operational costs, which could also benefit

the customer, if these lowered operational costs could mean lower fees for the performance exchange program.

6.2.1.4 Precision and recall trade-off

In this thesis, no preference is given to either precision or recall. However, this could be an interesting extension. The decision which of the two is preferred, depends on the costs of missing an alert, compared to falsely generating an alert. As the costs were unavailable for this thesis, this decision could not be made, but for further research finding this balance could reduce overall operating costs.

6.2.2 Practical recommendations

6.2.2.1 Keep track of used interventions

In conjunction with this thesis, another graduation assignment is currently being executed at the IAC. This assignment looks into the possibility of automated decision making for the selection of interventions. However, as the use and costs of interventions are badly documented, these assignments are unnecessary challenging. Furthermore, quantification of results is not possible, as no specific costs are defined per intervention. Even if these are dependent on a lot of factors, documenting them could still provide interesting insights, maybe by creating a regression model that predicts the costs and chance of success given an intervention, time scope and part number.

6.2.2.2 Alert shops instead of operational planners

The final recommendation is that the alerting tool in this thesis is created for the operational planners, but in the view of a SCT, the alerts would ideally directly go to the (internal) repair shops. A well-functioning alert tool would save the operational planners a lot of work. Currently, on a daily basis the operational planners are making and sending out priority lists to the internal repair shops. This process, however, can be fully automated if the shops see for which parts they are currently repairing, problems are arising.

6.3 Discussion

6.3.1 Contributions

Four practical contributions are presented to the Independent Aerospace Company. First, a machine learning method is constructed, which allows the prediction of meeting the agreed TAT. With an increasing number of customers using contracts with fixed TATs, the need for accurate predictions is increasing. Furthermore, as the Independent Aerospace Company previously had minor insight in what leads to underperformance, this classification model is a good first step. Secondly, insight is provided into which characteristics are often a cause for exceeding this TAT. With the sales teams promising increasingly competitive turnaround times, a useful finding has been that with truly short turnaround times, the turnaround time is almost always exceeded. Fourth, a methodology is described, which allows the approximation of the chance of backorders. Although there is still room for improvement, the model performs in line with the desires of the operational planners.

6.3.2 Limitations

6.3.2.1 *Noise in dependent variable*

After creation of the ANN for classification and construction of the entire simulation, additional noise in the dependent variable was found. This noise has to do with interventions executed by the operational planners, reducing throughput times of contracted repairs. Therefore, the model might sometimes correctly predict that the agreed TAT will be exceeded, but due to intervening by the operational planners, this did not happen. This causes confusion to the model, as almost identical situations have different class labels.

6.3.2.2 *Over-interpolation*

For obtaining the historical, aggregated turnaround times of shops, a lot of interpolation took place. For shops with a lot of orders, this caused no trouble. However, for shops that are used only several times per year, over-interpolation might have taken place, providing confusing data to the classification model. Furthermore, this way of interpolating created high correlation between the aggregated turnaround times over different time scopes, as for some shops only once every three months a repair was completed, meaning that the weekly, and three month TAT would be exactly the same.

6.3.2.3 *Outlier detection with statistical methods*

Due to time limitations, outlier detection only took place using statistical methods. The drawback of using these statistical methods, is that they are only capable of identifying univariate outliers, while the unsupervised learning methods would also be capable of identifying multivariate outliers. For future improvement of data quality, outlier detection by unsupervised learning could be a good addition.

6.3.2.4 *Limited availability of data*

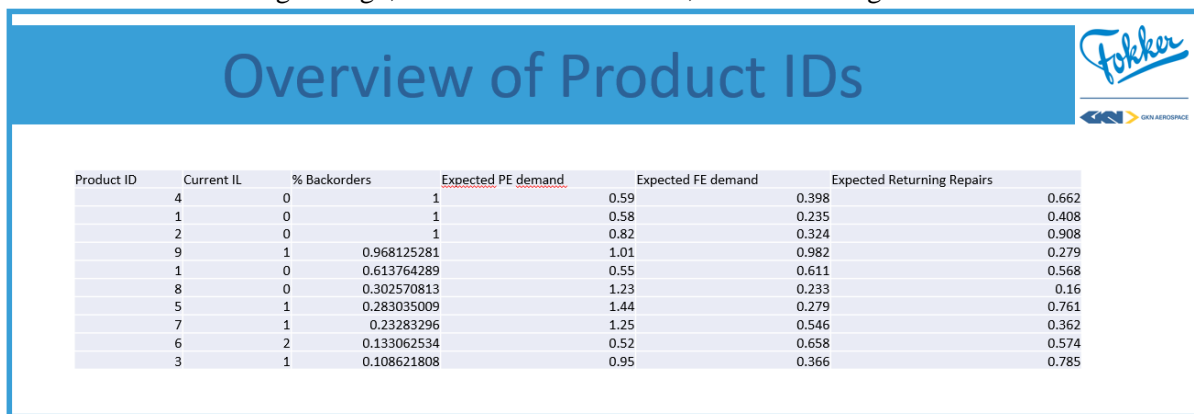
While the IAC has a lot more data than is used in this Thesis, this data was not available to me. Most of the data is stored within the ERP system of the IAC, which can only be reached if the IAC workstation is used, or the workstation is connected to the IAC network. Due to Covid-19, I was obliged to work from home, and as a graduation intern, was not provided with an IAC workstation. Therefore, the only data that was available, is based on a dataset which has been drawn from the ERP system. If in a later stadium the use of other interesting data was identified, this could no longer be obtained.

6.4 User interface design

In this section the proposed design for the user interface (UI) of the tool is discussed. Note that the data shown in the designs are completely fictional and can thus be a bit unrealistic. Furthermore, the what-if scenarios that are elaborated in more detail are just to provide an idea of how the what-if analysis works. The design is divided into three parts, which are the general overview, the part number overview, and the what-if analysis interface. These three parts are further elaborated in the remaining sections of this chapter.

6.4.1 General overview

The first screen the operational planners see when they open the tool, is the general overview. Here all part numbers are shown, sorted on the percentage of replications in which backorders occurred during the simulation. The rough design, filled with fictional data, is shown in Figure 39.

The image shows a screenshot of a software interface titled "Overview of Product IDs". It features a table with columns for Product ID, Current IL, % Backorders, Expected PE demand, Expected FE demand, and Expected Returning Repairs. The table contains 15 rows of fictional data. In the top right corner, there is a logo for "Fokker" and a smaller logo for "GKN AEROSPACE".

Product ID	Current IL	% Backorders	Expected PE demand	Expected FE demand	Expected Returning Repairs
4	0	1	0.59	0.398	0.662
1	0	1	0.58	0.235	0.408
2	0	1	0.82	0.324	0.908
9	1	0.968125281	1.01	0.982	0.279
1	0	0.613764289	0.55	0.611	0.568
8	0	0.302570813	1.23	0.233	0.16
5	1	0.283035009	1.44	0.279	0.761
7	1	0.23283296	1.25	0.546	0.362
6	2	0.133062534	0.52	0.658	0.574
3	1	0.108621808	0.95	0.366	0.785

Figure 39 UI design general overview

As can be observed, the general overview consists of one table, which shows which parts require the most attention by the operational planners. Furthermore, the current inventory level, together with the demand and supply are presented. These allow the operational planners to quickly pinpoint what might be the cause for the potential backorders. By double clicking on one of the parts, they will go to the overview of a specific part number.

6.4.2 Part number overview

After double clicking on part in the general overview, the operational planners will reach the overview at product level, as show in Figure 40. This screen provides more detailed information on which

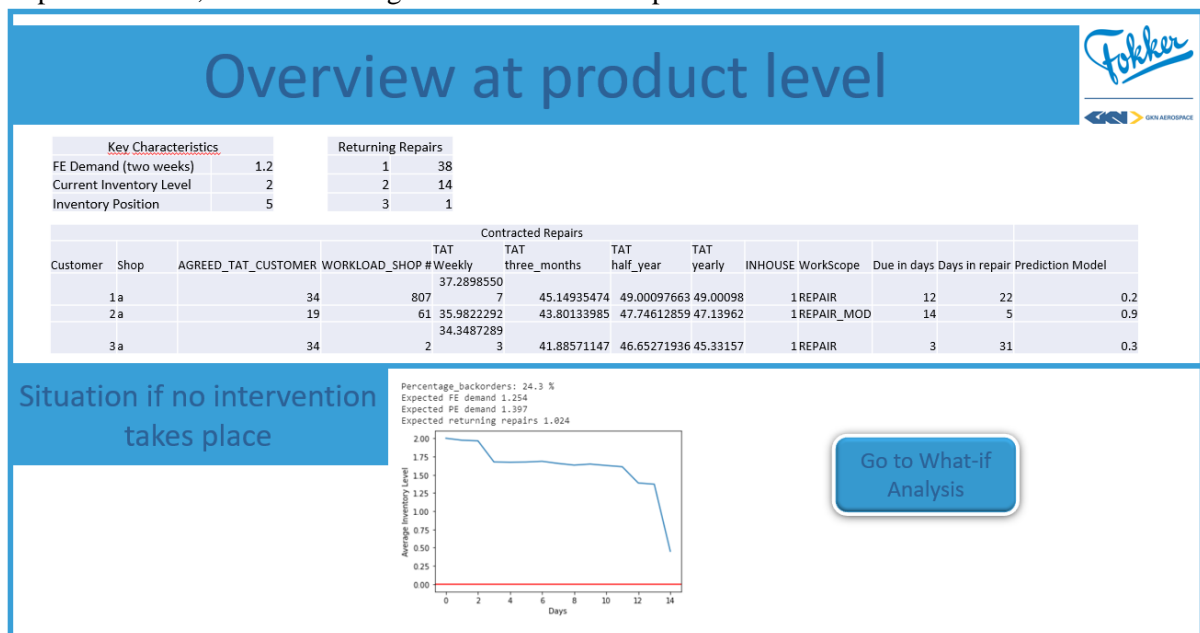


Figure 40 UI overview at product level

contracted repairs are currently open, how many parts are returning and how long they have been in the repair shop, the current inventory level, the expected FE demand for two weeks, and the total inventory position for this part. Furthermore, the results of the simulation are provided, to show how the inventory level changes overtime and in how many of the replications backorders occurred. This way the operational planners can get a better idea of where problems lay and get an indication on how these can be resolved. The final assistance in this process comes from the What-if Analysis, which can be reached by pressing the button on the bottom-right side.

6.4.3 What-if analysis interface

The What-if Analysis allows the operational planners to play with possible interventions and see their impact. As the goal of this thesis was to create alerts, and not necessarily how to resolve problems, the what-if analysis is considerably basic. The operational planners can, for example, remove a contracted repair because they contacted the customer, who informed them that the repair has to priority and no performance exchange is required. After they remove the contracted repair, the simulation is executed again, and the operational planners can view the impact of their intervention. For this specific intervention, the results are shown in Figure 41. Another option for the operational planners is to ask a repair shop to prioritize their order. For this intervention, the results are show in Figure 42. These two

analyses quickly reveal that prioritizing an order has a large reduction in chance of backorders than the option of asking the customer for suspension.

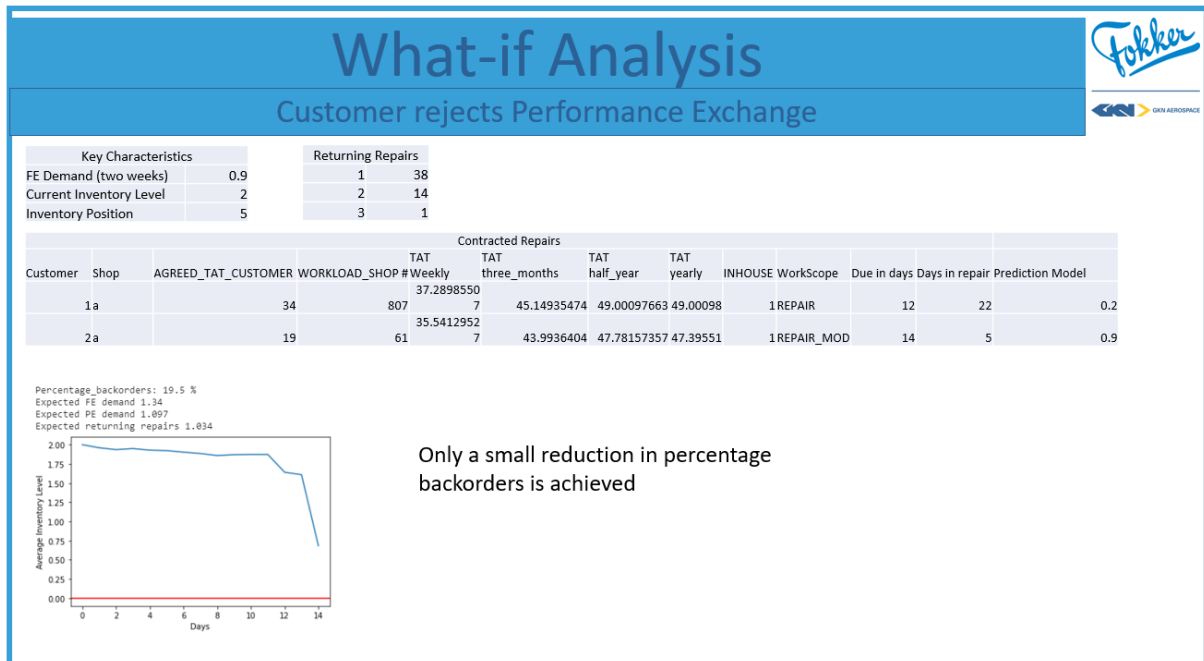


Figure 41 What-if Analysis customer rejects Performance Exchange

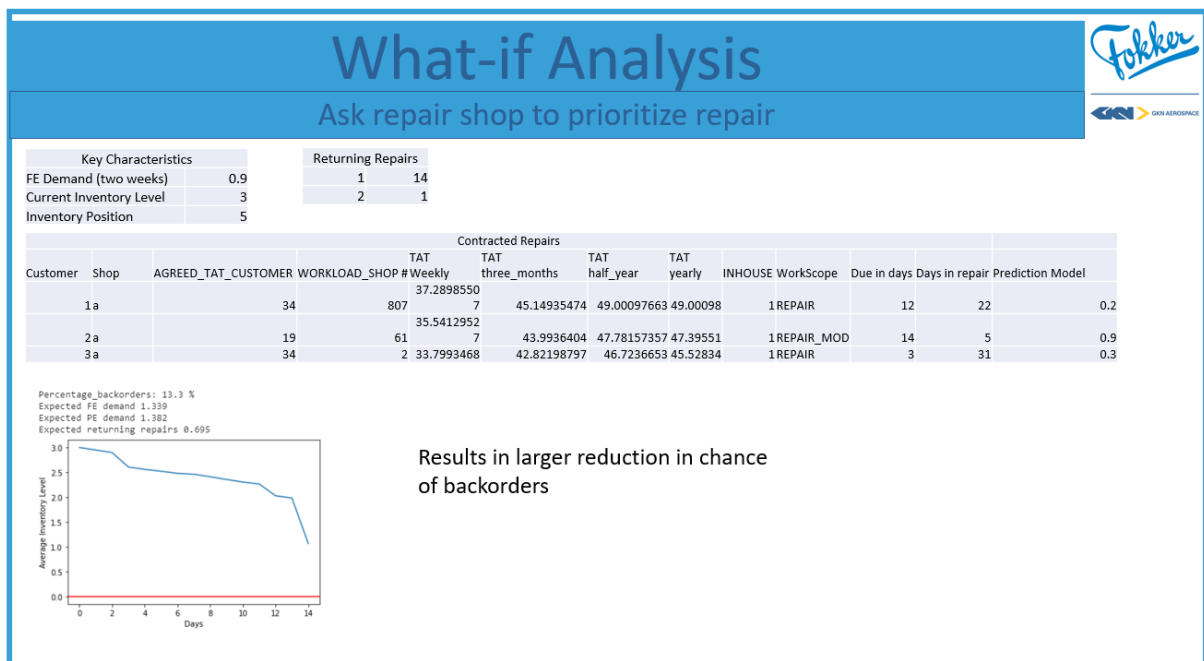


Figure 42 What-if Analysis Ask repair shop to prioritize repair

References

- Arnx, A. (2019). First Neural Network for beginners explained (with code). Retrieved from <https://towardsdatascience.com/first-neural-network-for-beginners-explained-with-code-4cfd37e06eaf>
- Bengio, Y. (2012). Practical recommendations for gradient-based training of deep architectures. In *Neural networks: Tricks of the trade* (pp. 437-478): Springer.
- Bergstra, J., & Bengio, Y. J. T. J. o. M. L. R. (2012). Random search for hyper-parameter optimization. *13*(1), 281-305.
- Bhattacharya, S. (2018). Model Evaluation Techniques for Classification models. Retrieved from <https://towardsdatascience.com/model-evaluation-techniques-for-classification-models-eac30092c38b>
- Bohanec, M., Borštnar, M. K., & Robnik-Šikonja, M. (2016). Integration of machine learning insights into organizational learning: A case of B2B sales forecasting. In *Blurring the boundaries through digital innovation* (pp. 71-85): Springer.
- Brownlee, J. (2014). Classification Accuracy is Not Enough: More Performance Measures You Can Use. *Machine Learning Process*. Retrieved from <https://machinelearningmastery.com/classification-accuracy-is-not-enough-more-performance-measures-you-can-use/>
- Brownlee, J. (2016a). A Gentle Introduction to the Gradient Boosting Algorithm for Machine Learning. *XGBoost*. Retrieved from <https://machinelearningmastery.com/gentle-introduction-gradient-boosting-algorithm-machine-learning/>
- Brownlee, J. (2016b). How to Grid Search Hyperparameters for Deep Learning Models in Python With Keras. Retrieved from <https://machinelearningmastery.com/grid-search-hyperparameters-deep-learning-models-python-keras/>
- Brownlee, J. (2019). Loss and Loss Functions for Training Deep Learning Neural Networks. *Deep Learning Performance*. Retrieved from <https://machinelearningmastery.com/loss-and-loss-functions-for-training-deep-learning-neural-networks/>
- Brownlee, J. (2020). Ordinal and One-Hot Encodings for Categorical Data.
- Cagliano, A. C., Rafele, C. J. G. I. m. S., quality, risks. Operations, & management, t. (2008). Simulation for logistics performance management: comparing different approaches. *9*, 423-442.
- Casas, P. (2019). *Data Science Live Book: Data Science Heroes*.
- Catal, C., Alan, O., & Balkan, K. J. I. S. (2011). Class noise detection based on software metrics and ROC curves. *181*(21), 4867-4877.
- Choi, D., Shallue, C. J., Nado, Z., Lee, J., Maddison, C. J., & Dahl, G. E. J. a. p. a. (2019). On empirical comparisons of optimizers for deep learning.
- Chollet, F. (2018). *Deep Learning mit Python und Keras: Das Praxis-Handbuch vom Entwickler der Keras-Bibliothek*: MITP-Verlags GmbH & Co. KG.
- Clements, J. (2019). Understanding Monte Carlo Simulation. Retrieved from <https://towardsdatascience.com/understanding-monte-carlo-simulation-eceb4c9cad4>
- Donges, N. (2018). Data Types in Statistics.
- El-Aal, A., A El-Sharief, M., Ezz El-Deen, A., & Nassr, A. B. J. J. J. o. E. S. (2008). Supply chain management simulation types: a literature review. *36*(3), 675-687.
- El Naqa, I., & Murphy, M. J. (2015). What is machine learning? In *Machine Learning in Radiation Oncology* (pp. 3-11): Springer.

- Fernández-Delgado, M., Cernadas, E., Barro, S., & Amorim, D. J. T. j. o. m. l. r. (2014). Do we need hundreds of classifiers to solve real world classification problems? , 15(1), 3133-3181.
- Fukunaga, K. (2013). *Introduction to statistical pattern recognition*: Elsevier.
- Fumo, D. (2017). Types of Machine Learning Algorithms You Should Know.
- Gamberger, D., Lavrac, N., & Dzeroski, S. J. A. A. I. (2000). Noise detection and elimination in data preprocessing: experiments in medical domains. 14(2), 205-223.
- García, S., Luengo, J., & Herrera, F. (2015). *Data preprocessing in data mining*: Springer.
- Guillemot, Heusèle, C., Korichi, R., Schnebert, S., Petit, M., & Chen, L. (2019). Tuning Neural network hyperparameters through Bayesian optimization and Application to cosmetic formulation data.
- Haldar, M., Abdool, M., Ramanathan, P., Xu, T., Yang, S., Duan, H., . . . Collins, B. M. (2019). *Applying deep learning to Airbnb search*. Paper presented at the Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining.
- Han, J., Pei, J., & Kamber, M. (2011). *Data mining: concepts and techniques*: Elsevier.
- Hill, T., Marquez, L., O'Connor, M., & Remus, W. J. I. j. o. f. (1994). Artificial neural network models for forecasting and decision making. 10(1), 5-15.
- Hwang, J.-N., Lay, S.-R., & Lippman, A. J. I. T. o. S. P. (1994). Nonparametric multivariate density estimation: a comparative study. 42(10), 2795-2810.
- Jaitley, U. (2018). Why Data Normalization is necessary for Machine Learning models.
- Jolliffe, I. T., & Cadima, J. (2016). Principal component analysis: a review and recent developments. 374(2065), 20150202.
- Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., . . . Liu, T.-Y. (2017). *Lightgbm: A highly efficient gradient boosting decision tree*. Paper presented at the Advances in neural information processing systems.
- Kenton, W. (2019). Backorder. Retrieved from <https://www.investopedia.com/terms/b/backorder.asp>
- Kho, J. (2018). Why Random Forest is My Favority Machine Learning Tool.
- Kohavi, R. (1995). *A study of cross-validation and bootstrap for accuracy estimation and model selection*. Paper presented at the Ijcai.
- Leys, C., Ley, C., Klein, O., Bernard, P., & Licata, L. J. J. o. E. S. P. (2013). Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median. 49(4), 764-766.
- Li, H. (2017). What Machine Learning Algorithm Should I Use?
- Maind, S. B., & Wankar, P. (2014). Research paper on basic of artificial neural network. *International Journal on Recent Innovation Trends in Computing Communication*, 2(1), 96-100.
- Marr, B. (2018). What Are Artificial Neural Networks - A Simple Explanation For Absolutely Anyone. Retrieved from <https://www.forbes.com/sites/bernardmarr/2018/09/24/what-are-artificial-neural-networks-a-simple-explanation-for-absolutely-anyone/#7403cd901245>
- Myung, I. J. J. J. o. m. P. (2003). Tutorial on maximum likelihood estimation. 47(1), 90-100.
- Niu, M., Li, Y., Wang, C., & Han, K. J. I. J. o. M. S. (2018). RFAMyloid: a web server for predicting amyloid proteins. 19(7), 2071.
- Press, G. (2016). Cleaning Big Data: Most Time-Consuming, Least Enjoyable Data Science Task, Survey Says.

- Ramberg, J. S., Dudewicz, E. J., Tadikamalla, P. R., & Mykytka, E. F. J. T. (1979). A probability distribution and its uses in fitting data. *21*(2), 201-214.
- Ronoghan, S. (2018). The Mathematics of Decision Trees, Random Forest and Feature Importance in Scikit-learn and Spark. Retrieved from <https://towardsdatascience.com/the-mathematics-of-decision-trees-random-forest-and-feature-importance-in-scikit-learn-and-spark-f2861df67e3#:~:text=Feature%20Importance-.Feature%20importance%20is%20calculated%20as%20the%20decrease%20in%20node%20impurity,the%20total%20number%20of%20samples.>
- Ruder, S. J. a. p. a. (2016). An overview of gradient descent optimization algorithms.
- Santoyo, S. (2017). A Brief Overview of Outlier Detection Techniques. Retrieved from <https://towardsdatascience.com/a-brief-overview-of-outlier-detection-techniques-1e0b2c19e561>
- Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., & De Freitas, N. J. P. o. t. I. (2015). Taking the human out of the loop: A review of Bayesian optimization. *104*(1), 148-175.
- Sharma, A. (2017). Understanding Activation Functions in Neural Networks. Retrieved from <https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0>
- Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. J. T. j. o. m. l. r. (2014). Dropout: a simple way to prevent neural networks from overfitting. *15*(1), 1929-1958.
- Steward, M. (2019). Simple Guide to Hyperparameter Tuning in Neural Networks. Retrieved from <https://towardsdatascience.com/simple-guide-to-hyperparameter-tuning-in-neural-networks-3fe03dad8594>
- Teng, C.-M. (1999). *Correcting Noisy Data*. Paper presented at the ICML.
- Topan, E., Eruguz, A., Ma, W., van der Heijden, M., & Dekker, R. (2019). A review of operational spare parts service logistics in service control towers. *European journal of operational research*.
- Verbaeten, S., & Van Assche, A. (2003). *Ensemble methods for noise elimination in classification problems*. Paper presented at the International Workshop on Multiple classifier systems.
- Vieira, S., Pinaya, W. H., & Mechelli, A. (2017). Using deep learning to investigate the neuroimaging correlates of psychiatric and neurological disorders: Methods and applications. *74*, 58-75.
- Wierenga, R. (2019). An Empirical Comparison of Optimizers for Machine Learning Models. Retrieved from <https://heartbeat.fritz.ai/an-empirical-comparison-of-optimizers-for-machine-learning-models-b86f29957050>
- Yiu, T. (2019). Understanding Random Forest.
- Yufeng, G. (2017). The 7 steps of Machine Learning. Retrieved from <https://towardsdatascience.com/the-7-steps-of-machine-learning-2877d7e5548e>
- Zhu, X., Wu, X., & Chen, Q. (2003). *Eliminating class noise in large datasets*. Paper presented at the Proceedings of the 20th International Conference on Machine Learning (ICML-03).
- Zhu, X., & Wu, X. J. A. i. r. (2004). Class noise vs. attribute noise: A quantitative study. *22*(3), 177-210.
- Zulkifli, H. (2018). Understanding Learning Rates and How It Improves Performance in Deep Learning. Retrieved from <https://towardsdatascience.com/understanding-learning-rates-and-how-it-improves-performance-in-deep-learning-d0d4059c1c10>

APPENDIX

APPENDIX A: Detailed Description of Available Data

Column name	Feature name	Feature Description
CUSTNAME	Customer name	Represents the name of the customer by which the order was placed
PRIORITY	Priority indicator	Represents the priority, as described by the customer. E.g. routine repair, or a AOG situation
TRANS_TYPE	Transaction type (Repair/Exchange)	Indicates the transaction type by which the order is closed. Either repair or exchange
EXCH_TYPE	Exchange type (Performance/Forward)	Indicates kind of Exchange by which the order is closed. Forward, Shop, or repair
PARTNUMBER	Unique part number	Part number of the respective part for which the order is opened.
PRODUCT_ID	Unique product identifier	Internal aggregation of part numbers, based on the availability of two-way interchangeability
KEYWORD	Product description keyword	Keyword which indicates the type of part
COND	Product Condition	Product condition as described by the customer.
LINE_ADDED	Repair shop entry date	Date on which the order line was created
DELIVERED	Delivery date	Date on which the delivery by the IAC took place
DEL_DUE_DATE	Delivery due date	Delivery due date, based on customer contracts, and industry standards
CORE_RCVD	Returning part receive date	Date at which the returning core unit was received by the repair shop.
STOCK_UPDATED	Warehouse stock update	Replenishment date of the inventory pool of the IAC
WORK_PERF	Work performed	The work performed by the repair shop
SHOP	Repair shop number	The name of the repair shop that has executed the repair
PROD_GROUP	Product group keyword	Product group to which the respective part belongs.
MAINT_TAT_AGR	Agreed turnaround time	Contracted turnaround time towards the customer

APPENDIX B: Interview questions operational planners

This interview/survey has been conducted for the operational planners of the Independent Aerospace Company. The goal of this interview is to identify: How alerts are received, for which situations alerts are received, how/which alerts are prioritized, and finally how the correct response to an alert is determined. An alert is defined as a call for action.

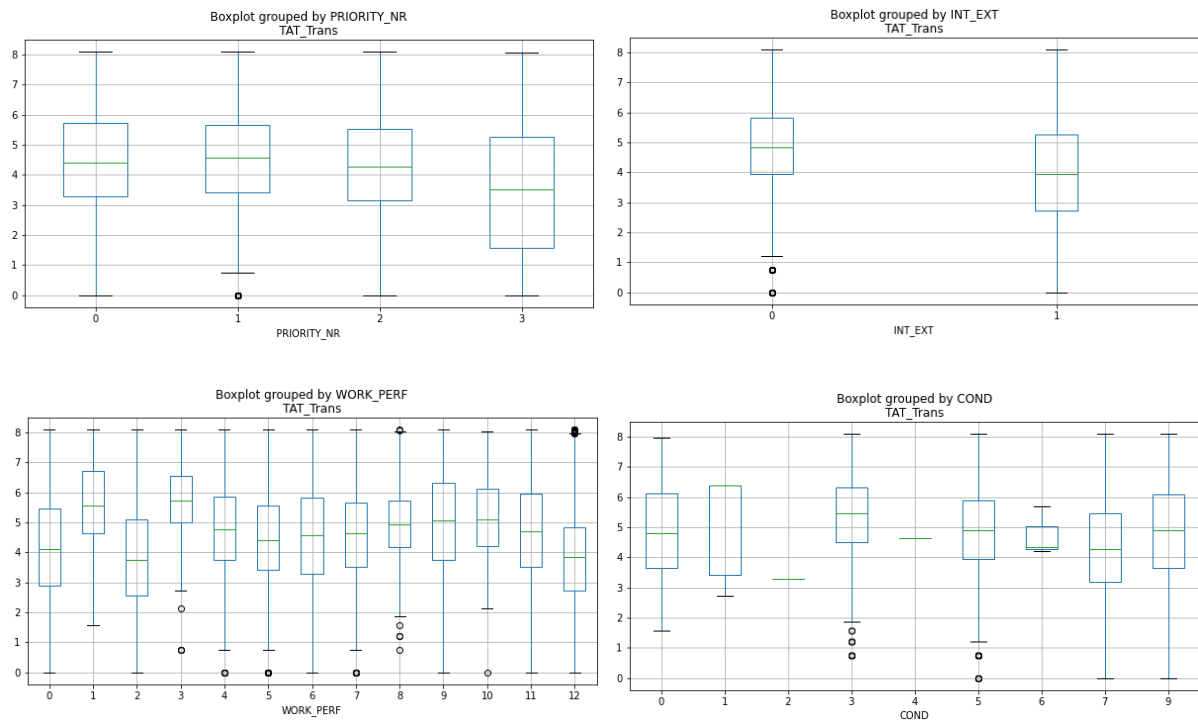
1. By which means do you receive alerts?
 - a. Reports (which)
 - b. Colleagues (both internal and external)
 - c. Other? Please elaborate
2. For which situations do you get alerts?
 - a. Direct backorders
 - b. No stock on hand
 - c. Turnaround time (repair shop)
 - d. Core return time (by customer)
 - e. Age of the order
 - f. Other? Please elaborate
3. For which situations would you like to get alerts?
 - a. Direct backorders
 - b. No stock on hand
 - c. Turnaround time (repair shop)
 - d. Core return time (by customer)
 - e. Age of the order
 - f. Other? Please elaborate
4. Could you distribute 100 points across options below, based on likelihood of occurrence, which occur most often?
 - a. Direct backorders
 - b. No stock on hand
 - c. Turnaround time (repair shop)
 - d. Core return time (by customer)
 - e. Age of the order
 - f. Other
5. Could you distribute 100 points across options below, based on level of impact, which has the highest impact?

Impact is defined as: measures (money, time, resources) required to resolve the disruption

 - a. Direct backorders
 - b. No stock on hand
 - c. Turnaround time (repair shop)
 - d. Core return time (by customer)
 - e. Age of the order
 - f. Other
6. How do you prioritize alerts?
 - a. Impact analysis
 - b. Qualitative methods
 - c. Quantitative methods
 - d. Past experience
 - e. Other? Please elaborate?
7. Could you distribute 100 points across these methods, how alerts are prioritized?
 - a. Impact analysis
 - b. Qualitative methods
 - c. Quantitative methods
 - d. Past experience
 - e. Other
8. When you get an alert, how do you determine your response (action)?

- a. Business rules
 - b. Past experience
 - c. Other?
- 9. Could you distribute 100 points across the following methods, how actions are determined?
 - a. Business rules
 - b. Past experience
 - c. Other
- 10. What information would support you in making better (more educated) decisions?
- 11. Would you trust a Neural Network (form of AI) as main source of alerts?
 - a. Why (not)?
 - b. If not → What would help you in gaining more trust in the model?

APPENDIX C: Box plots of categorical Data



APPENDIX D: Data pre-processing

In this appendix, a detailed overview of the data pre-processing steps is described.

Reducing missing data

In this section, for each column that contains missing data, the approach to deal with these missing values is described. For the columns that are not explicitly mentioned, the columns either contained none, or very few missing values. If this was the case (less than 0.5% missing values), the rows containing these missing values were dropped. The resulting dataset contains 271,822 rows.

Core Received Date

For the Core Received Date, 31% of the values is missing. However, behaviour is expected, as for a common repair, there is no Core Received. Therefore, this field expected to be empty for the repairs in the data set. Furthermore, there are recent exchanges for which the part has already been delivered, but the customer has not send the part back yet (as described in chapter 0, returning the core unit regularly takes more than 100 days). When these cases are excluded, the % missing values is lower than 0.1%, so these cases can be dropped without significantly harming the training set. After dropping the rows, which are missing the Core Received Date, the data set contains 271,364 rows.

Stock Updated Date

Another column with a large number of missing values, is the stock updated date. For this feature, similar results are found as for the core received date. When looking at the cases for which the feature is missing, all cases with stock updated missing are found when the repair for the part is rejected. So, it makes sense that the stock is never updated, as the part is not repaired. However, this behaviour is also occasionally observed for performance exchanges. This is odd, as that means that the customer receives apart from the IAC, as the agreed TAT cannot be met, but the original part is not repaired. However, this behaviour occurs rarely, so it is likely that these cases are just an administrative error, and the customer pays the IAC for the replacement part.

As the actual duration of a repair cannot be determined for parts that were not repaired, all cases with missing stock updated dates are dropped. After dropping the rows with missing values in the stock updated date column, 256,452 rows remain.

Agreed TAT Customer

The final column with many missing values, is the Agreed TAT Customer column. According to the tactical planner at the IAC, this is due to fact that these times are often not explicitly stated for regular repairs. Furthermore, these repairs generally do not fall under a performance exchange contract, so there is no obligation to deliver a performance exchange. Therefore, these repairs will only result in performance exchanges in rare, unpredictable cases. Furthermore, for the forward exchanges, no repair throughput times are contracted, as these are of no use. For the forward exchanges, the IAC has to deliver the part within 24 hours, however, the repair has not agreed throughput time, as it the IACs own part.

If the repairs without contracts and the forward exchanges are excluded, there are still 30,200 missing values for the Agreed TAT Customer column. Thus, dropping these rows, will cause a loss of more than 10 of the remaining data set, and therefore is not desirable. The tactical planner at the IAC said that the industry standard for the Agreed TAT Customer is 34 days. Therefore, instead of dropping all missing values, these are replaced with the value 34 to act in according with the market.

Conclusion

The final number of rows after removing and cleaning missing values is 256,452. This means that only about 5% of the rows has to be removed due to missing data. This is fortunate, as generally more data leads to a stronger machine learning model.

Noise reduction

Non-contracted repairs

A large part of the independent value noise arises from the different types of orders that are in the data set. The data set is a collection of forward exchanges, regular repairs (without contracts), regular repairs (with contracts), and performance exchanges. The repairs that do not have a contract, can generally not become performance exchanges, as there are no contracted TAT agreements with the customer. However, upon investigation, 4000 instances are found where repairs without a contract resulted in a performance exchange. According to the tactical planner at the IAC, this is due to performance exchanges being obligatory for contracted repairs, and optional for non-contracted repairs. This means that often human interaction is incorporated for such exchanges, making them near impossible to predict based on the available data. This will result in a lot noise, as the independent data is unable to explain the dependent data. In turn, this will result in a model trying to learn non-existing relationships. Therefore, only parts that are under contract will be considered in the training data set.

Furthermore, forward exchanges and non-contracted repairs are persistent in the data set. Although these indeed result in demand for the CMA pool, these do not provide any information on performance exchanges. Moreover, these instances are confusing to the model, as way longer turnaround times may be experienced than for performance exchanges. Therefore, these orders should also be dropped from the data set. Dropping all the forward exchanges and the non-contracted repairs, results in a data set containing only 51,180 rows, thus highly reduces the size of the training set.

Work performed

Another source of noise is the work performed attribute. This attribute describes the work that is performed in the repair shop, such as 'test', 'repair', 'repair with modification'. However, the work performed is entered by the repair shop employees, and they are free to enter the work performed how they want. This results in many variations for the same work performed, e.g. 'REP MOD', 'REP PLUS MOD', 'REPMOD' where found in the data set, while each of these obviously mean Repair with modification. Luckily, the total number of

Outlier detection

The second step in data cleaner is Outlier Detection. As described in chapter 3.2.1.2, this will be done using z-scores. However, upon further investigation, the data appears to be non-normal, making the z-score approach inappropriate. To overcome this problem, Leys et al. (2013) propose a method where the median and the mean absolute deviation (MAD) are used. An outlier is identified as a value x_i that lays outside the interval $\frac{x_i - M}{MAD} \leq |\mp 4|$. The deletion and reduction of missing values, and noise reduction, left a dataset of 54,656 rows, which have six columns numerical columns that might contain outliers. After transforming and filtering each of the columns, the final dataset contains 48,854 rows. This means that about 10% of the rows is dropped, due to outlying values.

Data transformation

In this section the transformation of the data is discussed, the first section is on categorical data and the third is on the numerical data.

Categorical data

Ordinal encoding

As discussed in chapter 3.2.2.2, categorical data can be transformed in two ways. Either by ordinal encoding, or by one-hot encoding. Ordinal encoding is only suitable for attributes where an ordinal relationship exists (such as 'bad', 'average', 'good'). This relationship only exists for the priority attribute, which indicates what the priority of the customer is for the repair.

Four categories are found in the data set, these are '-', 'Routine', 'Critical' and 'Aircraft On Ground', the '-' category indicates that the customer has not provided the priority of the order. The numeric values for these categories are 0, 1, 2, 3, respectively.

One Hot encoding

Three attributes require one-hot encoding, which are 'Condition', 'Shop name', and 'Work performed'. For these variables, no ordinal relationship exists, so simply assigning a number to each possible value will give the model confusing information. As discussed in chapter 3.2.2.3, one-hot encoding means that every possible value for the attribute gets its own column, where the column corresponding with the original value gets assigned a '1', and the other columns '0'. This way the model is still able to retrieve the information from the variables, without trying to fit non-existing relationships.

Numerical data

There are six numerical columns remaining in the dataset. A short description, together with some statistics are presented in Table 8. As none of the attributes follow a normal distribution, and the attributes are (mostly) on the same scale, the most suitable transformation is the Min-Max transformation. Where the interval, as provided in Table 8, for each attribute is scaled to [0-1].

Column name	Description	interval	median	mean
AGREED_TAT_CUST	Contracted throughput time for the customer	[12.00, 50.00]	34.00	30.94
NOW_IN_SHOP	Number of items in shop at the moment of entry	[1.00, 1136.00]	59.00	251
WEEKLY	Weekly moving average of TATs for shop of interest	[1.00, 111.35]	36.91	40.61
THREE_MONTHS	Three months moving average of TATs for shop of interest	[7.00, 88.23]	39.14	42.48
HALF_YEAR	Half year moving average of TATs for shop of interest	[7.00, 98.23]	39.95	43.09
YEAR	Yearly moving average of TATs for shop of interest.	[7.00, 95.19]	41.29	44.12

Table 8 Numerical Columns Training data set

Data reduction

Repair shops

As discussed in chapter 2.3.3, the IAC uses many different repair shops. Some of these repair shops are used very regularly, while others are rarely used. The shops which are rarely used provide the model with statistically insignificant data and might therefore cause unnecessary confusion. To overcome this problem, a cut-off percentage has been determined in corporation with the tactical planner of the IAC. It is decided that at least 0.05% of the orders has to be executed by a particular repair shop, in order for this shop to be significant. This means that about 140 orders have to be executed by a shop before it is

considered in the model. Due to this reduction, 40 less columns are in the data set, as the shop names are one hot encoded.

Principle Component Analysis

For the numerical columns, the main approach for reducing dimensionality described in literature is the PCA. However, due to the filtering and cleaning methods previously applied, the dimensionality has become less of an issue. On the other hand, a PCA can also improve accuracy of a model, by removing variables that provide little information to the model. To test if this is indeed the case, a simple random forests classifier with 100 trees is used to evaluate every possible reduction using PCA. The results of this test are found in Figure 43, where on the x-axis is the number of components to which the numerical columns are reduced by the PCA, and the y-column the performance indicators of the test are shown.

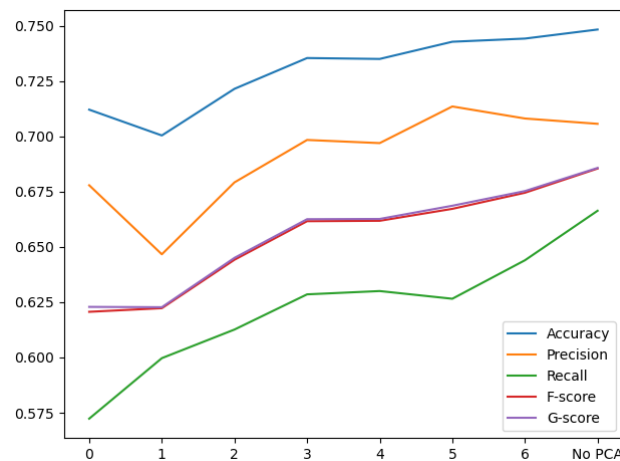


Figure 43 PCA Performance

From the test, it is clear that performing a PCA only reduces the performance, as the best results are found when no PCA is applied. The small performance dip at inclusion of one numerical value in the model is probably caused by the way a PCA works. The PCA tries to explain as much variance in the variables, in only one variable. When eight variables, have to be reduced into one variable, this variable can contain confusing results. As the best performance is achieved when no PCA is applied, all numerical values will be included in the model.

APPENDIX E: Bayesian Optimization

Iteration	Target Value (Accuracy)	Dropout %	Learning Rate	Percentage Neurons	Neuron Shrink
1	0.7204	0.2081	0.07203	0.01011	0.3093
2	0.7433	0.07323	0.009234	0.1944	0.3521
3	0.5712	0.198	0.05388	0.425	0.6884
4	0.5712	0.102	0.08781	0.03711	0.6738
5	0.5712	0.2082	0.05587	0.149	0.2061
6	0.5712	0.3996	0.09683	0.3203	0.6954
7	0.5712	0.4373	0.08946	0.09419	0.04866
8	0.5712	0.08475	0.08781	0.1074	0.4269
9	0.5712	0.478	0.05332	0.695	0.3224
10	0.7192	0.3426	0.08346	0.02811	0.7526
11	0.7431	0.003852	0.01619	0.7123	0.7671
12	0.5712	0.2208	0.04135	0.5523	0.7468
13	0.5712	0.3111	0.02232	0.9595	0.8066
14	0.5712	0.01058	0.08079	0.9652	0.7051
15	0.7403	0.1818	0.00224	0.1609	0.9718
16	0.5712	0.4967	0.03252	0.8187	0.05419
17	0.7292	0.4896	0.02704	0.8107	0.4521
18	0.5712	0.1621	0.06358	0.4065	0.754
19	0.6336	0.4966	0.04256	0.7989	0.4604
20	0.7208	0.2102	0.06413	0.01915	0.3037
21	0.737	0.0594	7.24E+00	0.2101	0.3334
22	0.7345	0.481	0.005133	0.8372	0.4597
23	0.5712	0.1168	0.0447	0.5546	0.9497
24	0.5712	0.471	0.03526	0.8614	0.4273
25	0.7311	0.4854	0.01602	0.8162	0.4551
26	0.5712	0.09784	0.04004	0.5202	0.07939
27	0.4288	0.09178	0	0.2259	0.3497
28	0.5712	0.06672	0.05078	0.8902	0.5715
29	0.7407	0.04848	0.006358	0.21	0.3558
30	0.7227	0.4082	0.01635	0.02488	0.1694
31	0.7405	0.000541	0.000463	0.7055	0.7989
32	0.7413	0.03407	0.007566	0.1873	0.352
33	0.7313	0.3501	0.03842	0.2757	0.4943
34	0.7411	0.05529	0.02207	0.1881	0.3337
35	0.5712	0.4247	0.0783	0.9032	0.1025
36	0.7397	0.04802	0.00495	0.1612	0.3329
37	0.7425	0.1917	0.000623	0.1562	0.9863
38	0.7129	0.2438	0.006419	0.3363	0.6438
39	0.5761	0.01039	0.03677	0.7175	0.7989
40	0.5712	0.3269	0.02178	0.7056	0.9079
41	0.7294	0.2074	0.02553	0.1667	0.9716
42	0.5712	0.3693	0.09839	0.1688	0.7437
43	0.5712	0.1795	0.05069	0.7276	0.5257
44	0.4288	0	0	0.6846	0.7695
45	0.5712	0.4644	0.09125	0.7574	0.6925
46	0.5712	0.05568	0	0.1858	0.3432
47	0.5712	0.1747	0.09429	0.3856	0.2965
48	0.5712	0.2155	0.06788	0.9323	0.6788
49	0.7407	0.3218	0.01252	0.6826	0.2643
50	0.7282	0.2103	0.04181	0.8267	0.3644
51	0.7321	0.4826	0.01519	0.8272	0.4679
52	0.5712	0.4942	0.0644	0.09693	0.07991
53	0.7372	0.04918	0.00295	0.2322	0.342
54	0.7372	0.3439	0.01178	0.5481	0.7025
55	0.5712	0.1897	0.08484	0.4168	0.3791
56	0.7436	0.04069	0.02224	0.1805	0.35
57	0.702	0.4772	0.02406	0.8277	0.4505
58	0.7343	0.03504	0.01915	0.1733	0.331
59	0.7448	0.143	0.01039	0.855	0.1497
60	0.5712	0.3592	0.0287	0.2535	0.184
61	0.7438	0.02673	0.02187	0.2055	0.3508
62	0.7284	0.2798	0.08159	0.7101	0.1682
63	0.7436	0.2427	0.002796	0.3359	0.6542
64	0.5712	0.3594	0.05727	0.6623	0.9914
65	0.738	0.0513	0.007272	0.2157	0.3398
66	0.5712	0.4874	0.05389	0.9846	0.794
67	0.7476	0.04639	0.01939	0.1998	0.373
68	0.5712	0.03959	0.04147	0.195	0.3599
69	0.7444	0.05785	0.01842	0.1743	0.3138

70	0.5712	0.06021	0.03897	0.157	0.3308
71	0.7339	0.3989	0.0272	0.7017	0.4495
72	0.7409	0.00062	0.003938	0.7246	0.782
73	0.5624	0.3529	8.21E+00	0.5275	0.7065
74	0.6973	0.3366	0.02113	0.5646	0.6992
75	0.7413	0.05523	0.01677	0.2295	0.3186
76	0.7401	0.07259	0.001826	0.1926	0.3553
77	0.4288	0	0	0.7252	0.8069
78	0.7429	0.2844	0.003483	0.8401	0.5449
79	0.7325	0.1806	0.02487	0.1584	0.9944
80	0.7016	0.4803	0.02082	0.8088	0.4438
81	0.7444	0.05117	0.01782	0.2053	0.3198
82	0.6969	0.1913	0.01893	0.1522	0.9764
83	0.5712	0.2631	0.08081	0.7921	0.6468
84	0.7358	0.1902	0.01218	0.1743	0.9846
85	0.7335	0.03451	0.09755	0.938	0.285
86	0.5712	0.4294	0.0312	0.6811	0.8946
87	0.7217	0.2081	0.02427	0.1875	0.9789
88	0.5712	0.3015	0.06901	0.6557	0.4209
89	0.746	0.09376	0.009769	0.5817	0.9024
90	0.5712	0.1376	0.03929	0.5725	0.1284
91	0.7495	0.01713	0.008099	0.1757	0.3535
92	0.5712	0.3806	0.06583	0.306	0.6894
93	0.7196	0.2086	0.06449	0.02608	0.3067
94	0.7499	0.06186	0.02349	0.2082	0.3315
95	0.7374	0.02297	0.004337	0.1915	0.3684
96	0.5712	0.119	0.09532	0.9706	0.5436
97	0.5712	0.2706	0.0627	0.503	0.1909
98	0.7333	0.05496	0.04872	0.9084	0.2961
99	0.7444	0.007437	0.01052	0.7258	0.7584
100	0.7444	0.03621	0.01564	0.1835	0.3919
101	0.5712	0.2108	0.07201	0.8414	0.4219
102	0.7472	0.02077	0.003166	0.183	0.3369
103	0.7374	0.01007	0.01705	0.1914	0.3474
104	0.7317	0.02667	0.000116	0.2219	0.3407
105	0.7474	0.05572	0.01239	0.2186	0.3382
106	0.7407	0.1192	0.03061	0.4848	0.1374
107	0.7411	0.05518	0.000193	0.2072	0.3883
108	0.7352	0.1227	0.02027	0.8005	0.8581
109	0.5712	0.2075	0	0.1733	0.9669
110	0.5712	0.3211	0.07991	0.2961	0.4397

APPENDIX F: Hyper parameter tuning of Random Forest

Iteration	n_estimators	Min samples split	Min samples leaf	Max features	Max depth	bootstrap	Mean accuracy
0	400	5	1	sqrt	30	WAAR	0.753525291
1	2000	5	1	sqrt	10	WAAR	0.724663392
2	1200	5	2	sqrt	10	ONWAAR	0.724094796
3	2000	2	4	auto	30	ONWAAR	0.744405022
4	1600	2	4	sqrt	10	WAAR	0.722070597
5	800	5	4	sqrt	30	ONWAAR	0.744405022
6	1000	5	2	sqrt	100	ONWAAR	0.753093159
7	600	5	1	sqrt	60	ONWAAR	0.74242631
8	1000	2	1	auto	50	ONWAAR	0.738719068
9	1800	5	4	auto	10	ONWAAR	0.723276019
10	400	10	4	auto	70	WAAR	0.740311135
11	800	5	1	sqrt	90	ONWAAR	0.742653748
12	2000	10	1	sqrt	10	ONWAAR	0.725504913
13	1600	5	2	sqrt	10	ONWAAR	0.724185771
14	800	10	2	sqrt	30	ONWAAR	0.751387373
15	1800	2	4	auto	50	ONWAAR	0.744996361
16	600	5	2	auto	70	ONWAAR	0.753002183
17	1000	2	1	sqrt	20	WAAR	0.752092431
18	1800	10	2	auto	110	ONWAAR	0.752592795
19	600	5	1	auto	80	WAAR	0.750068231
20	1800	10	1	sqrt	30	ONWAAR	0.752911208
21	1600	5	1	sqrt	70	ONWAAR	0.742812955
22	1400	5	4	sqrt	80	WAAR	0.739765284
23	1800	2	2	auto		WAAR	0.751046215
24	1400	5	1	sqrt	80	ONWAAR	0.742358079
25	400	2	1	sqrt		ONWAAR	0.73849163
26	1400	2	1	auto	40	ONWAAR	0.741016194
27	1000	5	2	sqrt	20	WAAR	0.743927402
28	2000	10	4	auto	100	ONWAAR	0.744655204
29	1200	2	2	sqrt	20	WAAR	0.745337518
30	1200	10	4	sqrt	20	ONWAAR	0.739628821
31	800	2	2	sqrt	50	ONWAAR	0.752638282
32	800	5	1	sqrt	100	ONWAAR	0.742016921
33	800	10	4	sqrt	50	WAAR	0.739333151
34	1800	2	4	sqrt	90	WAAR	0.739697052
35	800	10	2	sqrt	20	ONWAAR	0.74540575
36	1200	5	2	sqrt	20	WAAR	0.74433679
37	800	2	1	auto	100	WAAR	0.743358806
38	800	5	2	auto		WAAR	0.751319141
39	1000	2	2	sqrt	60	WAAR	0.751933224
40	200	5	4	auto	10	WAAR	0.722275291
41	600	10	2	sqrt	60	WAAR	0.749954512
42	800	2	4	sqrt	90	WAAR	0.739947234
43	400	10	4	sqrt	90	WAAR	0.740129185
44	200	5	2	auto	90	ONWAAR	0.753661754
45	1000	2	1	sqrt	110	WAAR	0.743245087
46	2000	2	2	auto	90	WAAR	0.75250182
47	400	10	4	sqrt	80	ONWAAR	0.743995633
48	1200	2	4	sqrt	70	ONWAAR	0.74540575
49	600	2	2	sqrt	110	ONWAAR	0.753411572
50	1800	2	1	auto	20	ONWAAR	0.751819505
51	2000	10	2	auto	50	ONWAAR	0.752888464
52	1000	10	4	auto	50	ONWAAR	0.744473253
53	1000	5	4	auto	30	ONWAAR	0.743904658
54	1400	5	2	sqrt		ONWAAR	0.75316139
55	600	2	4	sqrt	60	ONWAAR	0.744700691
56	1600	5	1	auto	10	WAAR	0.724617904
57	1800	2	2	auto	80	WAAR	0.751887737
58	1400	2	1	auto	100	WAAR	0.743472525

59	1400	10	2	sqrt	80	WAAR	0.75077329
60	200	2	1	sqrt	50	WAAR	0.743586245
61	400	2	4	sqrt	10	WAAR	0.72143377
62	1000	10	4	auto	80	ONWAAR	0.744450509
63	1200	10	2	auto		ONWAAR	0.752729258
64	1600	10	1	sqrt	20	WAAR	0.750068231
65	1600	10	2	auto		WAAR	0.750090975
66	1800	2	4	auto	10	ONWAAR	0.723185044
67	1400	2	2	auto	70	WAAR	0.751296397
68	1000	10	1	sqrt	80	ONWAAR	0.748726346
69	2000	10	2	auto	60	ONWAAR	0.753024927
70	1400	2	4	sqrt	80	ONWAAR	0.744791667
71	800	2	4	sqrt	20	ONWAAR	0.738991994
72	1800	5	2	sqrt	60	WAAR	0.751592067
73	400	5	1	auto	90	ONWAAR	0.741584789
74	1600	5	1	auto	90	WAAR	0.75029567
75	400	10	2	sqrt	90	ONWAAR	0.753229622
76	1600	10	1	sqrt		WAAR	0.755276565
77	2000	5	1	sqrt	100	WAAR	0.750022744
78	1000	5	2	sqrt	10	WAAR	0.724003821
79	200	5	4	auto	80	WAAR	0.739674309
80	2000	2	4	auto	60	ONWAAR	0.745087336
81	600	10	2	auto	100	WAAR	0.750363901
82	1000	5	4	auto	100	WAAR	0.740925218
83	800	10	2	auto		ONWAAR	0.752456332
84	800	5	4	sqrt	70	ONWAAR	0.744200328
85	600	10	1	sqrt	40	ONWAAR	0.750977984
86	400	5	1	sqrt	100	WAAR	0.749636099
87	1600	2	4	sqrt	80	WAAR	0.740333879
88	800	10	4	sqrt	100	WAAR	0.739628821
89	1000	10	2	sqrt	10	WAAR	0.723548945
90	600	10	1	sqrt	110	WAAR	0.755117358
91	2000	5	4	sqrt		ONWAAR	0.744768923
92	800	5	1	sqrt	40	WAAR	0.752228894
93	600	2	4	auto	40	WAAR	0.740424854
94	600	2	4	sqrt	30	ONWAAR	0.744723435
95	400	10	2	auto	40	ONWAAR	0.752797489
96	1000	2	1	auto	10	ONWAAR	0.725618632
97	200	5	4	auto	100	ONWAAR	0.744996361
98	2000	2	2	sqrt	20	WAAR	0.745360262
99	2000	10	2	auto	40	WAAR	0.749886281

APPENDIX G: Distribution Fitting

Skewness: 2.169197

Kurtosis: 5.237512

Fitted cauchy distribution with error=0.001409437517092033)

Fitted chi2 distribution with error=0.0009554324149101949)

Fitted expon distribution with error=0.0011837759389542683)

Fitted exponpow distribution with error=0.0018329320039017804)

Fitted gamma distribution with error=0.0009597227273189872)

Fitted lognorm distribution with error=0.0008109146635623592)

Fitted norm distribution with error=0.002769263358963135)

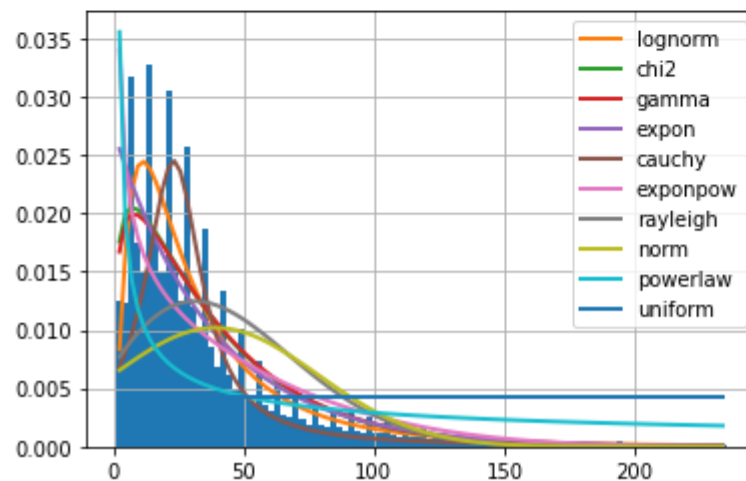
Fitted powerlaw distribution with error=0.0032316443868206654)

Fitted rayleigh distribution with error=0.0022160225255246683)

Fitted uniform distribution with error=0.004945923870322529)

	sumsquare_error	aic	bic	kl_div
lognorm	0.000811	1348.483872	-1.800149e+06	0.064536
chi2	0.000955	1376.564329	-1.784275e+06	0.086057
gamma	0.000960	1371.292651	-1.783841e+06	0.086947
expon	0.001184	1347.074005	-1.763543e+06	0.094398
cauchy	0.001409	1440.335892	-1.746654e+06	0.127503
exponpow	0.001833	1347.688701	-1.721211e+06	0.152732
rayleigh	0.002216	1578.003874	-1.702851e+06	0.256701
norm	0.002769	1625.026871	-1.681279e+06	0.325412
powerlaw	0.003232	1159.545187	-1.666321e+06	0.467987
uniform	0.004946	1095.064223	-1.625139e+06	1.104483

(0.8967516441773755, -1.0975617043188368, 27.19767814236445)



APPENDIX H: Drawing from conditional probability

If in the future runtimes do cause problems, an alternative way of generating random variates is by using the following property:

Let U be distributed $Uniform(0,1)$ and let $X = F^{-1}(U)$. Then X has CDF F .

So, to generate random variates of X , we can feed random uniform variates to the inverse Cumulative Density Function of X (Clements, 2019). However, these variates are currently unbounded (e.g. they can take values on the entire interval $[0, \infty)$). To generate random variates that are subject to the minimal duration condition, the lower bound of the Uniform distribution is determined by feeding the minimal duration to the CDF of X . Now every drawn random variate meets the condition, and only one random number has to be drawn for each returning repair. However, as the runtimes of the first approach are reasonable, and the implementation much easier, this approach will be used.