Rafael Vendramini Savi

USE OF DIRTY AND INCOMPLETE CLAIM DATA ON THE INFERENCE OF PRODUCT RELIABILITY WITH STATISTICAL AND NEURAL NETWORK MODELS

Dissertação submetida ao Programa de Pós-Graduação em Engenharia de Automação e Sistemas da Universidade Federal de Santa Catarina para a obtenção do Grau de Mestre em Engenharia de Automação e Sistemas
Orientador: Prof. Dr. Marcelo Ricardo Stemmer

Florianópolis
2019

Ficha de identificação da obra elaborada pelo autor através do
Programa de Geração Automática da Biblioteca Universitária da UFSC.

Savi, Rafael Vendramini
Use of dirty and incomplete claim data on the inference of product reliability with statistical and neural network models / Rafael Vendramini Savi ; orientador, Marcelo Ricardo Stemmer, 2019.
134 p.

Dissertação (mestrado) - Universidade Federal de Santa Catarina, Centro Tecnológico, Programa de Pós Graduação em Engenharia de Automação e Sistemas, Florianópolis, 2019.

Inclui referências.

1. Engenharia de Automação e Sistemas. 2. Análise de confiabilidade. 3. Weibull. 4. Classificação de Texto. 5. Redes Neurais Artificiais. I. Ricardo Stemmer, Marcelo. II. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Engenharia de Automação e Sistemas. III. Título.

Rafael Vendramini Savi

USE OF DIRTY AND INCOMPLETE CLAIM DATA ON THE INFERENCE OF PRODUCT RELIABILITY WITH STATISTICAL AND NEURAL NETWORK MODELS

Esta Dissertação foi julgada adequada para obtenção do Título de Mestre em Engenharia de Automação e Sistemas e aprovada em sua forma final pelo Pós-Graduação em Engenharia de Automação e Sistemas da Universidade Federal de Santa Catarina

Florianópolis, 19 de maio de 2019.

Prof. Dr. Werner Kraus Junior
Coordenador do Curso

Banca Examinadora:

Prof. Dr. Marcelo Ricardo Stemmer
Orientador
Universidade Federal de Santa Catarina

Prof. Dr. Rodrigo Bastos Fernandes
Universidade Federal de Santa Catarina

Prof. Dr. Ubirajara Franco Moreno
Universidade Federal de Santa Catarina

Prof. Dr. Jomi Fred Hübner
Universidade Federal de Santa Catarina

DISCLAIMER

The following master's thesis contains confidential data of Bosch Rexroth AG. Publications and reproductions of the Master Thesis – even in excerpts – are not permitted without the express permission of Bosch Rexroth AG. The master's thesis is only available to the proofreaders and the members of the examination board.

AGRADECIMENTOS

The realization of this work relied not only on me, but also on the support of various individuals and organizations.

I could not be where I am today without my parents Valmir Anastácio Savi and Marilene Vendramini Savi, who together have always granted me education and care.

My friends and colleagues Amadeu Plácido Neto and Maria Eduarda Bastos Lübcke were always present in the moments I mostly needed. Moreover, I have always been able to count on my friends Isis Machado Silva, Ivana Japjec and Alexandre Orth, who even far away, have always encouraged me to go ahead and give my best.

I would like to thank my mentor and professor, Dr. Marcelo Ricardo Stemmer, for trusting me thousands of kilometers away and believing in my capacity of realizing a valuable work.

The researches performed in this master work could only be possible with the immense amount of data provided by the company Bosch Rexroth AG. However, I certainly would not be able to achieve the same results without the technical support of my supervisors Danny Köllmann and Jens Göbel. Their previous experiences and industry expertise were essential for the understanding of the problem and the respective solutions.

Finally, I would like to express my appreciation for the financial support contributed by CAPES during my studies in Brazil.

“If you’re a scientist, and you have to have an answer, even in the absence of data, you’re not going to be a good scientist.” – Neil deGrasse Tyson, Astrophysicist

RESUMO

Este trabalho se preocupa em apresentar e utilizar técnicas para tratamento de dados de reclamações de campo extremamente deficientes, a fim de estimar modelos de confiabilidade das famílias de produtos mais importantes do portfólio elétrico da empresa Bosch Rexroth. Esta é uma questão relevante para empresas que desejam tomar decisões sobre estratégias de mercado e desenvolvimento de produtos com base em seus comportamentos em campo. Todo o processo de limpeza e classificação dos dados até a escolha do modelo de confiabilidade apropriado é guiado por um estudo detalhado do estado da arte. Uma restrição presente neste trabalho é que a quantidade extremamente grande de dados armazenados nos últimos vinte anos deve ser processada automaticamente com a menor iteração humana possível. Isso motivou a pesquisa de resoluções alternativas envolvendo métodos de aprendizado de máquina que normalmente não são adotados no campo de confiabilidade abordado. Técnicas para processamento de texto natural encontradas no ramo da análise de sentimento foram adaptadas para a classificação de reclamações com base em comentários escritos durante os reparos. Para tal, um classificador baseado em um modelo de regressão logística foi treinado e obteve uma precisão de 86,2% ao analisar *features* extraídas dos textos. Ao comparar com a classificação desconsiderando a análise textual, houveram 10,9% de correções. Visto que os bancos de dados são incompletos devido a falhas que não são relatadas à empresa, métodos de ajuste adequados para dados contendo suspensões são empregados nos estimadores de *median rank* antes de gerar funções de confiabilidade empíricas. Os mapeamentos destas funções são usados para gerar modelos paramétricos e não-paramétricos. A primeira abordagem os utiliza para encontrar os parâmetros de uma distribuição Weibull que mais os aproxima. Um ajuste fino dos hiperparâmetros da topologia mais adequada de uma rede neural artificial *feed-forward* é demonstrado antes de ela ser usada para modelar os mesmos dados. Os desempenhos de ambas as abordagens de modelagem são comparados utilizando todos os dados de campo disponíveis pela empresa. Os resultados mostram que, enquanto a regressão da distribuição de Weibull é realizada milhares de vezes mais rapidamente que o treinamento das redes neurais, as últimas alcançaram até 212 vezes menor erro de predição.

Palavras-chave: Análise de Confiabilidade. Weibull. Dados de campo. Aprendizado de Máquina. Classificação de texto. Redes Neurais Artificiais.

ABSTRACT

This work is concerned about presenting and making use of techniques to treat very deficient datasets of field claims in order to estimate reliability models of the most important product families of Bosch Rexroth's electrical portfolio. This a relevant issue to corporates that wish to take decisions regarding product development and market strategies based on the behavior of their products in the field. The entire process from cleansing and classifying the data until the choice of appropriate reliability model is guided by a detailed study of the state-of-the-art techniques. A constraint present in this work is that the extremely large amount of data stored in the past twenty years should be processed automatically with the least possible human iteration. This motivated the research for alternative resolutions involving machine learning methods that are not usually adopted in the field of reliability assessment based on claim data. State-of-the-art techniques for natural text processing found in the branch of sentiment analysis were adapted to the classification of claim events based on text remarks of the repairs realized. For this purpose, a soft classifier based on a logistic regression model was trained and obtained an accuracy of 86.2% when analyzing features extracted from the text fields. When comparing to the classification of the claims disregarding the text analysis, it suggested corrections to 10.9% of the labels. Since the prepared databases are incomplete due to failures that are not reported to the company, adjustment methods that are suitable for data containing suspensions are employed to the median rank estimators before generating empirical reliability functions. These mappings are used to evaluate both parametric and nonparametric models. The first approach fits the two-dimensional data to a Weibull distribution, which is considered one of the most important statistical distributions for modeling life data. A fine tuning of hyperparameters of a feed-forward artificial neural network is demonstrated before being used to fit the same data. The performance of both modeling approaches are compared utilizing all the data available in this work. The results show that, while the regression to the Weibull distribution is realized several thousand times faster than training the artificial neural networks, the latter achieved up to 212 times smaller prediction error.

Keywords: Reliability Assessment. Weibull. Field Data. Machine Learning. Text Classification. Artificial Neural Networks.

LIST OF FIGURES

Figure 1: Timeline structure of a field claim pipeline 36

Figure 2: Censoring mechanisms 39

Figure 3: Plot of the sigmoid function 43

Figure 4: Multilayer perceptron with architecture 1-3-3-1 45

Figure 5: Plot of probability density function based on a Weibull distribution 55

Figure 6: Example of a Weibull probability plot 59

Figure 7: Censoring mechanisms in field data 67

Figure 8: Quantity of delivered products by year and family 68

Figure 9: Histogram of quantities in repair entries 70

Figure 10: Method to solving duplicated claims 73

Figure 11: Flowchart of claim classification through available fields 75

Figure 12: Structure of the failure classifier 76

Figure 13: Feature extraction process 79

Figure 14: Interface of the logistic regression model 79

Figure 15: Histogram of count of characters in text fields 80

Figure 16: Histogram of weights across features 82

Figure 17: Quantity of items sold by contract customer class 84

Figure 18: Bathtub curve for the hazard rate over time 88

Figure 19: Heatmap of all event tables 89

Figure 20: Matching deliveries and claims to find unclaimed deliveries 90

Figure 21: Histogram of last event times in datasets 94

Figure 22: Histogram of range of event times in datasets 95

Figure 23: Histogram of smallest median rank values in datasets 95

Figure 24: Histogram of range of median rank values in datasets 96

Figure 25: Plot of Leaky ReLU and ELU 97

Figure 26: Algorithm for grid search of optimum hyperparameters of ANN 98

Figure 27: Heatmap of selected event tables 98

Figure 28: Lengths of median rank tables 100

Figure 29: Boxplot of training times in the grid search for hyperparameter ____ 100
Figure 30: Boxplot of prediction MSE in the grid search for hyperparameter ____ 101
Figure 31: Plot of performance of individual hyperparameter configurations ____ 102
Figure 32: Training time versus number and size of hidden layers _ 103
Figure 33: Prediction MSE versus number and size of hidden layers 103
Figure 34: Training time versus learning rate and batch size ____ 104
Figure 35: Prediction MSE versus learning rate and batch size ____ 104
Figure 36: Training time versus input scaling and batch size____ 105
Figure 37: Prediction MSE versus input scaling and batch size ____ 105
Figure 38: Data flow ____ 108
Figure 39: Software GUI - Tab 'Configuration' ____ 109
Figure 40: Software GUI - Tab 'Calculation' ____ 109
Figure 41: Software GUI - Tab 'Failure Text Classifier'____ 110
Figure 42: Spectrum of time to fit and MSE observations ____ 112
Figure 43: Correction of installed base____ 133
Figure 44: Creation of *event table*____ 134

LIST OF TABLES

Table 1: Relation between reliability functions 49
Table 2: Deviation of approximations of the median rank [89] 51
Table 3: Demonstration of the rank adjustment method 53
Table 4: Reliability functions of the three-parameter Weibull distribution 56
Table 5: Distribution and meaning of V_CODE 71
Table 6: Table for threshold confidences 76
Table 7: Regular expressions based on the Perl syntax 78
Table 8: Confusion matrix of model's prediction 81
Table 9: Most significant features 82
Table 10: Performance comparison by the number of features 83
Table 11: Confusion matrix of model's prediction after limiting features 83
Table 12: Example of an event table 89
Table 13: Collection of fixed and configurable hyperparameters 99
Table 14: Performance comparison for input normalization (median values) 106
Table 15: Collection of chosen hyperparameters 106
Table 16: Benchmark of the failure classifier 111
Table 17: Percentage of datasets in each cluster 113
Table 18: Fitting time (s) of the Weibull model (cluster median values) 113
Table 19: Fitting time (s) of the ANN model (cluster median values) 113
Table 20: Prediction MSE of the Weibull model (cluster median values) 114
Table 21: Prediction MSE of the ANN model (cluster median values) 114
Table 22: Time improvement of the Weibull over the ANN model (cluster median values) 114
Table 23: MSE improvement of the ANN over the Weibull model (cluster median values) 115
Table 24: Python libraries utilized 131

ABBREVIATIONS AND ACRONYMS

Adam	Adaptive Moment Estimation
AI	Artificial Intelligence
ANN	Artificial Neural Network
BGD	Batch Gradient Descent
CDF	Cumulative Distribution Function
CV	Coefficient of Variation
DNN	Deep Neural Network
ELU	Exponential Linear Unit
EM	Expectation–Maximization
FFNN	Feedforward Neural Network
GUI	Graphical User Interface
KM	Kaplan-Meier
LReLU	Leaky Rectified Linear Unit
LS	Least Squares
MBGD	Mini-batch Gradient Descent
ML	Machine Learning
MLE	Maximum Likelihood Estimation
MLP	Multilayer Perceptron
MR	Median Rank
MRR	Median Rank Regression
MSE	Mean Squared Error
MTTF	Mean Time To Failure
NFBR	Non-Failed But Reported
NFF	No Fault Found
NLP	Natural Language Processing
OEM	Original Equipment Manufacturer
PDF	Probability Density Function
RBA	Reduced Bias Adjustment
ReLU	Rectified Linear Unit
RR	Rank Regression
RRX	Rank Regression on X
RRY	Rank Regression on Y
RUL	Remaining Useful Life
SGD	Stochastic Gradient Descent
TF-IDF	Term Frequency-Inverse Document Frequency
TTF	Time-To-Failure

SYMBOLS AND NOTATIONS

Reliability assessment

t	Time (variable)
$P\{x\}$	Probability function
$L(x)$	Likelihood function
$F(t)$	Cumulative distribution function
$f(t)$	Probability density function
$S(t)$	Survival function
$h(t)$	Hazard function
N	Sample size (number of units placed on test)
N_{susp}	Total number of suspensions
N_{eos}	Number of suspensions due to *end-of-study*
N_{fup}	Number of suspensions due to *loss-to-follow-up*
N_{dfm}	Number of suspensions due to *different-failure-mode*
T	Random failure time
t_{c_j}	Censoring time of unit *j*
t_w	Warranty time
t_j	Time when event j occurred
e_j	Event type associated to each unit (1 or F for failure and 0 or S for suspended)
$r(t)$	Number of failed items at time t
$s(t)$	Number of units survived until t, or the number of units at risk at t, including the instant t.

Rank regression and statistical distributions

β	Shape parameter of Weibull distribution
η	Scale parameter of Weibull distribution
γ	Location parameter of Weibull distribution
i	Corrected order number assigned to a failed unit. Sometimes equivalent to j, like in the special case of tests without suspensions (complete data)
i_{-}	Corrected order number (rank) that precedes i
Δi	Order number (rank) increment

j	Position assigned to each unit of sample according to its event time ranked in chronological order and starting at 1
θ	Vector of input parameters of any theoretical distribution
r	Correlation coefficient

Machine learning

$\boldsymbol{x}$	Input vector
$\boldsymbol{y}$	Output vector (prediction)
n_i^l	Value of node i of layer l
$w_{i,j}^l$	Weight of link between node number i of layer l and j of layer $l+1$
b_i^l	Bias applied to node i of layer l
k^l	Number of nodes in layer l
$\tanh(x)$	Hyperbolic tangent function
η	Learning rate
d	Depth, or length of input vector $\boldsymbol{x}$

SUMMARY

1 INTRODUCTION 25

1.1 MOTIVATION 25

1.2 INTRODUCTION TO RELIABILITY MODELS 25

1.3 PROBLEM DESCRIPTION 27

1.3.1 Introduction to the problem 27

1.3.2 Objective 28

1.4 METHODOLOGY 29

1.5 DISSERTATION STRUCTURE 30

2 LITERATURE REVIEW 33

2.1 RELIABILITY ANALYSIS WITH FIELD DATA 33

2.1.1 Motivation 33

2.1.2 Obstacles 33

2.2 INTRODUCTION TO MACHINE LEARNING 40

2.2.1 Introduction to notations and process pipelines 40

2.2.2 Logistic regression for soft classification of text 41

2.2.3 Artificial neural networks 44

2.3 ESTIMATION OF RELIABILITY MODEL 49

2.3.1 Empirical reliability function 50

2.3.2 Parametric modelling with statistical distributions 53

2.3.3 Non-parametric modelling 62

3 DATA PREPROCESSING 65

3.1 APPROACHES TO THE FIELD DATA 65

3.2 DATA SELECTION AND CLEANSING 68

3.3 CLASSIFICATION OF EVENT TYPE 74

3.3.1 Architecture of the failure classifier 74

3.3.2 Feature extraction 77

3.3.3 Training data for the logistic regression model 79

3.3.4 Performance of the logistic regression model 81
3.4 CORRECTION OF INSTALLED BASE 84
4 RELIABILITY MODELLING... 87
4.1 PREPARATION OF DATA.. 87
4.1.1 Assuring homogeneity 87
4.1.2 Creation of the event table 89
4.1.3 Median Rank calculation 91
4.2 ESTIMATION OF RELIABILITY MODELS..................... 93
4.2.1 Parametric distribution... 93
4.2.2 ANN for modeling empirical reliability functions................ 94
5 IMPLEMENTATION AND RESULTS.......................... 107
5.1 IMPLEMENTATION... 107
5.2 CLASSIFICATION OF EVENT TYPE 111
5.3 ESTIMATION OF RELIABILITY MODELS.................. 112
6 CONCLUSION 117
6.1 FURTHER IMPROVEMENTS AND RESEARCHES...... 118
6.1.1 Classification of incomplete observations 118
6.1.2 Parametric modelling... 119
6.1.3 Measurement of error... 119

1 INTRODUCTION

1.1 MOTIVATION

Increasing advances in technology and development of highly sophisticated products as well as higher customer expectations have put new pressures on manufacture processes. Today’s world is a competitive marketplace where companies are more engaged in developing optimized products in every aspect. Competitors rush to deliver new solutions and fight for a place in the international environment. In this context, while overestimated product performance is not as affordable as in the past, there is still a high degree of interest in improving the quality and the overall capabilities [1]. The increasing complexity and diversification of products has led to more difficult design and evaluation of products performances, but also gave place to new researches and refined methods. Reliability is a measure of performance and is described by [2] as the probability that the item will perform its intended function under certain conditions throughout a specified period of service time. There are many reasons to adopt this measure: assessing the effect of a design change, determining if regulations and customer requirements have been met, detecting potential failure modes to correct in future versions, guiding the estimation of warranty costs, checking veracity of advertising claim, etc. Due to the economic importance of repair services, reliability assessment is essential for the classification of applications and customers that under- or overuse the products. Especially in the context of this work, it is used for the identification of scenarios where repairs are claimed less than expected.

1.2 INTRODUCTION TO RELIABILITY MODELS

The reliability assessment can be done through a variety of means and result in models with different characteristics. These are called in the literature interchangeably as *reliability model*, *lifetime data model* or *time-to-failure (TTF) model* [3]. Despite the conception differences, a common feature shared between most of the forms is the *prognostics* functionality, as defined by [4] to be the prediction of the probable remaining useful life (RUL), that is, when a failure may occur. While RUL is the remaining time from a given time until failure, *time-to-failure* (TTF) is a similar measure that is defined as the operating time accumulated from the first use until failure [5]. According to the same standard, *failure* is an event defined by the loss of ability to perform as required.

The many types of lifetime models can be grouped in different ways, as extensively reviewed by [6], and the structure proposed by these authors specifically designed for RUL prediction will be used here as baseline. There are four main groups (which can be refined into many other subgroups) as follows:

I. **Knowledge-based models**: RUL estimation is realized through a series of behavior rules established by an expert in the field that must predict every input and estimate an output. Therefore, it requires a well understood problem where the potential failures are simply defined. Normally, small accuracy is achieved.

II. **Physical models**: A mathematical model representing the physical behavior of processes, such as degradation, describes the relationships between the measured inputs and the derived effects. They receive this name thanks to its affinity to the physical and material sciences [7], but they can also be called white-box models [8]. A very accurate model is achieved, but only if the operating conditions are monitored or statistically represented.

III. **Stochastic models**: It assumes that the times to failure of identical items can be considered statically identical and independent random variables. Therefore, if there are significant and representative samples of a small set of dominant failure modes, it is possible to generate a probabilistic model. Since its stochastic nature do not require real-time condition monitoring, it is not useful to track a specific asset, but for overall maintenance management. In contrast to other models, these can provide various estimates of reliability and uncertainty [9].

IV. **Data-driven models**: Usually approached using machine learning techniques, it treats the system as a black-box (or gray-box in hybrid solutions) in order to model its health-related inputs into reliability indicators. It usually requires huge amount of modeling data and real-time monitoring in order to forecast the status of a specific machine, and is especially useful when physical or stochastic models are impractical or not applicable.

The only family of models that does not require deep knowledge of the product nor condition monitoring is the stochastic, which is chosen to cover the issues of this work, as justified later in the next sections. The data used to estimate such models is usually originated by experiments during the manufacturing phase, or by field returns [10]. In the problem approached by this work, the data available and utilized is originated from field claims. As described by [11], field data can provide more reliable information concerning the life distribution of equipment in actual use than laboratory test data, despite demanding obstacles that are addressed in the development chapters.

1.3 PROBLEM DESCRIPTION

> "Without big data analytics, companies are blind and deaf, wandering out onto the web like deer on a freeway" – Geoffrey Moore

1.3.1 Introduction to the problem

In the last decades, the world is starting to realize the potential of data. It is changing the way companies do business and the economy environment is evolving towards those that use it properly [12]. Inserted in this context is Bosch Rexroth, one of the worldwide leading specialists in drive and control technology. With more than 30,000 associates all over the globe, the company supplies solutions for electrical and hydraulic machinery for mobile and manufacturing applications [13]. Rexroth products can be found in a wide range of sectors such as industrial applications, factory automation, mobile applications and use of renewable energies. Being a leading company in this field with substantial selling volumes, it is critical that management choices are made based on reliable intelligence.

This work took place at the Bosch Rexroth's leading plant for electrical products, located in Lohr am Main, in southern Germany. It is the largest and most important production plant for electrical portfolio of the company, which consists mostly of electric drives, controllers and motors.

Every sold product is associated with a registered customer, in such way that it is possible to retrieve information regarding the place of shipment and industry branch where it is installed, for example. These are stored in a "delivery database" system. Since most of the products are not yet ready for real-time condition monitoring – which is becoming a trend – the company cannot track their use nor health state until the customer reports a claim. This is the biggest factor that encourage the choice of the use of stochastic models over the others presented in chapter 1.1. When a customer reports a claim, it can hold different meanings, such as repair of a problem caused by poor design or misuse, or a request to remanufacture or to return the product in case of dissatisfaction. Activities performed during repair operations are filled in job sheets, and then submitted to a "claim database" system.

The scope of this work targets the main product families of motors, drives, controllers and power supplies. Their operating power ranges from 400W to 100kW.

1.3.2 Objective

This master work is focused on the acquisition of life data models entirely based on the data available from the delivery and claim databases. This can provide to the business manager several advantages in the market in comparison to the competitors, since it is rare the case where a company is able to accomplish this in such a large scale. The biggest challenges encountered in this kind of problem are the flaws of the data, which requires extensive data preparation and manipulation in order to be processed [14].

There are several possible applications provided by reliability models, and the use case currently mostly expected with the results of this work is the estimation of the *repair share*: The department of service has noted that many customers maintain a strong relationship during the warranty period by sharing fault and operation information and requesting assistance. However, when this period expires, this connection weakens and many customers no longer return their products. Assuming that failures can occur after the warranty also, there is a loss of clientele in the market of repairs of products. This loss is quantified with an index called *repair share* (1.1), which express the ratio between the claims collected by the company and the total occurring potential.

$$repair\ share = \frac{collected\ claims}{field\ failures} \tag{1.1}$$

The repair share can be applicable for any product family, country, or even customer. The coverage level is normally chosen by the business decision makers, as this index is highly meaningful during the allocation of efforts to improve less profitable branches. OEMs with low repair shares could be performing repairs with outsourced services or on their own, which would mean losses of Rexroth's potential business, and maybe even inferior product performance and reliability after the repairs. If these customers could be identified individually, it would be possible to approach them directly and investigate how to fortify the cooperation.

It is worth mentioning that although the service operations can also be lucrative, in such market, it is not desired that the number of possible claims increase, because that would mean that the products became more unreliable. Instead, Bosch Rexroth targets to hold all claims addressed by the costumers that possess its products, improving the repair share toward 100%.

1.4 METHODOLOGY

This work is inserted in the context of reliability assessment [2] and is concerned on presenting and making use of techniques to use a very deficient life dataset in the estimation of reliability models of the most important product families of Bosch Rexroth's electrical portfolio. The result is a software tool capable of scanning essential databases and providing automatically as output the reliability function and estimated failures in a given period for any product, taking into consideration the influencing factors. These are factors that play a role in the determination of the life expectancy, and that this work is concerned to identify.

In the literature, there are many works and studies with the same goal of generating reliability models given failure data. However, as presented in [2] and [15], they already take into account that the data is ready to be used. This work, on the other hand, addresses raw data produced manually by those responsible for repairs and claims around the world. For this reason, plenty of relevant information is incorrect or missing, and the recovery of each field is treated independently based on different strategies. In order to support these operations, techniques involving natural text processing are studied and their purposes presented. Instead of being related to the restoration of fields or failure analysis, many of the related works encountered in the literature are focused on sentiment analysis, which labels texts positively or negatively. This work shows how their adaptation to the restoration of incomplete fields providing useful results.

After the databases are cautiously filtered and their fields restored, they can be further processed in the estimation of reliability models. In this phase, the claim entries are as complete as possible, although not all failures are claimed. Therefore, extrapolation methods such as the interpretation of censored data [16] are analyzed. Most of the applications in this field rely on the approximation of empirical reliability functions (created after the failure data) to a pre-defined statistical distribution, such as the Weibull [17]. However, previous knowledge regarding the nature of failure modes present in the item under study is recommended in order to choose the most appropriate theoretical distribution. Since the tool intended by this work must evaluate the reliability for many different families and types of products with as little expert knowledge as possible, choosing every distribution is undesired. In order to solve this problem in a more flexible way, artificial neural networks are adopted for the approximation of the empirical reliability functions. These networks were adopted by [18] and [19] for similar objectives and shined due to their flexibility in modelling data of any shape. This work concludes with an extensive performance comparison of the modelling of failure data using Weibull distributions and artificial neural networks.

1.5 DISSERTATION STRUCTURE

The first chapter introduces the reader to the problem addressed in this work. It begins by presenting the motivations and benefits behind it, and finishes with a glance of the steps involved in the solving process.

The second chapter is subdivided into three parts. An initial introduction to reliability analysis based on field data is given, and all recurrent obstacles and difficulties are presented. Some of these difficulties are solved in this work using machine learning techniques, which are presented in the following subchapter. Finally, a review on the methods for translating field data into reliability models is evaluated and presented.

The development of the proposed solution is realized chronologically by preprocessing the raw data available, and then using it to fit the reliability models. These two steps are separated into the chapters three and four. The first is responsible for showing the format and characteristics of the databases used in this work, and for proposing strategies of cleansing and treating inconsistencies. The last development chapter addresses the automated generation of empirical reliability functions, and how the modelling using statistical distributions and artificial neural networks can be most appropriately realized. This includes justifying the selection of the statistical distribution used and the best topology and hyperparameters for the neural networks.

Subsequently, the fifth chapter explains how all development is fitted into one software tool, and how it performed. Different metrics are used to show advantages and disadvantages of different modelling approaches.

The last chapter concludes the work by summarizing the realized tasks and the difficulties encountered, as well as highlighting the importance of the results of the approached methods and their influence on the company where the project was developed. Moreover, possible improvements for further developments as well as company procedures are suggested.

2 LITERATURE REVIEW

2.1 RELIABILITY ANALYSIS WITH FIELD DATA

2.1.1 Motivation

In chapter 1, an introduction to general reliability models was presented. This section will approach specifically the use of data originated from field claims to build stochastic models.

Field data can be split on warranty data and after-warranty data. Warranty is a contractual obligation connected to a product, in which the manufacturer is committed to realize repairs free of costs, given that the concerned failure was not caused by misuse. Warranty data analysis is exclusively focused on the claims reported during the warranty period of a product, which is normally more complete, while the other uses data of products that are not covered by warranty. In some applications, like automotive, warranty analysis can be considered a two-dimensional problem, since coverage depends not only on the age of the product, but also on the usage, that can be given in kilometers, for example [20]. The warranty contract covered by the products addressed in this work is only concerned on the age of product, therefore one-dimensional analysis will be used.

Field data can capture actual usage profile and the combined environmental exposure that are difficult to simulate in laboratory [21]. These factors play an important role in product reliability [17]. For example, a motor can wear out in different speeds if it is undersized or oversized to a specific application. Furthermore, electronic components can corrode differently if placed next the sea, or in a dry desert. There are many factors that deprive the laboratory tests from the best approximation of customer usage, and two of the most important are the time consumption prior to launch a product and costs. Nevertheless, as described by [22], the ultimate test of a manufactured product is how well it performs in the field, that is, in the hands of the customer.

2.1.2 Obstacles

> “You can have data without information, but you cannot have information without data.” – Daniel Keys Moran

Unlike laboratory tests, field data analysis deal with uncertainties of an uncontrolled environment. If not treated correctly, these can lead to wrong conclusions at most. This is described by [15] as “dirty” data. Most of the properties of such data are explored by [11] and [23] and are here formally presented. The subsequent chapters will address the solutions for these obstacles.

2.1.2.1 Homogeneity

Products that are installed in different environments may excite distinct failure modes in different speeds, and domain expertise may be necessary to identify these cases. [24] explains how important it is to analyze a homogeneous population, that is, with similar reliability characteristics. [25] show how the correct partition of life data of a population of installed transformers has led to sufficient accuracy in the estimation of reliability. [26] present a few measures to determine homogeneous parts of the population of installed transformers of an energy company, for example manufacturer and date of installation. They justify their choices with domain knowledge of the field.

2.1.2.2 Missing or wrong field entries

The claims are composed not only of a single field for the return date, but by many other supporting fields, like the failure description or the delivery date. When some data is missing, the occurrence may lose partially or entirely its meaning, which can lead to overestimated reliability values. Sometimes, more than one entry in the system represents the same failure, which leads to underestimated indices. Some fields may have commonly accepted values, and outliers or wrong data types may be a signal that the claim might be valueless. These problems are experienced by [27] and [28] in the book “Case Studies in Reliability and Maintenance”. In order to resolve situations where uncertainty is present and interpretation is needed, [14] suggests the use of text mining techniques, since warranty claims can also be expressed in text forms.

2.1.2.3 Invalid claims

For simplicity reasons, or maybe to focus on a different problem, most authors assume that claims are reported due to failures of interest [29] when dealing with field data. However, they do not always represent a failure in a product’s life, since their demand depend essentially on human choices made by customers. Invalid claims are those that do not account as a product failure, and therefore special treatment should be addressed. They can be classified in three categories (non-exclusively):

2.1.2.3.1 No-fault found (NFF)

Also known as non-failed but reported (NFBR), it implies that a failure was reported to have occurred during a product's use, but upon subsequent analysis or testing, the failure was not observable. [30] made a study using real data composed of ten years of reports of two leading electronic manufacturing and concluded that NFF accounted for 44% of the claims. This situation can arise because of many factors: One can list lack of sufficient training on product usage or inaccurate operational manual, which can lead the operator to mistakenly believe that the products are failed. False alarms are a common behavior in avionic environment studied by [31], where the system erroneously reports a failure to the user. Another very recurrent property of electronic systems is the presence of intermittent failures, as explained by [32]. An intermittent failure is the loss of some function or performance characteristic in a product for a limited period of time and subsequent recovery of the function. A common example of this phenomenon in our lives occurs when a computer hangs up to start. Clearly a "failure" has occurred. However, if the computer is rebooted, it often works again. A failure that is not intermittent, in the other hand, represents the termination of the ability of an item to perform a required function. ("IEEE 100", 2000).

2.1.2.3.2 Invalid failure cause

Sometimes the device presents a failure, but it has not been originated by failures modes of interest. This is usually because the cause of the failure was actually human error. According to [5], "failure cause" is the set of circumstances that leads to failure, and "human error" is the discrepancy between the human action taken or omitted, and that intended or required. To be labeled as invalid failure cause in the reliability analysis, the circumstances are not limited to misuse or accidents, but also shipping damages, for example. The cause of the failure is most of the times delicate to be identified.

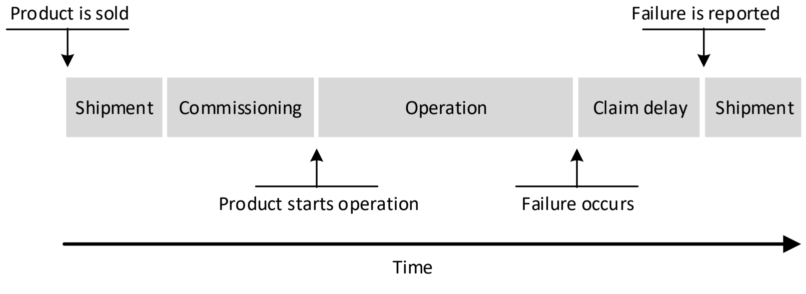
2.1.2.3.3 Other services

An entry in a claim database does not necessarily stand for a reported failure. In the case of Bosch Rexroth, for example, besides repairs, it is also offered services of remanufacturing and adaptation of products. In the remanufacture, defective parts are repaired and worn parts are exchanged for new ones. Adaptation is usually requested when customers ordered equipment with wrong hardware or software configurations.

2.1.2.4 Delays and time deficiencies

The delivery and claim databases usually contain two dates: the date of first shipment and the date that the defective product arrived at the repairing company. Straightforward estimation of the failure time distribution, treating the difference of these two dates as the time-to-failure, would lead to an overestimate of the reliability of the system [34]. This happens because the observed time window is the length of the longer pipeline that includes many other processes, and their inclusion would imply that the product lived longer than its actual time-to-failure. The basic elements of a field claim pipeline of industrial components is demonstrated in Figure 1, where the gray boxes represent periods of time and arrows events.

Figure 1: Timeline structure of a field claim pipeline



Naturally, the difference between the occurrence of a failure and the start of operation is the only relevant time period to the reliability assessment, although it is usually unknown. Especially in this field, commissioning time plays a big role in the delay for the beginning of operation. This is the time necessary to install and prepare a device into the desired working condition. Since the industrial components are normally installed in bigger and more complex machines with several axes, they only begin to operate once the machine is completely build and installed at the customers base. The term “claims delay” can be split into two types. Type I reporting delay assumes that a failure is reported immediately, but it might take some time before the claim database’s acknowledgement. Type II assumes that the failure might take some time to be reported, but delays in the database processing are negligible. [35] attempts to solve this problem by treating the delays as a random distribution. [36] deals with critic time applications, where they estimated the amount of occurred but not yet reported events (OBNR). [37] compares absolute and stochastic approaches for treating report delays. Besides delays, field data can also be aggregated, which means that it might not be possible to retrieve the exact dates of the events, since they can be grouped in months [23]. In the literature, many attempted to solve problems in which the aggregation period was too big relative to the desired analysis range [38].

2.1.2.5 Limited time window

Sometimes we want to make a prediction of the number of failures in a population expected by an amount of time t_a, even if the only observations up to t_b are available, with $t_a > t_b$. Therefore inferences and predictions involving extrapolation are often required [17].

2.1.2.6 Incomplete observations

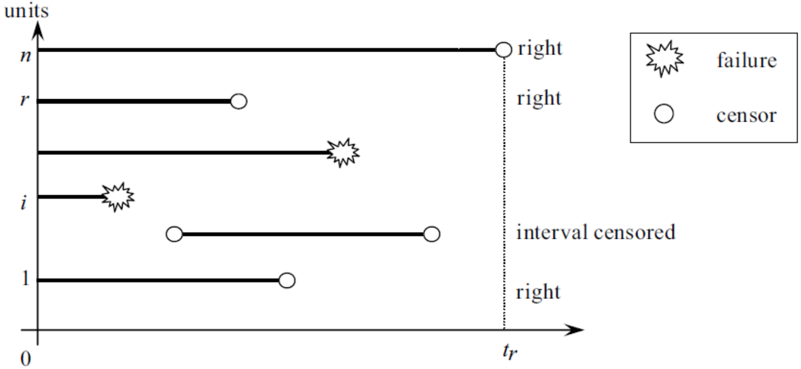
When the exact times of the expected events are obtainable for all samples under observation, the data is called complete, if not, the data is called censored or incomplete [39]. Censored data contains censored observations, which are believed to happen but the precise time is unknown. Instead, there is only an indication of the period of time that it happened or will happen [40]. This is broadly studied by both fields of medicine and reliability. The first one is normally concerned about treatments of patients, and a death may introduce uncertainties about its progress. Medical analysis also take into account truncated data, which are observations that, if not observable, it is not possible to affirm that they will happen [16]. In reliability however, since failures are always supposed to happen in a product’s life, data is usually not truncated. Examples of truncated data in the engineering field is the analysis of defective soldering of microchips on electronic boards, when not all boards can be verified [37]. Censoring can be classified by different taxonomies [41] :

I. **Type**:
   a. Type I: The end of observations is determined by a fixed time parameter t_c.
   b. Type II: The end of observations occurs when a number of failures is observed.
   c. Type III: It is a combination of the Type I and II. The end of the period of observation occurs at $\min(t_c, t_r)$, where t_c is a time limit and t_r is the time when r items failed.

II. **Plurality**:
   a. Individual: All units that have survived until a mutual censoring time t_c are considered suspensions. It is usually observed in laboratory tests.
   b. Multiple: Censoring time t_{c_j} varies from unit to unit. It is also called progressively, hyper and arbitrarily censored, and it is frequently found in field operations. It is also called Type IV by some authors [42].

III. **Direction**:
   a. Right: When an analysis is finished and some units are still functioning, their failure times are known to be beyond the current running times. These units are called in the literature unfailed, run-outs, survivors, removals, and suspended units [40]. An early suspension is one that was suspended before the first failure, while a late suspension is suspended after the last failure.
   b. Left: A failure time is known to be before a certain time t_c. It occurs if a sample is observed at t for the first time and it is already faulty. In this case, a failure must have happened before t.
   c. Interval: A failure time is contained in a time range. It is common in situations where failure cannot be constantly observed, but after every periodic inspection.

Figure 2: Censoring mechanisms



Source: [41]

Figure 2 shows visually the effects of different censoring mechanisms. Censoring is possibly the aspect of data deficiency that mostly deserves attention and that is widely studied in the academy. Field data are seriously censored and they must not be ignored during the reliability assessment, since they contain meaningful information [43] [44]. [3] present the effects of the assumptions and the ignorability of censoring mechanisms. [26] show the difference of censored and truncated data in a practical example of prediction for remaining life (RUL). [10] focus on the methods for reliability assessment of suspended tests. [25] show how to deal with heavily censored failure data in the analysis of transformers. [45] treats non-returned items after warranty as suspensions and proposes an approach using maximum likelihood estimation to solve this problem. [46] considers a multiple censoring case in a warranty study with staggered entry, that is, where units with starts on different dates.

2.2 INTRODUCTION TO MACHINE LEARNING

2.2.1 Introduction to notations and process pipelines

Artificial Intelligence (AI) is a branch of computer science that focuses on allowing machines to perform functions that normally require human intelligence or interpretation [47]. Machine learning (ML), a branch of artificial intelligence, was defined by Arthur Samuel [48] as the "field of study that gives computers the ability to learn without being explicitly programmed". This learning process occurs similarly to the biological one: it focuses on remembering experiences, adapting actions and generalizing solutions. One common characteristic shared across ML techniques is that it requires data in order to work. Data is used to fit a model, and then it is replaced by the model. Essentially, a set of *input* variables (also called set of *observations* or *feature vector*) must be fed to a model. The length of the vector is a measure of dimensionality.

The two most famous variants of this branch are *unsupervised* and *supervised learning*. The first is mostly concerned on discovering patterns and relations among collections [49]. A well-known application for this approach is the categorization of data together, called *clustering*.

In supervised learning, an addition set of *output* variables is necessary to train the model. These are also called *labels*, and each input vector requires an associated label vector. The labeled input is a training *example* and can be seen as a pair $(\boldsymbol{x}, \boldsymbol{y})$, where $\boldsymbol{x}$ and $\boldsymbol{y}$ are the input and output vectors respectively. It is assumed that the inputs are independent and the outputs are influenced by the former. If the predicted outputs are continuous or quantitative, it is said to be a *regression* or *prediction* problem. If the predicted outputs are categorical or qualitative, then the problem is of *classification* [8].

An example of a supervised learning process starts by initializing a model and using it to make a prediction $\hat{y}_i$ for a given input x_i. The correct label y_i is presented and an error is calculated using a *loss* or *cost* function. The goal is to minimize the loss using *optimizers* that tweak internal parameters of the model [50]. Parameters that are not adjustable by the optimizer and are related to either taxonomy or configuration of methods utilized by the model are called *hyperparameters*. These are of responsibility of the ML design engineer.

2.2.2 Logistic regression for soft classification of text

In this work, the development chapters will address the cleaning of the data and the discovery of fields that are hard for a programmed computer to interpret. These are text fields written by humans during the repair operation.

Natural language processing (NLP) is an area of study that explores how computers can be used to understand and manipulate natural language text or speech [51]. Its first work presented to the public was an automatic translation from Russian to English in 1949 and it has been evolving since then [52]. In the last decades it has been proving immense capabilities in applications such as language generation and understanding, speech recognition, translations and spelling correctors [53]. When it comes to understanding the text fields of entries of the claim database in order to classify them binarily, it is said to be a *text binary classification* problem. Also defined as *text polarity classification* by some authors, it belongs to the field of *sentiment analysis*.

While sentiment analysis relies on models (predictors) like in other problems addressed by machine learning techniques, it holds a big field of research aiming on improving the understanding of language by these models. In order to accomplish that, raw text must be converted to a feature vector understandable by the algorithm. This process is called *feature engineering* and its state-of-the-art techniques were recently studied by [54]. The author explains the representation of textual data as a vector of feature weights, whereas the words (terms, phrases) are used to form features. This process includes the following techniques:

Feature extraction or *construction* involves creativity of the developer to come up with measures that are relevant to the identification of correct labels of the given inputs. The main consolidated techniques are following described, but the designer is free to create domain-specific features constructors.

- Tokenization: It is a fundamental technique for *text segmentation* that focus on splitting texts into individual words. It must be capable of dealing with different punctuation patterns. Traditionally, it can also eliminate irrelevant symbols or *stop words*, which are words or individual letters that are too general to provide useful information [55]
- Stemming and lemmatization: These techniques are related to *text normalization* and aim to reduce the

complexity of the vocabulary and increase efficiency by joining words that share the same radical or meaning together. While the first utilizes mechanized algorithms to cut out custom suffixes and prefixes, the second is based on the morphological analysis of word, usually implying more time consumption [56].

- N-grams: In computational linguistic, an n-gram is a contiguous sequence of n items from a given sequence of text. These items can be phonemes, character, words, and others [57]. For example, the sentence “Classification is possible” have three word-n-grams of size 1 (unigrams): “Classification”, “is” and “possible”; and two word-n-grams of size 2 (bigrams): “Classification is” and “is possible”. When a negation (such as the word “not”) is included in the text, it does not represent anything alone, but it modifies the meaning of adjacent words. This problem is called *polarity shift.* [58] and [59] highlight how the use of n-grams help capturing negations.
- Part-of-speech (POS) tagging: It assigns tags to words to label them according to their category or lexical category [56].

Feature selection is a form of dimensionality reduction, as it tries to ignore irrelevant features and select the most important ones.

Feature weighting is a technique aimed on modifying the importance of features in the training and prediction. It is for example related to the form that the presence of features in a sentence is accounted. The simplest approach is the *binary*: it assigns *True* if the feature is present, and *False* if not. The *term frequency* counts the number of times that this feature is identified in a single input. One of the most used weighting schemes is the *term frequency-inverse document frequency* (TF-IDF). This approach assumes that if a word is important for the classification of a document, it should repeatedly appear in that document whereas it should rarely appear in other documents [60]. It is mathematically given by (2.1), where the parameter tf_{ij} is defined as the number of times word i appears in document j, df_i as the number of documents in which word i appears at least once, and N_d the number of available documents.

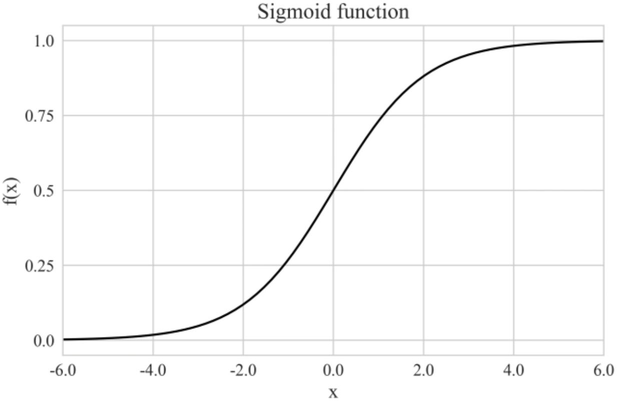
$$TF\text{–}IDF_{ij} = tf_{ij} \times \log\left(\frac{N_d}{df_i + 1}\right) \tag{2.1}$$

Feature smoothing are numerical transformations applied to the weights assigned to features. Examples are *Lidstone*, where a positive value is summed to all features, and *Good-Turing*, where a power function with flexible exponent is applied. Some authors claim that feature smoothing can improve the estimation performance of the model [61].

Sentiment analysis on labelled data for classification can use many of the models available in the supervised machine learning branch. Different algorithms were presented by the survey [62], such as Naïve Bayes and Neural Networks. [63] and [64] also highlighted the advantages of using multiple classifiers in a voting framework, known as *ensemble* approach.

A special type of classification is the *soft classification*. It assigns probabilistic values of belonging to different labels, while the *hard classification* assigns only one label to an instance. One way of achieving this is by using the logistic regression model [65]. This model is very similar to a linear regression, but a sigmoid function is introduced at the output in order to map the prediction smoothly in the range from zero to one. The sigmoid function $f(x) = 1/(1 + e^{-x})$ is graphically presented by Figure 3.

Figure 3: Plot of the sigmoid function



Finally, the logistic regression is defined by the equation (2.2) [66], where $\boldsymbol{x}$ is the feature vector of size d, $\boldsymbol{w}$ is the weight vector of size $d+1$, and $h(x)$ can assume values between zero and one.

$$h(\boldsymbol{x}) = \frac{1}{1+e^{-(w_0+w_1x_1+w_2x_2\,\dots\,w_dx_d)}} = \frac{1}{1+e^{-\boldsymbol{w}^T\boldsymbol{x}}} \tag{2.2}$$

$$h(\boldsymbol{x}) = \begin{cases} > 0.5, & if\ \boldsymbol{w}^T\boldsymbol{x} > 0 \\ < 0.5, & if\ \boldsymbol{w}^T\boldsymbol{x} < 0 \end{cases} \tag{2.3}$$

This model can be seen as a binary hard classifier if the output values are rounded up or down. However, the use of the continuous range of output values gives a probability of the input belonging to one of the two classes. For example, if $h(x) = 0.75$, there is 75% probability that the input $\boldsymbol{x}$ belongs to the class 1 and 25% probability that it belongs to the class 0.

This model is solved through MLE by minimizing the following negative log-likelihood function, where n is the number of training examples and $C > 0$ is a penalty parameter [67]:

$$L(w) = \mathrm{C}\sum_{i=1}^{n} \log\left(1+e^{-\boldsymbol{w}^T x_i}\right) + \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} \tag{2.4}$$

2.2.3 Artificial neural networks

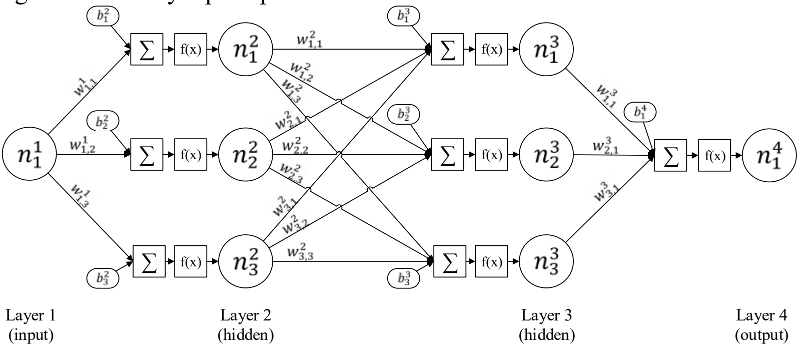
Artificial neural network (ANN) models are very popular for modeling a variety of relationships. They are distributed nonlinear machines built from many different processing elements (PEs) (sometimes called nodes or neurons), where each PE receives connections from other PEs and/or itself. The signals transmitted between the PEs are scaled by adjustable parameters called weights. Each PE produce only one output that is a nonlinear function of the sum of its contributions. This output become either the system's output or is sent to one or more PEs [68].

According to [69], an ANN can also be expressed by a *directed graph*, which is a graph of nodes connected by edges, where the edges are represented by arrows pointing the direction of the flow of signals. The graph's architecture describes the network layout. Essentially, it embraces the following properties:

- Each node n_i^l of layer l is considered a state variable.
- A real valued weight $w_{i,j}^l$ is associated with each link between nodes i of layer l and j of layer $l + 1$.
- A real valued bias b_i^l is associated with each node i of layer l.
- A transfer function is defined for each node, which determines its state as a function composed of its bias, the weights of incoming links, and the states of nodes connected to it.

This work will cover multilayer perceptrons (MLPs), which are feed-forward networks with one or more layers of nodes (hidden) between the inputs and outputs [70]. It has been proved by [71] that MLPs with as little as one hidden layer are considered universal approximators, provided that sufficient hidden units are available. For understanding purposes, an example of an MLP architecture is presented in Figure 4.

Figure 4: Multilayer perceptron with architecture 1-3-3-1



In an MLP, the transfer function of a node n (except for input nodes) can be defined as (2.5), where k is the number of nodes in layer l and $f(x)$ is the activation function associated with the node n_j^{l+1}.

$$n_j^{l+1} = f\left(b_j^{l+1} + \sum_{i=1}^{k} n_i^l w_{i,j}^l\right) \tag{2.5}$$

Except for the output layer, activation functions of an MLP must express nonlinear relationships in order to be meaningful for the learning process and performance. If different layers utilized only linear functions, they could be simply summarized by a single linear transformation. One of the most traditional activation functions is the sigmoid $f(x) = 1/(1 - e^{-x})$. However, as shown by [72], its antisymmetric adaptation given by the hyperbolic tangent function $\tanh(x) = (1 - e^{-2x})/(1 + e^{-2x})$ provides faster learning rates. Nevertheless, when the Rectified Linear Unit (ReLU) was introduced by [73], it soon became a better replacement for both the sigmoid and hyperbolic tangent functions in hidden nodes due to its impressive performance [74]. Deep neural networks with ReLUs can train several times faster than their equivalents with *tanh* units [75]. The ReLU function is defined by (2.6).

$$ReLU = \begin{cases} 0 & if\ x \leq 0 \\ x & if\ x > 0 \end{cases} \tag{2.6}$$

A good choice for the initial values of the synaptic weights and biases are essential for a successful network design. Values too big may saturate neurons and values too small may set them in stationary regions [76]. There are many approaches for randomly choose the most appropriated values, such as using uniform or normal distributions. The Glorot normalized initializer was proposed in 2010 [77] and is the current state-of-the-art technique [78]. The weights are initialized according to (2.7), where $U(a, b)$ is the uniform function in the range a to b, w^l are the weights between layer l and l +1, and k^l and $k^{l+1}$are the number of nodes of layers l and $l + 1$ respectively.

$$w^l \sim U\left(-\frac{\sqrt{6}}{\sqrt{k^l + k^{l+1}}}, \frac{\sqrt{6}}{\sqrt{k^l + k^{l+1}}}\right) \tag{2.7}$$

When feeding a neural network, one can use either the original data or transformed data. There are ways of applying transformations, but the two most common are the linear and the statistical normalization [79]. The former linearly interpolate the data into a customizable range that can be $[0,1]$ or $[-1,1]$ for example. The latter computes the deviation from the mean and divides each value by the standard deviation. This helps to deal with different input dimensions and matching with appropriate network working range of values. Tests conducted by [80] achieved five to ten times less estimation errors and one order of magnitude less training time when using normalized values.

The training process is realized by means of a learning method, which is an optimization algorithm that modifies weights of the ANN in order to obtain desired outputs for given inputs [81]. A popular method for the training of multilayer perceptrons is the back-propagation, which is based on the gradient descent algorithm [76]. It consists of two phases:

I. **Forward phase**: With fixed weights, the signal is propagated throughout the network until the output layer. Each node computes its output by means of a transfer function.

II. **Backward phase**: An error signal is calculated by a cost function by comparing the network's output with the desired one. This signal is propagated back through the network and successive adjustments are applied to the weights based on the computed gradient vector of the error surface with respect to the weights connected to the node.

The updating of the weights is realized only during the backward phase, which can occur in different times depending on the type of gradient descent algorithm. These make a trade-off between the accuracy of the parameter update and the time it takes to perform an update [82]:

I. **Batch gradient descent** (BGD): This method calculates the gradient of the cost function only after the whole dataset has been forward propagated. Therefore, the convergence is normally very slow for big datasets.

II. **Stochastic gradient descent** (SGD): Instead of performing an update after the whole dataset is processed, this method updates the weights iteratively after each training example (also called on-line learning). This is much faster but produces a high variance in the error function. This fluctuation can sometimes help avoiding getting stuck in suboptimal solutions.

III. **Mini-batch gradient descent** (MBGD): This is a hybrid solution that aims to execute the backward phase once a fixed number of training examples (batch size) is processed. If the batch size is set to one, it becomes the SGD, while if it is set to the size of the dataset, it becomes the BGD.

In their most original form, GD algorithms are bounded to some performance limitations such as static and equal learning rates shared across the entire network, slow training on large datasets, and exposure to suboptimal solutions. Adaptive moment estimation (Adam) was proposed by [83] and has become one of the most consolidated improvement for the gradient descent method. In only four years, Adam has been cited over sixteen thousand times in the literature and has become the default optimizer of big deep learning computing libraries such as TensorFlow. It has been proved that Adam is faster than any other optimizer [84]. Under the Adam optimization, each weight of the network keeps track of the past gradients in order to adapt the future learning rates, while accounting with a momentum induced by the past weight correction.

2.3 ESTIMATION OF RELIABILITY MODEL

The reliability model of interest in this work is stochastic and can be made out of a group of continuous functions that relates time (or a time period) with a probability of failure. Their notation utilizes the probability function P with the random variable $T \geq 0$ and observable variable $t \geq 0$, where T represents the time-to-failure for a given item, and t the current operating time of the item. Operating time is the total time for which an item is in an operating state [5]. The four most relevant reliability functions are following described. They are fully interchangeable, which means that it is possible to retrieve any of them, if one is provided, as shown in Table 1 [42].

I. Cumulative distribution function (CDF) $F(t) = P\{T \leq t\}$. It is the probability that an item fails in the period $(0, t]$. It is also called unreliability function.

II. Survival function $S(t) = P\{T > t\} = 1 - F(t)$. It is the probability that an item survives the interval $(0, t]$.

III. Probability density function (PDF) $f(t) = \mathrm{P}\{\mathrm{t} < \mathrm{T} < \Delta\mathrm{t}\}$. It is the probability that an item will fail in the time interval $(t, t + \Delta t]$

IV. Hazard function $h(t) = \mathrm{P}\{\mathrm{t} < \mathrm{T} < \Delta\mathrm{t} | T > t\}$. It is the probability that an item will fail in the time interval $(t, t + \Delta t]$ when we know that the item is functioning at time t.

Table 1: Relation between reliability functions

	$F(t)$	$S(t)$	$f(t)$	$h(t)$
$F(t) =$	$-$	$1 - S(t)$	$\int_0^t f(u)du$	$1 - \exp\left(-\int_0^t h(u)du\right)$
$R(t) =$	$1 - F(t)$	$-$	$\int_t^\infty f(u)du$	$\exp\left(-\int_0^t h(u)du\right)$
$f(t) =$	$\frac{d}{dt}F(t)$	$-\frac{d}{dt}S(t)$	$-$	$h(\mathrm{t}) \cdot \exp\left(-\int_0^t h(u)du\right)$
$h(t) =$	$\frac{dF(t)/dt}{1 - F(t)}$	$-\frac{d}{dt}\ln S(t)$	$\frac{f(t)}{\int_t^\infty f(u)du}$	$-$

There are two main approaches to fitting a reliability model to data. The first derives empirical non-parametric reliability functions from the data. The second is a parametric approach and assumes that there is a statistical distribution that suits the data, and tries to infer its parameters directly. Since the second approach usually follows the first one [41], both are evaluated in this work.

2.3.1 Empirical reliability function

Given a collection of observations of events in times $t_1, t_2 \dots t_N$, the empirical methods studied in this work are used to infer the unreliability $0 < \hat{F}(t_j) < 1$ for each event t_j, where the event e_j is a failure and not a suspension. Nevertheless, it is also possible to infer $S(t)$ and $h(t)$ by other empirical manners. Because it gives as output a collection of coordinates $(t, \hat{F}(t))$, it is also known as the plotting method [15].

A survey showed that many different estimators for the desired function have been proposed and tested by scientists and companies. Their simplest forms are $\hat{F}(i, N) = i/N$ and $\hat{F}(i, N) = (i - 1)/N$, which belong to the family of mean rank estimators. However, when $i = 1$ (first event) or $i = N$ (last event), they estimate undesired values for unreliability $\hat{F}$ (0 and 1). If the estimations $\hat{F}(j_1, N) = 0$ or $\hat{F}(j_2, N) = 1$, it would mean that it is *absolutely* impossible to observe a failure in the population before j_1 and after j_2 respectively [85]. Besides that, mean ranks also provide biased estimators [15]. Therefore an estimator called median rank (MR) was proposed by [86]. It is based on the binomial distribution (2.8), and a solution Z for each failure j of a collection of N failures must be evaluated [87].

$$0.5 = \sum_{k=j}^{N} Z^k N_k (1 - Z)^{N-k} \tag{2.8}$$

[88] demonstrates that immense computational power is necessary to solve (2.8), as the required time grows exponentially and only twenty failures led to three hours of calculation in a personal computer. Equation (2.8) is said to return the exact median rank value, but in order to reduce the calculation complexity different authors proposed approximations. Most of them fits in the form of (2.9) [89], where C is a constant that is set so that the deviation between the approximation and the exact value is minimized.

$$\hat{F}(i, N) = \frac{i - C}{N + 1 - 2C} \tag{2.9}$$

One of the most widely used approximation was proposed by [90]. It is commonly called Bernard's MR approximation (2.10), and is equivalent to setting $C = 0.3$ [91]. This approximation is recommended by [39] and [15]. [92] proves its higher accuracy in comparison to simpler estimators such as the mean ranks. [87] shows that the Bernard's approximation reaches deviation errors up to 0.45% in comparison to the binomial equation, when dealing with a sample of size five.

$$\hat{F}(i,N) = \frac{i - 0.3}{N + 0.4} \tag{2.10}$$

However, there are other estimators that deserve attention as well. Besides the estimators previously cited, Filliben's estimator [93] is also taken into account in a accuracy comparison realized by [94]. This estimator assumes $C = 0.3175$, resulting in (2.11).

$$\hat{F}(i,N) = \frac{i - 0.3175}{N + 0.365} \tag{2.11}$$

[89] provides a broad comparison of the performance of the previously cited estimators. The measure used is the largest deviation between an approximation and the exact value of the median rank for a given sample, excluding the first and the last failure. Table 2 summarizes the experiment results.

Table 2: Deviation of approximations of the median rank [89]

Name	**Estimator**	**Largest deviation**
Filliben	$\hat{F}(i,N) = \frac{i - 0.3175}{N + 0.365}$	0.00028
Bernard	$\hat{F}(i,N) = \frac{i - 0.3}{N + 0.4}$	0.00125
Hazen	$\hat{F}(i,N) = \frac{i - 0.5}{N}$	0.0145
Weibull	$\hat{F}(i,N) = \frac{i}{N + 1}$	0.0215
IEC 56	$\hat{F}(i,N) = \frac{i - 0.5}{N + 0.25}$	0.0385

The presented methods are the most used in the literature, but [95] reviews almost thirty estimators that were proposed between the year of 1814 and 2017. Before Weibull distribution was presented, they were utilized for working with general extreme value distributions.

In summary, the median rank functions transform a collection of ranks i of observed failures to their respective unreliability value in the range from zero to one. In complete sets, the rank i is equivalent to j, which is the position of the event in a chronological order. However, the inclusion of suspensions in the set requires an adaptation of the rank i. Johnson provided this adaptation in his article [86] and called as *adjusted median rank*. It is widely used by scientists and industrials and are also suggested by [96] and [41]. Johnson's method is calculated iteratively event after event chronologically. The value Δi is initialized to 1 and is recalculated in the occurrence of every suspension as (2.12). In the occurrence of a failure, (2.13) is used instead. The function $s(t)$ denotes the number of units survived until time t and $i_{\_}$ is the value of the rank in the previous iteration.

$$\Delta i = \frac{N + 1 - i_{\_}}{s(t) + 1} \tag{2.12}$$

$$i = i_{\_} + \Delta i \tag{2.13}$$

The rank adjusting methods is based on the following rules [10]:

- A suspension event implies on the calculation of a new increment Δi.
- Only failure events have a failure order (rank) i and its respective value unreliability value.
- If a suspension and a failure occur simultaneously, the failure is processed first.
- If there are k suspensions or failures at the same time, the calculations of rank increments and ranks must be executed k times.

Drew Auth's simplified version of Johnson's rank adjusting method was presented by [15], which proved to provide the same results. One of the computational advantages is that it is not necessary to calculate the rank increment Δi. Auth's method is defined by (2.14) and is recommended by [39].

$$i = \frac{s(t) \cdot i_{-} + N + 1}{s(t) + 1} \tag{2.14}$$

In order to demonstrate the use of the Auth's rank adjusting method, an example with a censored sample of size $N = 6$ is proposed and solved in Table 3. The median rank field is calculated using Filliben's estimator (2.11).

Table 3: Demonstration of the rank adjustment method

Time	**Event position**	**Event type**	**Number of units survived at t**	**Adjusted rank**	**Median Rank (%)**
t	j	e_j	$s(t)$	i	$\hat{F}(i, N)$
100	1	F	6	$\frac{6 \cdot 0 + 6 + 1}{6 + 1} = 1$	11%
200	2	S	5	-	-
300	3	F	4	$\frac{4 \cdot 1 + 6 + 1}{4 + 1} = 2.2$	30%
400	4	F	3	$\frac{3 \cdot 2.2 + 6 + 1}{3 + 1} = 3.4$	48%
500	5	S	2	-	-
600	6	F	1	$\frac{1 \cdot 3.4 + 6 + 1}{1 + 1} = 5.2$	77%

2.3.2 Parametric modelling with statistical distributions

Empirical functions are very simple to generate, but difficult for a computer to understand due to its discrete nature. Sometimes parametric approaches are essential in the reliability analysis, and this is done using suitable theoretical distributions. When the model is based on a parametric distribution, the reliability functions as described in chapter 2.3.2 take, besides time t, a vector of parameters θ. Its shape and domain depend on the applied distribution and is usually small, containing between one and four parameters. For large datasets, it is more practical to have a few parameters describing the whole model behavior than estimating an empirical function for each time stamp. Parametric models make possible to extrapolate in time the lower and upper tail, and also provides smooth estimates [17]. Statistic holds a large collection of distributions, however a few of them are usually much more employed than others. The next subchapters will present the most common theoretical distributions used in reliability assessment.

Typically, the traditional measures of statistical distributions (e.g. mean, standard deviation or median) are not of primary interest. Instead, engineers, managers, and customers are interested in specific measures of product reliability such as failure probabilities or the B% life [17]. The B% life is a quantile mensuration proposed by Waloddi Weibull that has become a standard for quoting life expectancy of some components, such as bearings. This measure has the same unit of the studied t, and its value is the time when a certain percentage of the population fails. The "%" symbol is replaced by a number between 0 and 100 to specify the percentage of failed items. For example, B10 is the time when 10% of the population fails, and $F(B10) = 10\%$.

2.3.2.1 Theoretical distributions used in reliability analysis

2.3.2.1.1 Exponential

The exponential distribution applied to reliability implies that the probability of occurrence of failure is constant over time. It is based on a single parameter $\lambda > 0$, which is equivalent to the hazard function h. Its unreliability function is defined by (2.15).

$$F(t) = e^{-\lambda t} \tag{2.15}$$

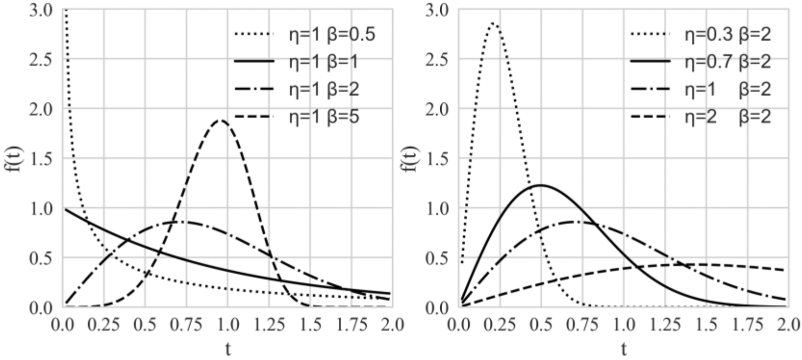
This distribution has many applications in many fields, for example to describe the decay of radioactive atoms or in the theory of waiting queues [97]. According to Rausand, it is the most commonly used life distributions in applied reliability analysis, especially for small and reliable electronics. The reason for this is its mathematical simplicity and that it leads to realistic lifetime models for certain types of items [42]. However, its easiness is also a reason why this distribution is widely misused in inappropriate applications. Due to its constant failure rate, it cannot model the life of mechanical components (e.g., bearings) subject to some combination of fatigue, corrosion, or wear. A distribution with an increasing hazard rate is, in such applications, usually more appropriate [17].

2.3.2.1.2 Weibull

The Weibull distribution is one of the most widely used life distributions in reliability analysis. The distribution is named after the Swedish professor Waloddi Weibull (1887-1979) who developed the distribution for modeling the strength of materials. The Weibull distribution is very flexible, and can, through an appropriate choice of parameters, model many behaviors [42]. For this reason, it was also adopted by reliability practitioners and by other fields such as wind energy [9], [98]. Because of its popularity and versatility, and the letters assigned to the parameters may change. This work uses the standard described in [39] and most reliability books.

The Weibull distribution can assume many forms with different amounts of parameters. The most common one is the two-parameter Weibull, which uses the scale parameter $\eta > 0$ and shape parameter $\beta > 0$. The scale parameter is given in the same unit of t and represent the time of the 0.632 quantile of the unreliability function. The shape parameter, also known as slope in some plots, can assume different ranges: $0 < \beta < 1$ describes a decreasing failure rate; $\beta = 1$ is a special case where the failure rate is constant and the distribution is equivalent to the exponential; $\beta > 1$ models increasing failure rates. Other special cases occur when $\beta = 2$, and it becomes the Rayleigh distribution, and when $\beta \approx 3.4$, and it approximates a symmetrical distribution [96]. Figure 5 shows graphically the effects of changes in β and η.

Figure 5: Plot of probability density function based on a Weibull distribution



Sometimes, an additional location parameter γ might be necessary and the distribution is called a three-parameter Weibull. It is a shift in the time axis that can assume negative or positive values. When $\gamma = 0$, the three-parameter Weibull is simplified to the two-parameter version. When $\gamma > 0$, the location parameter can also be defined as failure-free time or minimum life, since no event can occur in this period. $\gamma < 0$ in the other hand, describes situations when the analyzed units begin to degrade even before they are set to operate, for example while waiting on shelves. Sometimes, practitioners want to fit their data into three-parameter distributions to gain more flexibility. However, more parameters require more sufficient data to fit [99]. Meeker recommends a minimum of one hundred observations [15].

Table 4 presents the mostly used functions associated to a Weibull distribution.

Table 4: Reliability functions of the three-parameter Weibull distribution

Unreliability function (CDF)	$F(t) = 1 - e^{-\left(\frac{t-\gamma}{\eta}\right)^{\beta}}$	(2.16)
Survival function	$S(t) = e^{-\left(\frac{t-\gamma}{\eta}\right)^{\beta}}$	(2.17)
Hazard rate	$\lambda(t) = \beta \frac{(t-\gamma)^{\beta-1}}{\eta^{\beta}}$	(2.18)
Probability Density Function (PDF)	$f(t) = \beta \frac{t^{\beta-1}}{\eta^{\beta}} e^{-\left(\frac{t-\gamma}{\eta}\right)^{\beta}}$	(2.19)

2.3.2.1.3 Log-normal

A log-normal distribution can be fitted to the data if the random variable time-to-failure T follows a log-normal distribution, or in other words, if $\ln(T)$ follows a normal distribution. It is applicable when random variables are constrained by zero but have a few very large values. The resulting distribution is asymmetrical and positively skewed. It has many uses in engineering, biology and economy [100]. Similarly to Weibull, it is also defined by the parameters of scale $-\infty < \mu < +\infty$, shape $\sigma > 0$ and location $-\infty < t_0 < +\infty$. Its CDF is defined by (2.20).

$$F(t) = \frac{1}{\sigma_N \sqrt{2\pi}} \int_0^t \frac{1}{u} exp\left[-\frac{1}{2}\left(\frac{\ln(u) - \mu_N}{\sigma_N}\right)^2\right] du \qquad (2.20)$$

Log-normal is less convenient computationally but are still applied frequently [24]. [15] explains that the Weibull family is normally preferred, and the engineer should appeal to the log-normal distribution in case of bad fit. It is often the case that both may provide a good fit, especially in the middle of the distribution. However, the Weibull distribution has an earlier lower tail of $F(t)$ and produces a more pessimistic estimate of the component life [96].

2.3.2.2 Estimation

Due the highest relevance of the Weibull distribution to this theme, this chapter will discuss approaches of estimating its parameters. Different methods have been suggested so far, but the two most commonly used today are the rank regression and the maximum likelihood (MLE). The discussion regarding the performances and applicableness of these two have been happening since decades in the reliability community.

One of the biggest advocates of the rank regression method is Dr. Robert B. Abernethy, which helped to improve reliability methodologies regarding Weibull, such as the creation of the Weibayes estimator. The rank regression is a technique that uses the calculated ranks (i.e. median ranks) to fit a line through a regression technique, such as least squares (LS), in order to estimate a two-parameter Weibull distribution. When the median ranks are used, it is called a median rank regression (MRR). Naturally, as observed from equation (2.16), the unreliability function $F(t)$ is not linear. In practice, transformations are applied to achieve equation (2.21), which is masked as a linear relation in the form of $y = ax + b$. Notice that $\gamma = 0$ is omitted because this method does not support the inclusion of the third parameter.

$$F(t) = 1 - e^{-\left(\frac{t}{\eta}\right)^{\beta}}$$

$$\frac{1}{1 - F(t)} = e^{\left(\frac{t}{\eta}\right)^{\beta}}$$

$$\ln\left(\frac{1}{1 - F(t)}\right) = \left(\frac{t}{\eta}\right)^{\beta}$$

$$\ln\left(\ln\left(\frac{1}{1 - F(t)}\right)\right) = \beta\ln(t) - \beta\ln(\eta) \tag{2.21}$$

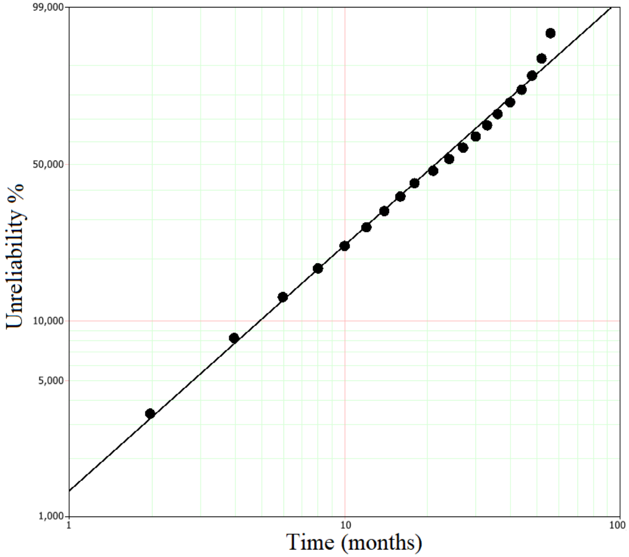
$$y = ax + b$$

Where:

$$y = \ln\left(\ln\left(\frac{1}{1 - F(t)}\right)\right)$$
$$x = \ln(t)$$
$$a = \beta$$
$$b = \beta\ln(\eta)$$

As proved in (2.21), when the data follows a Weibull distribution, the plot of Weibull’s median ranks forms a straight line, if the scale transformations $\ln(t)$ is set to the abscissa and $\ln\left(\ln\left(\frac{1}{1-F(t)}\right)\right)$ to the ordinate. Figure 6 is an example of such plot, which is called a Weibull probability plot. In this example, the X-axis is the age in months and the Y-axis is the cumulative distribution function of a sample of twenty failures.

Figure 6: Example of a Weibull probability plot



Plot made in software Reliasoft Weibull++

The slope of a Weibull probability plot is determined uniquely by the shape parameter, and the intercept by both shape and scale parameters. With the median ranks are plotted, a best fitting line must be found. For many years in the past, this whole process would be done manually by hand, including the fitting of the best line. However, today it is implemented in many software packages. When using the ordinary least squares (LS) to fit the best line, the goal is to minimize the sum of squared residuals of either the X or Y axis. The former is called rank regression on X (RRX) and the second rank regression on Y (RRY). [15] and [39] define the RRY as the standard and best practice over other regression methods, although [85] reports that a weighted least squares (WLR) might provide less bias for small samples. The principle of RRY is to find the estimates $\hat{a}$ and $\hat{b}$ of the linear function through the minimization of the right side of the equation (2.22).

$$\sum_{j=1}^{N}\left(\hat{a}+\hat{b}x_i-y_i\right)^2 = min\sum_{j=1}^{N}(\hat{a}+bx_i-y_i)^2 \tag{2.22}$$

The minimization is realized as follows:

$$\hat{a} = \frac{\sum_{j=1}^{N} y_j}{N} - \hat{b}\frac{\sum_{j=1}^{N} x_j}{N}$$

$$\hat{b} = \frac{\sum_{j=1}^{N} x_j y_j - \frac{\sum_{j=1}^{N} x_j \sum_{j=1}^{N} y_j}{N}}{\sum_{j=1}^{N} x_j^2 - \frac{\left(\sum_{j=1}^{N} x_j\right)^2}{N}} \tag{2.23}$$

One of the goodness-of-fit measures applicable to evaluate the quality of the regression is the Pearson correlation coefficient r (2.24). It is recommended by [15] and it analyses how good is the correlation of the data to the linear model, or in other words, how well does the data fits in a Weibull distribution. An r next to one is a very good positive correlation indication, while $r = 0$ indicates no correlation.

$$r = \frac{\sum_{j=1}^{N} x_j y_j - \frac{\sum_{j=1}^{N} x_j \sum_{j=1}^{N} y_j}{N}}{\sqrt{\left(\sum_{j=1}^{N} {x_j}^2 - \frac{\left(\sum_{j=1}^{N} x_j\right)^2}{N}\right)\left(\sum_{j=1}^{N} {y_j}^2 - \frac{\left(\sum_{j=1}^{N} y_j\right)^2}{N}\right)}} \tag{2.24}$$

Besides rank regression, another widely used method to find the distribution parameters is the maximum likelihood estimator (MLE). Although it relatively more computationally complex, it does not require median rank estimates and can directly infer the distribution parameters from the data. It also provides excellent support to censored data, such as left censored, interval censored and right censored [96].

The MLE equations were firstly introduced by [101]. The equation designed for both exact failure times and for right suspensions is defined in (2.25). Note that $f(t;\theta)$ is the probability density function of a Weibull distribution with parameters θ, and $e_j = 1$ when the observation j is a failure. For a two-parameter Weibull distribution, the likelihood estimator takes the form of (2.26).

$$L(\theta) = \prod_{j=1}^{N} \left[f(t_j;\theta)\right]^{e_j}\left[1 - F(t_j;\theta)\right]^{1-e_j} \tag{2.25}$$

$$L(\beta,\eta)=\prod_{j=1}^{N}\left[\beta\frac{t^{\beta-1}}{\eta^{\beta}}e^{-\left(\frac{t_j}{\eta}\right)^{\beta}}\right]^{e_j}\left[e^{-\left(\frac{t_j}{\eta}\right)^{\beta}}\right]^{1-e_j} \tag{2.26}$$

Like in many other cases, it is easier to achieve the maximization of the log of the likelihood function and, since a logarithm function is a monotonically increasing function, is equivalent to maximizing the original function [102]. After applying the natural logarithm transformation to (2.26), it becomes:

$$\ln L(\beta,\eta)=\sum_{j=1}^{N}e_j(\ln\beta-\beta\ln\eta)+(\beta-1)\sum_{j=1}^{N}e_j\ln t_j-\sum_{j=1}^{N}\left(\frac{t_j}{\eta}\right)^{\beta} \tag{2.27}$$

Equation (2.27) does not have an analytical solution, and therefore numerical methods such as the expectation-maximization (EM) are necessary.

Statisticians may be biased to use maximum likelihood estimation to solve this problem because of its popularity and good statistical proprieties such as consistency, invariance, as well as asymptotic normality and efficiency. Unfortunately, these are only met with a large amount of data, and undesired bias are commonly observed by the modelling of failure data [15]. According to [103] the MLE provides biased solutions that depend on the parameter β and the number of samples N, but its precise amount is unknown. [104] provide a review on unbiasing methods proposed in the literature. Some of them adapt the likelihood functions with weights in the attempt of creating unbiased estimators, while others are transformations applied to θ in order to adjust the bias. Abernethy suggests the use of a Reduced Bias Adjustment (RBA) factor to reduce the bias on the shape parameter, as described in (2.29). The RBA factor was empirically found by the author to be the 3.5th power of a common correction factor found in statistical quality control called C_4, defined in (2.29).

$$\beta_{unbiased}=\hat{\beta}\cdot RBA \tag{2.28}$$

$$RBA=C_4^{3.5}=\left[\sqrt{\frac{2}{N-1}}\frac{\left(\frac{N-2}{2}\right)!}{\left(\frac{N-3}{2}\right)!}\right]^{3.5} \tag{2.29}$$

Abernethy, [39] and many other authors recommend the use of an unbiased MLE only if there is a large amount of data or if the data is censored in many ways besides right censoring. Abernethy specifically suggests the use RBA-MLE for datasets with over five hundred observations, and is unsure for quantities between one and five hundred observations. For smaller data, MRR should provide the best results. The survey realized by [105] also shares this opinion. Another problem of MLE is that it does not provide a graphical visualization of the fit, as in MRR. If an MLE estimation is plotted in a Weibull probability plot, it usually does not look as a nice fit [15].

The discussion between MRR and MLE is complex and many advantages and disadvantages can be pointed out. For example, one of the major problems with MRR is that the times of the suspensions between the failures or after the last failure do not interfere the ranks' positions and values. This do not happen with MLE, since it uses all data including suspensions in the estimation [91].

[85] compares also others estimators for the Weibull parameters such as the method of moments (MME) and the good linear unbiased estimator (GLUE), however the least squares and MLE performed very well.

2.3.3 Non-parametric modelling

2.3.3.1 Kaplan-Meier Estimator

Non-parametric methods do not rely on parameters, and therefore no assumption regarding the life distribution is necessary. One of the most famous non-parametric estimators is the Kaplan-Meier (KM) estimator, proposed by E. L. Kaplan and Paul Meier on [106]. It belongs to the family of product-limit estimators, whose performance on censored datasets are compared by [107]. It is equivalent to a mean rank $\hat{F}(i,N) = i/N$ when used on complete datasets [94], but it was created with the goal of processing heavily censored datasets with ease [16] and does not require extra steps to deal with suspensions as in median rank approaches.

The KM estimator for the reliability function $S(t)$ is a right-continuous, non-decreasing step function (2.30) with jumps on failure times, where each step size depends on the pattern of the previous suspensions [108].

$$\hat{S}(t) = \begin{cases} \prod_{j:\, t_j \leq t} \left[1 - \frac{r(t_j)}{N - j + 1}\right]^{e_j} & if\ 0 \leq t \leq t_N, \\ 0 & if\ t > t_N\ and\ e_N = 1, \\ undefined & if\ t > t_N\ and\ e_N = 0 \end{cases} \tag{2.30}$$

This estimator is based on the assumption of noninformative censoring, which means that knowledge of a censoring time for an item is statistically independent of its failure time. It is well defined except for times larger than the observed study time t_N. This problem is also addressed by [107] and is undesired in the analysis of claim data observed in a short period of time. Nevertheless, it is widely used in the medical industry, where it was created [15].

2.3.3.2 Artificial Neural Networks for the Reliability Assessment

Artificial neural networks (ANN) is a technique that belongs to the field of AI that have attracted considerable interest over the past years in series forecasting, mainly due to their capacity of learning from experimental observations and generalization [109]. Although an ANN is composed of many weights that "parametrize" the network, it is considered a non-parametric approach because its black-box characteristic hides the weights from the prediction.

ANN holds the ability to understand complex non-linear relations between input and output patterns and have achieved many successful results in the field of reliability. Especially when the underlying mathematical model is complicated to be estimated, their parallelism and multi-parametric character make them a powerful computational tool [110], [111]. They have been used in many applications such as preventive maintenance in the experiments of [112] and [113] for predicting unreliability indexes. [114] uses ANN in the prediction of RUL using also data from previous suspensions. [115] also explored hybrid approaches of fuzzy neural networks for predicting reliability.

The use of neural networks for predicting reliability for non-monitored systems, however, has not earned too much attention. One of the first works done in the direction was realized by [116], where he trained neural networks to identify underlying distributions of data, and obtained better results than the chi-square test for small samples. An hybrid approach was introduced by [117], with a called neural-Weibull lifetime model. [118] pioneered the first approaches to this problem by disregarding theoretical distributions completely. He compared ANN models with other distributions using data from helicopter parts and achieved much better performance with the former. [119] indicated the greater accuracy of ANN over analytical models. According to him, a big advantage is that the assumption that the data follows a specific distribution does not need to be made when using this IA technique, since it automatically develops its internal model relationships. In addition to statistical distributions and ANN, [120] also compared the performance of logistic regression in the modelling of survival functions for patients with breast cancer. A comparison between four different approaches of feed-forward neural networks was provided by [121], however they still do not cover its entire potential.

A great review published by [19] compared many techniques of the IA branch in the task of modelling the cumulative distribution function of some components of three types of industries. The random forests (RF) and ANN obtained the best scores. In a study released by [18], they showed that even simple feed-forward networks with only one hidden layer has proven to achieve better performance than LS when creating a model from the median rank positions. The authors utilized nineteen groups of field data for this test. In this approach, a collection of pairs $\left(t_j, \hat{F}(t_j)\right)$ are used as input and output data respectively, where $\hat{F}(t_j)$ is the median rank estimate of the failure time t_j. Therefore, the neural network only uses one input and one output node. The authors also showed that a minimum of five neurons in the hidden layer are necessary for an optimum performance, and more than that does not provide much improvement.

3 DATA PREPROCESSING

This chapter approaches the raw data available for the estimation of the models. This is not yet usable and therefore a preprocessing phase is necessary. This includes importing, cleansing and feature extraction. These operations are following described addressing solutions to the obstacles listed in chapter 2.1.2.

3.1 APPROACHES TO THE FIELD DATA

The reliability analyses approached here considers the subjects as non-repairable systems, even though they are claimed to be maintained. This is because only the first failure of each unit should be considered, ignoring the effects that a repair causes on a product's future performance [87].

Since units are placed into operation in different times, it is said that this is data with staggered entry. Of course, failures are represented in only part of the claims and both are much less than the total number of deliveries, indicating the presence of censoring mechanisms i.e. incomplete observations in a limited time window. Censoring occurs when the exact date of failure of a product could not be observed, and censoring mechanism is the process that leads to a censoring. The field data addressed by this work is subjected to the following censoring mechanisms:

I. Units have not failed yet. This represents a case of *end-of-study* censoring, and its quantity is denoted by N_{eos}.

II. Units have failed and not reported. This is a *loss-of-follow-up* case, and its quantity is denoted by N_{fup}.

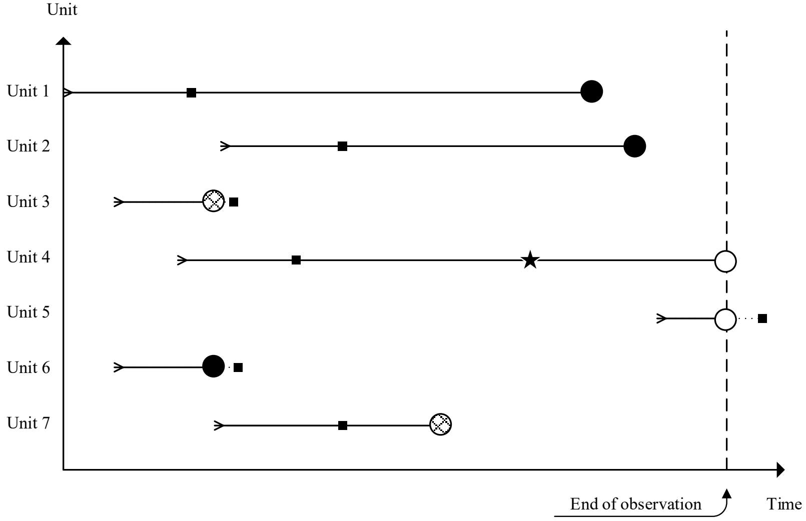
III. Units are claimed with failures that are not of interest, such as human error [5]. This is a suspension caused by *loss-to-different-failure-mode*. The sum of all suspensions due to this mechanism is denoted by N_{dfm}.

The three types of censoring described above are considered multiple or progressively right censoring. In a given sample of N observations, the total quantity of suspensions is given by $N_{susp} = N_{eos} + N_{fup} + N_{dfm}$. In these cases, a censoring time t_c when the unit is last seen in operation (and therefore a failure must occur after it) is necessary for the evaluation. This work will focus on identifying these mechanisms, but more importantly, on how to differentiate a suspension from a non-censored event. In order to use the techniques addressed in the literature review, one must assume that the existing failure mechanisms satisfy the requirement of independent censoring [42]. This means that the censoring time t_{c_j} for a unit j is completely independent of its actual random failure time T_j. This is considered random or non-informative censoring. In some applications, the end-of-study censoring sometimes do not satisfy this, for example if a destructive test is set to finish just before a few units are failing in order to save expenses, but it is not this case since the units are placed randomly into operation. The decision of a customer of not reporting the failures must be considered independent of the time-to-failure of these units. Finally, it is assumed that failures not related to physical failure modes are also random, for example if a user spilled water on an electronic device.

One of the most important assumptions is to say that a customer possessing an item under warranty will always report a failure, since it is free of charge. Consequently, if an item is not claimed during the warranty, it is assumed that is has not failed. A warranty contract follows all items sold by Rexroth. This is a period of two years in which a repair can be realized free of charge, given that the reason for the failure was a lack of design or reliability, and not a human failure, for example. The products can sometimes be sold with a customized warranty contract, but these exceptions are not considered.

The example in Figure 7 is built to demonstrate the different censoring possibilities present in this problem. The graph displays seven units and their timeline. The end of observation can be simply the actual day of analysis. A fully painted circle represents a claim with a product failure, a dashed circle represents a claim without a failure (*loss-to-different-failure-mode*), and an empty circle is a suspension of either *loss-of-follow-up* or *end-of-study*. Naturally, product births labeled as right arrows are spread in time, and the small squares represent the end of the respective warranty period. Both the black and the dashed circles come from a claim, and the chapter 3 is concerned on their differentiation. According to the censoring mechanisms listed above, the third one is present in unit three and seven. Unit 5 was still under warranty when the observation finished and no claim was reported, therefore it is clear that it became a suspension at that time. A more complicated situation occurs with unit four. The item has been in field for a long time after the warranty but no claim was realized by the day the observation period ended. The only assurance is that it did not fail in the warranty period, although it can have failed already in another date such as during the star icon in the plot. Based on the assumption that all failures occurred in the warranty period are reported, this works approaches these cases by labeling them as suspensions in the date that the warranty expires ($t_c = t_w$). In a more generalized definition, a non-returned unit j is labeled as suspension at time $t_{c_j} = \min(t_w, t_j)$, where t_w is the warranty period and t_j is the age in the end of observation.

Figure 7: Censoring mechanisms in field data

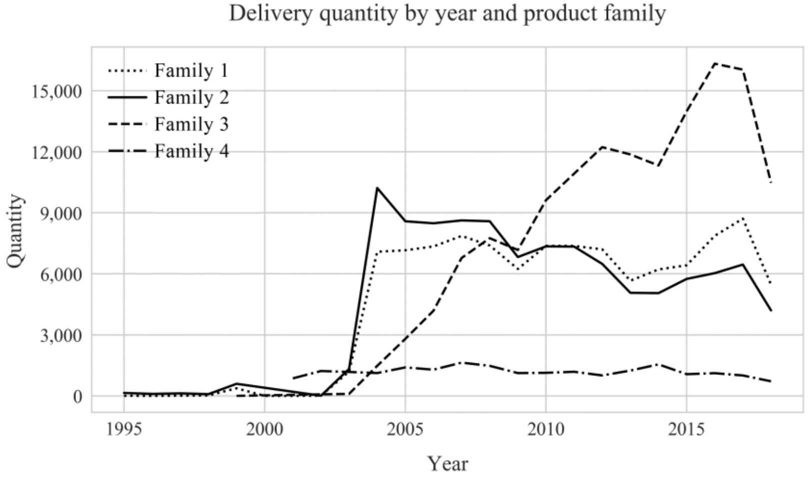


3.2 DATA SELECTION AND CLEANSING

This work will focus on the most active families of electric and electronic products from the company Bosch Rexroth. Following an organizational topology used in the company, an identification of a product can be done in three ways:

I. Product family: A family contains products that are related to each other in terms of application type and compatibility. The four families addressed by this work sum up over 450,000 repairs since 1995. Their distribution of sold units by year is shown in Figure 8:

Figure 8: Quantity of delivered products by year and family



II. Product type: A product family offers a few different products types. This work is limited to analyzing electrical motors, drivers, power supplies and controllers.

III. Material description: This is a long alphanumeric name as for example MDD112D-N-030-N2M-130GA2. This is hierarchically built, so that the left-most digits are more significative. Therefore, it is usually shortened to a few digits in the daily use. When limited to three digits for example, this code receives the name KT3, where KT stands for the German word “Kurz-Typ”, that means “short type”. The above described families and types have together 38 different KT3 codes and 454 different KT7 codes. Every KT7 is related to only one product family and type, but the same is not applicable to KT3s.

This work will take into account all worldwide deliveries and claims related to the products previously cited. The company holds a few different databases regarding the deliveries and claims with distinct information details. Some of them could not render good results for this project, since they omitted failure knowledge and precise dates. Finally, two databases (delivery and claim) split into four files due to space limitations were adopted. They are of type Access Database and are stored locally.

The utilized delivery database contains almost one million entries with the following most important fields:

I. Qty: It is the quantity of delivered units in a single shipment. Intuitively values greater than zero are expected, however negative entries are also present due to returns of orders. In this case, the negative quantities are supposed to balance out positive deliveries that were aborted.

II. Delivery Date: It is the date of shipment. This field contains only information regarding year and month.

III. Contract Customer ID: It is the identification number of the customer that realized the order.

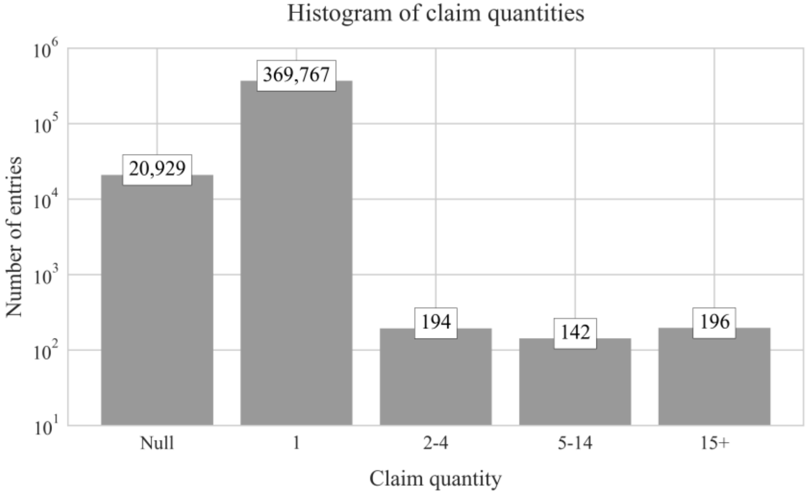
IV. Consignee Customer ID: It is the identification number of the customer that received the product.

V. Material Description: Hierarchical alphanumeric code describing the product.

The claim database contains almost 450,000 entries and their most important fields are following described:

I. Qty: It is the quantity of claimed units. Especially because each claim is assigned to a serial number that is a unique identification code, the quantity should be equal to one. However, its distribution shown in Figure 9 shows many blank and outlying values.

Figure 9: Histogram of quantities in repair entries



II. Delivery Date: It is the date of shipment. This field contains information regarding year, month and day.

III. Claim Date: It is the date that the unit arrived at the repair shop. This field contains information regarding year, month and day.

IV. Customer ID: It is the identification number of the customer that sent the product back.

V. Material Description: Hierarchical alphanumeric code describing the product.

VI. P_SORT_NR: It is a numeric field indicating the order of failure. A primary failure is represented by zero, which is a failure not caused either directly or indirectly by the failure of another item [122].

VII. CODE_CuCo: This categorical field serves as identification of the type of service realized.

VIII. ART_KURZTEXT, ARK_LANGTEXT and ART_K_L_TEXT2: These three fields are in form of sentences and are typed by the person responsible for the realized repair. They express the problem and the repair procedures in their rawest form. Therefore, it can be written in any language, although the majority is in German or English. These fields are known as the *failure description*.

IX. V_CODE: This categorical field is composed of at least one character describing the type of failure encountered in the service. Unfortunately, the company believes that this field is only properly filled in services executed in Germany and not abroad. The available characters and their meanings are listed in Table 5. Note that, since a service entry can employ more than one V_CODE at once, the sum of the frequency column exceeds 100%. In the analyzed database, 35 combinations of codes are present in total.

X. Warranty Code: This code indicates the decision of the company regarding accepting or rejecting the warranty request, and how the costs were split.

Table 5: Distribution and meaning of V_CODE

V_CODE	Frequency inside Germany	Frequency outside Germany	Definition
B	41,0%	44,2%	Product failure in field
O	29,7%	12,0%	Failure not found (NFF)
K	20,1%	15,4%	Human failure
S	3,5%	24,6%	Other failures (not categorized)
U	3,8%	9,9%	Remanufacture
F	2,1%	1,1%	Manufacturing failure
E	1,1%	2,1%	Development failure
L	1,7%	0,4%	Supplier failure
D	0,1%	0,0%	Damage due to transportation

Prior to analyzing the fields, the correct data type is assigned i.e. converting strings to date or integer depending on the case. If a necessary transformation cannot be realized due to inappropriate values, the field is treated as empty. Secondly, fields that are essential for the validity of the claim entry are identified. If an entry does not have all of them properly assigned, it cannot be interpreted and therefore can be simply removed. Every delivery must contain both a date and a material description. A claim, in the other hand, must also contain the claim date and the serial number.

The fields related to dates present some outliers outside of the working range, such as entries in the year 1900. These account for 1.59%

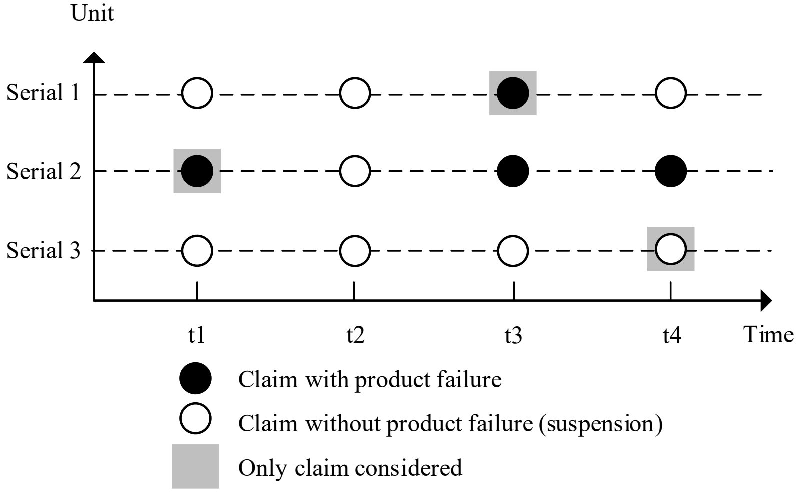
of the total and are removed. Moreover, some claims show the repair date before the delivery data. These are also removed.

Claims with quantity equal to zero are simply set to one; following the belief that since it is an obvious field some users might forget to fill it. If the quantity is greater than two, the entry is removed due to being too erroneous and consequently present suspicious information.

When an item is repaired, more than one type of failure can be present, and one claim entry is created to each. These duplicates can be identified by the field P_SORT_NR. Entries with values representing others but primary failures are removed from the data selection.

Naturally, the same product can return to the plant several times for different services and because of different problems. However, because each claim must be related to one delivery, and there is only one delivery entry for each item, only one claim must remain. The selection of duplicated claims to be removed is based on the serial number, which persist invariable over the time. Because the systems are treated as non-repairable in this work, only the first product failure is relevant, while the others can be ignored. However, if there exist more than one claim for one serial number and none of them represents a failure event, but a suspension, the logic is inverted. Suspension events only serve to define the last date in which a failure has not been observed. Thus, if all entries are suspensions, only the last one should be taken for granted. The method described for selecting the relevant claim is graphically demonstrated in Figure 10 with three examples of four claims each.

Figure 10: Method to solving duplicated claims


Unit
Serial 1
Serial 2
Serial 3
t1
t2
t3
t4
Time
● Claim with product failure
○ Claim without product failure (suspension)
▨ Only claim considered

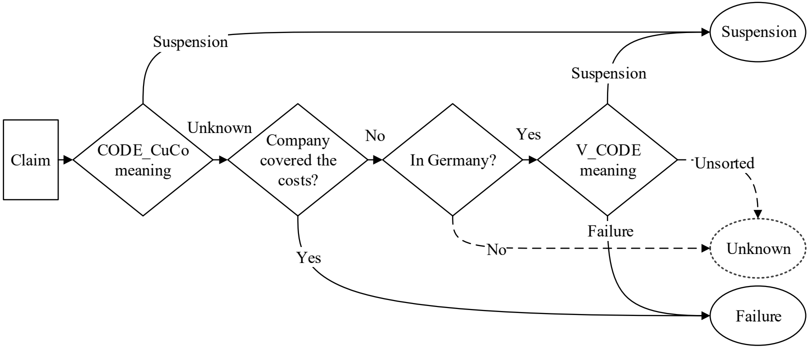
3.3 CLASSIFICATION OF EVENT TYPE

3.3.1 Architecture of the failure classifier

Many of the previously cited approaches and future estimations rely on the classification of claims regarding the event type. Each claim can represent either a failure or a suspension. These two values of event type can also be labeled as ‘F’ or 1 for failure and ‘S’ or 0 for suspension. While ‘F’ stands for a failure related to the product’s reliability, ‘S’ can arise from many situations such as human failure and claims that are not related to repair services. Discerning different types of suspensions is possible based on the field CODE_CuCo. These are mostly cases where a product returned to the factory for modification or upgrade to a different version. Nevertheless, if the service realized was a repair, more advanced methods are required to interpret the type of failure addressed by it.

As explained in the last chapter, certain values for the field of warranty code indicates that the company assumed the repair costs. This event implies that the failure was caused due to a product failure. However, if the company has not borne the costs, it is not possible to infer the nature of the failure. In order to support that, the field V_CODE exists mainly for the identification of the type of failure present in the claimed item. The code ‘B’ stands for product failure. If an entry does not contain ‘B’, it can be considered as suspension by *loss-to-different-failure-mode*. Unfortunately, there is an assumption that this field can only be entirely believed in repairs realized in German plants, because other countries’ personal may not know the proper meaning of the codes. Figure 11 presents a flowchart of the classification of a claim given the presented fields of information.

Figure 11: Flowchart of claim classification through available fields



The dashed lines in Figure 11 leads the classification to an unknown state, since the available information is not enough to identify the type of event occurred in the claim. In order to overcome this critical deficiency, a binary classifier based on a machine learning model is proposed. This classifier is designed so that it is capable of understanding natural text in order to enhance the overall classification accuracy. Besides all fields previously cited, the three failure description fields i.e. ART_KURZTEXT, ARK_LANGTEXT and ART_K_L_TEXT2 are utilized. The working principle of the classifier is based on the following assumptions:

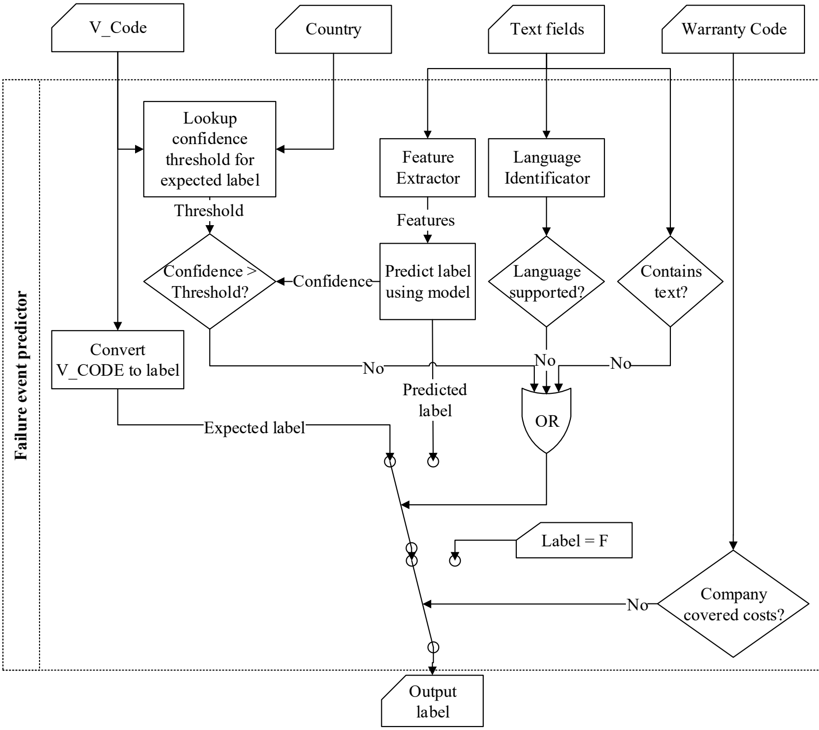
I. Every claim has information regarding repairing country and V_CODE.

II. The information regarding the repairing country, warranty code and failure description is always trustful, although the last two are not necessarily always available.

III. The reliability of the field V_CODE is dependent on its value and the repairing country. Germany is the only country where it is entirely trusted.

A logistic regression model as described in chapter 2.2.2 is the core algorithm of the failure classifier, which itself incorporates other interfaces and perceptions. The architecture of the failure classifier is presented in Figure 12.

Figure 12: Structure of the failure classifier



The function for looking up the confidence threshold respects a table of values given by the user. An example is provided by Table 6. If a given threshold is 75%, for example, $|1 - y| > 0.75$, where y is the output predicted by the logistic regression model that varies from zero to one.

Table 6: Table for threshold confidences

V_CODE	Threshold inside Germany	Threshold outside Germany
B	100%	85%
O	100%	85%
K	100%	75%
S	60%	60%
U	100%	75%
F	100%	75%
E	100%	75%

L	100%	75%
D	100%	75%

3.3.2 Feature extraction

At the core of the classifier, a logistic regression model predicts the probability of the failure description fields addressed to a product failure or not. In order to feed the input data to the model properly, features need to be extracted from the texts. This process involves natural language processing (NLP) practices, which is the field concerned with the ability of a computer to understand and analyze human language.

A feature extractor module is implemented following the state-of-the-art techniques described in chapter 2.2.2. This module is composed of three phases: cleaning, n-gramming and vectorization. The cleaning phase aims to normalize the text and remove prediction noises. It is composed of the following steps:

I. Transform all characters to lower case. This avoids duplicated features with the same interpretation, which would imply different prediction effects depending on the character case.

II. Convert the punctuation and symbols defined in “#!"$%&\()*+,-./:;<=>?@[\\]_{|}~” to white spaces.

III. Combine multiple white spaces into one only. Since the tokenization is realized based on white space separation, this avoids creating tokens containing spaces only.

IV. Remove dates assuming that they do not add any relevant information.

V. Remove quantities and dimensions such as “5x454” or “10x4”.

VI. Remove material identification numbers such as r911321083.

The last four steps are executed using regular expressions, which are sequences of characters that describe a search pattern. The Perl syntax is used to create the patterns listed in Table 7.

Table 7: Regular expressions based on the Perl syntax

Step	**Regular expression**
Multiple spaces	“ +” → “ ”
Dates	“\d+\d[./-]\d\d[./-]\d\d+” → “”
Material dimensions and quantity	“\d+\s*x\s*\d+” → “”
Material number identification	“r\s*\d\d\d\d\d\d\d\d\d” → “”

The second phase, called n-gramming, transforms the cleaned text into a list of n-grams following the steps chronologically:

I. Tokenize: Words (tokens) in a text are individually identified based on the white space separation, and the array of tokens is returned. Tokens too big are removed since they are normally the result of words combined without white spaces or also not meaningful codes.
II. Remove stop words that do not enhance predictability. These are mostly articles, pronouns and prepositions [55].
III. Stemming words to reduce them to their radical form: This is a normalization technique that aims to avoid feature redundancies. Lemmatization is not employed due to performance requirements.
IV. A multiple n-gram operation that extracts every contiguous sets of words varying from size one to three.

The first two phases are applied to each text field individually. If they were to be concatenated into one text only prior to the analysis, the n-gram operation would join the first and last words of each field together, which it is undesired. The output of the n-gramming phase is a list of n-grams identified in each text field. N-grams identified in the training set are stored in a dictionary for posterior use. The final phase proceeds as following:

I. Merge the lists of n-grams into one only. Duplicates are maintained.
II. The TF-IDF feature weighting method is employed to calculate the pertinency (numeric measure) assigned to each feature of the dictionary.

The three phases together transform the three text fields into a numeric feature vector as shown graphically in Figure 13.

Figure 13: Feature extraction process

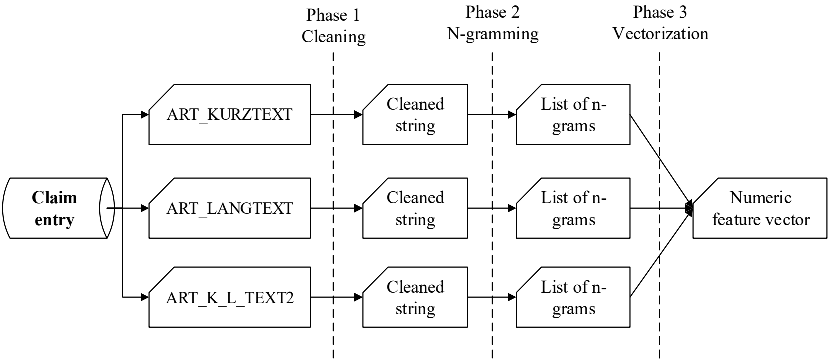
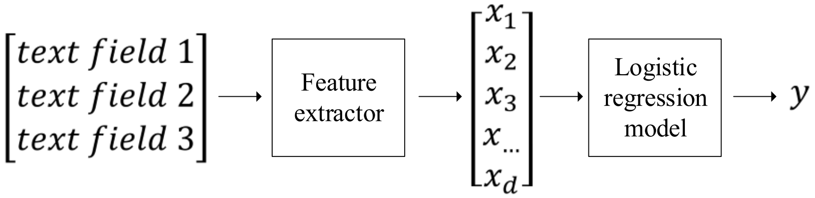


Figure 14 describes the interfaces of the feature extractor and the predictor, where the array $\boldsymbol{x} \in \mathbb{R}^d$ is the array of $d = 40{,}000$ features (n-grams) and $0 < y < 1$ is the prediction output. The closer y is to one, the higher is the probability that the three text fields are extracted from a claim addressing a product failure.

Figure 14: Interface of the logistic regression model

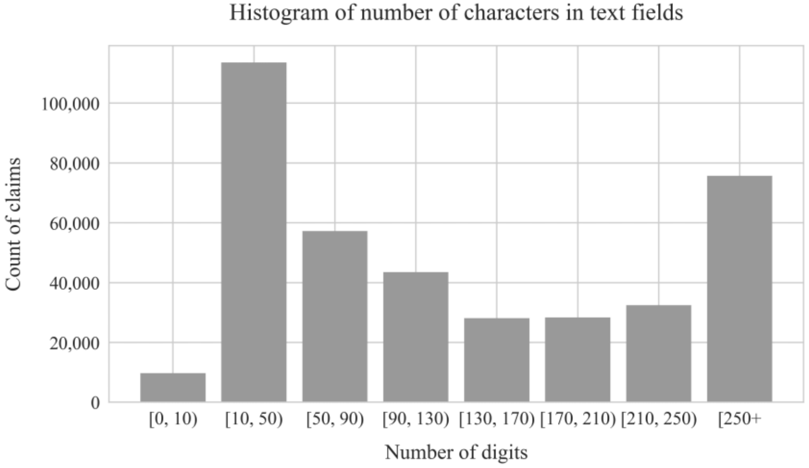


3.3.3 Training data for the logistic regression model

The learning method of the logistic regression model used is supervised, which means that it needs to be trained with examples containing both input and labels prior to use. This subchapter addresses the attainment of the training set.

Based on the assumptions explained in chapter 3.3.1, the repairs realized in Germany are used for the training phase. The model's input are the three text fields while the label is generated from the field V_CODE. Initially, all repairs realized in Germany that have non-empty failure description are imported. Figure 15 shows the distribution of claims by the sum of the lengths of the text fields.

Figure 15: Histogram of count of characters in text fields



After importing the data, the following manipulations are executed in order to obtain the training data set:

I. **Filter language**: The text language present in the training set should match future uses since it is based on the same dictionary of features. The English language is chosen for this purpose. It accounts 39% and 93% of the claims inside and outside Germany respectively.

II. **Generate label**: The only label used in the predictor indicates whether the claim is a product failure. The only V_CODE that indicates that is the code “B”, and therefore is assigned to label = 1. If the V_CODE does not contain the digit “B”, it is assigned to label = 0. The code “S” is left for unsorted failures and therefore respective rows are removed from the data set.

III. **Remove duplicated**: Duplicated examples do not add extra information for the training process, therefore the duplicates can be removed. In 0.99% of the cases, duplicated inputs are found with two distinct labels. These situations are undesirable and both cases are removed.

The last step before using the prepared data to train the model is to balance it. Unbalanced sets tend to induce a bias in the prediction due to the different amount of accumulated errors for different labels. The balancing is done by resampling with replacement the smallest set in the size of the largest set. In the end, equal quantities of training examples with each label are stored in a dataset with 117,162 examples.

3.3.4 Performance of the logistic regression model

The available dataset is shuffled and split in two parts: 80% become the training data, used to fit the model, and 20% the validation data, used to evaluate the performance of the model. The confusion matrix in Table 8 is used to visualize the number of false and true positives and negatives. For example, this model incorrectly classified as product failures 1,334 out of 11,475 claims.

Table 8: Confusion matrix of model's prediction

	Predicted = 0	**Predicted = 1**
Actual = 0	10,371	1,334
Actual = 1	1,587	10,141

The model presents an accuracy of 87.5%, which is considered very good for such a simple model as the logistic regression. This means that it assigned the correct labels to 87.5% of the examples.

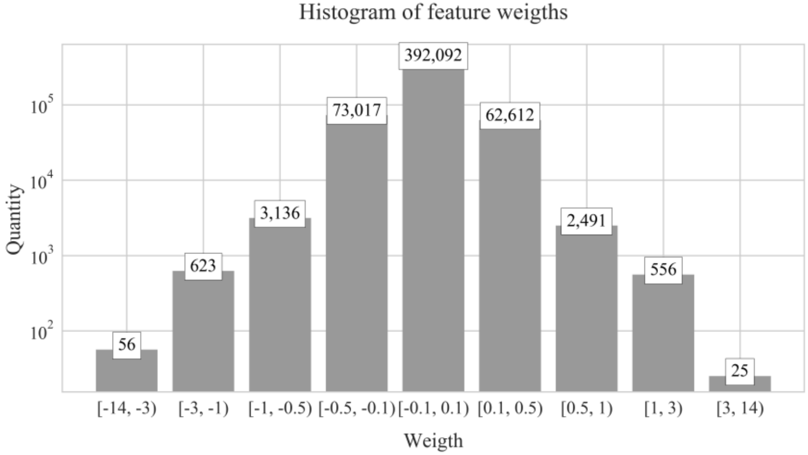
Precision is the ratio of correctly predicted positive observations (label = 1) to the total predicted positive observations, while recall is the fraction of correctly predicted positive observations over all positive observations. These measures achieved scores of 88.4% and 86.5% respectively.

After the training is completed, a weight is assigned to every feature of the dictionary. The most significant features hold the largest weights. These are listed in Table 9. Features with positive weights are identified as favorable to the occurrence of a product failure, and negative to the opposite. Notice that features are not necessarily made of complete words, since the stemming operation reduces them to a more general form.

Table 9: Most significant features

Feature	Weight	Feature	Weight
faulti	10,32	custom	-12,37
defect	9,37	no fault	-8,59
defekt	9,22	no error	-7,88
board	5,58	without	-7,61
electr faulti	5,58	without fault	-6,98
electr	5,45	contamin	-6,87
without function	5,42	mechan	-5,99
def	4,83	without error	-5,95
short	4,45	oili	-5,31
sensor	4,25	solder	-5,24
electr fault	4,12	pollut	-5,22

Figure 16: Histogram of weights across features



As shown in the histogram of Figure 16, the majority of features do not present a significative weight and therefore do not add much to the prediction. Due to performance optimization, a selection of the most relevant ones is realized. Out of the 534,604 different n-grams, the 40,000 most important are selected and the rest is removed. The feature's importance is proportional to the absolute value of its weight. This provokes a reduction of computer requirements while maintaining the prediction performance. Table 10 compares the performance of both alternatives running on an Intel i7-4810MQ 2.8GHz.

Table 10: Performance comparison by the number of features

	Using all features	**Using the 40,000 most significant features**
Time for fitting logistic the regression model	17.2s	9.2s
Time for predicting 100 text inputs containing 20 words	467ms ± 14ms (mean ± σ of 7 runs)	428ms ± 48ms (mean ± σ of 7 runs)
Space required in disk for storing classifier	40.1 MB	14.9 MB
Accuracy	87.5%	86.2%
Precision	88.4%	86.5%
Recall	86.5%	85.8%

The confusion matrix for the model based on limited features is presented in Table 11. It assembles very much the previous matrix of Table 8, showing that eliminating features with small weights do not affect the prediction performance significantly.

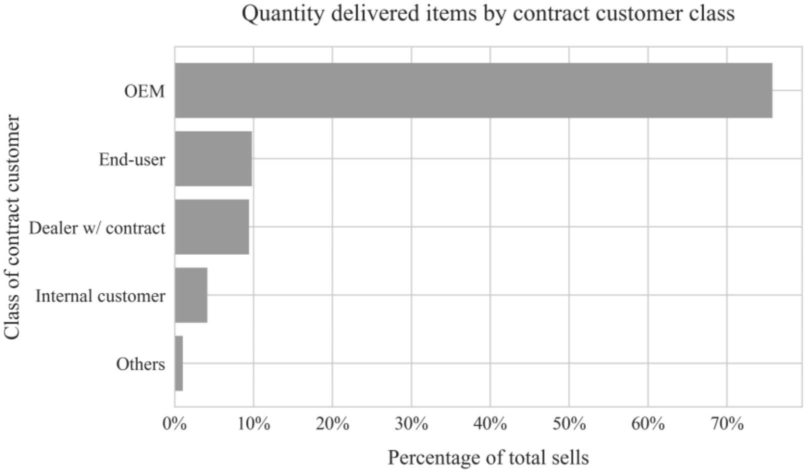
Table 11: Confusion matrix of model's prediction after limiting features

	Predicted = 0	**Predicted = 1**
Actual = 0	10,133	1,572
Actual = 1	1,660	10,068

3.4 CORRECTION OF INSTALLED BASE

Both the number of working units and failure occurrences are necessary to evaluate reliability indices. Therefore, if the calculation is realized in a single country, it is essential to know how many of its units are installed and repaired. Based on the delivery database, it is possible to know where the customers are located and to which classes they belong. The plot in Figure 17 shows the distribution of sold units to different classes of customers.

Figure 17: Quantity of items sold by contract customer class



A significant fraction of products is sent to OEMs (Original Equipment Manufacturer), which are specialized in building machines to further exporting to other clients that can be in the same country or not. In order to consider this, a research was conducted to develop a database containing export rates for machinery equipment of any given country to any destination during each year since 1990. Using data of 162 countries for 29 years, this table sums up 761,076 combinations of export rates. The data was obtained and cross-validated using the online portals Mechanical Engineering Industry Association (VDMA) and the World Integrated Trade Solution (WITS), which has as partners The World Bank, World Trade Organization (WTO) and the International Trade Centre (ITC) and the United Nations (UN).

Based on the assumption that OEMs customers share the same exporting profile as their country's average (for the same application branch), this database was used to emulate further deliveries and move fractions of installed base across the globe. The algorithm follows the steps of Figure 43 located in APPENDIX B – Functional Diagrams.

The location of the defective item is also essential for the correlation with the deliveries. In order to be as effective as possible, the approach tries firstly to intercept the country of the customer that sent the item. However, since it relies in a field that is not always present, the algorithm uses as last resource the country of the plant that realized the repair.

4 RELIABILITY MODELLING

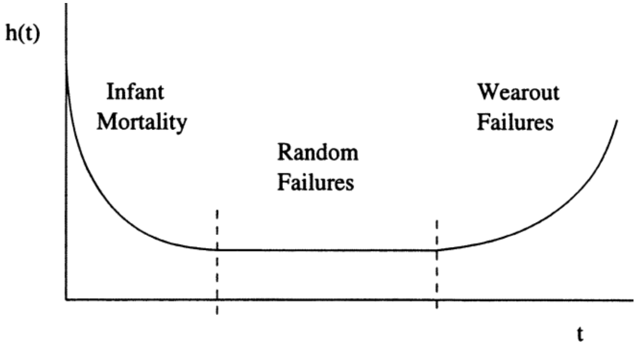
4.1 PREPARATION OF DATA

4.1.1 Assuring homogeneity

The modelling of a reliability function based on data must approach individual or similar failure modes [15]. If a population of items satisfy this, it is called a homogeneous population. Appropriately, the delivered and claimed items are clustered with regard to the similarity of failure characteristics. Intuitively, items within a cluster must be more similar to each other than they are to another belonging to a different cluster [123]. The measure used to quantify this is the material description. Different depths (or number of digits from left to right) divide the products in different detailed levels. If a low depth is chosen, the models can become too general and not describe correctly specific products. If too much detail is included, the clusters can become too small and not enough data are available for the modelling. In total, almost one thousand different configurations of devices exist. The first three digits are the first level of differentiation and are useful, for example, to differentiate an electrical motor from its controller. When seven digits are included, different sizes and types of motors are identifiable. Further digits are only used to differentiate small details such as encoder output, presence of brakes or communication protocol. Expert knowledge of the repair team helped to establish the differentiation factor as seven digits. This means that products with material descriptions that share the first seven digits (or have the same KT7) are clustered together for the estimation of reliability models. This process is also called stratification and the groups are also called batches [26].

When new products are presented to the market, they can present a higher failure rate in the beginning of the life. This behavior is also called infant mortality [124] and is known as the first phase of the bathtub curve shown in Figure 18.

Figure 18: Bathtub curve for the hazard rate over time

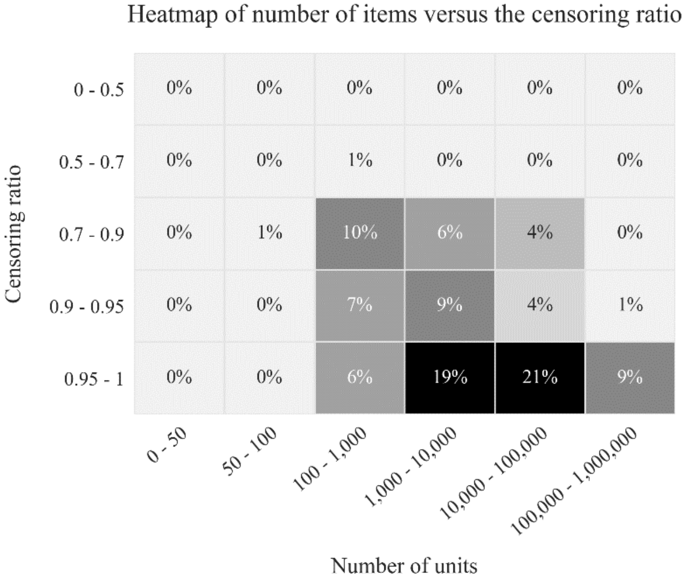


Source: [124]

Rapidly after the release of products with higher than expected infant mortality, the problematic aspects are normally identified and the subsequent produced versions of the same product are corrected. After the corrections, the product is said to become stable. In order to avoid this influence than can disturb the homogeneity of the data, the beginning of the life in the market is ignored and only products delivered after a “stable delay” are taken into account in the estimation of the reliability.

When splitting the data covered by this work respecting the homogeneity aspects discussed in this subchapter, two hundred datasets are identifiable. Their main properties are shown in Figure 19. This heatmap is a three-dimension visualization of the number of datasets distributed according to the censoring ratio, which is the quantity of suspensions divided by the quantity of units, and the total quantity of units.

Figure 19: Heatmap of all event tables



4.1.2 Creation of the event table

Field failure data are distributed in both delivery and claim tables and need to be transformed to event (or TTF) table in order to run through estimation algorithms. An example of an event table for censored data is shown in Table 12. It is composed of three columns: The time of occurrence of the event, the quantity of units, and the event type e, which can be F for failure or S for suspension. Sometimes, event type can also be defined as a Boolean or numerically, such as $F = 1$ and $S = 0$. If data is not censored, the type of event is not necessary and all times are failure times (TTF).

Table 12: Example of an event table

Time	Event type	Quantity
2	F	20
4	F	30
6	F	25
6	S	10

In this phase, the delivery dataset must contain the quantity delivered in each date, and the claim dataset the delivery and claim dates, as well as the quantity and the event type. Both are related to a homogeneous population.

The first step in the creation of the event table is to correct the dates by compensating the accounted delays. These are defined as two types:

I. **Commissioning delay**: It accounts for the entire time window between the delivery of a product and its operation start.

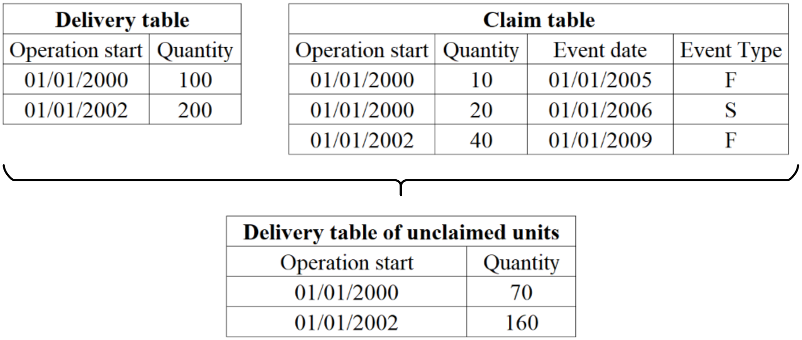
II. **Claim delay**: This delay is usually short and is the interval between the occurrence of a failure and the arrival of the defective product at the repairing company.

The commissioning delay is added to the delivery dates of both delivery and claim datasets, while the claim delay is subtracted from the claim dates of the claim dataset. In order to have clearer notations, these operations are assigned to new variable names, as described in the given equations:

$$\begin{aligned} start\ of\ operation &= delivery\ date + commissioning\ delay \\ event\ date &= claim\ date - claim\ delay \end{aligned} \tag{4.1}$$

The next step is to identify all deliveries that were not claimed to set them as censored units. This is done by matching units in both datasets with regard to the quantity put into operation in each date. In other words, the delivery dataset should keep only unmatched units (that were never claimed). An example of this operation is given in Figure 20.

Figure 20: Matching deliveries and claims to find unclaimed deliveries



Delivery table	
Operation start	Quantity
01/01/2000	100
01/01/2002	200

Claim table			
Operation start	Quantity	Event date	Event Type
01/01/2000	10	01/01/2005	F
01/01/2000	20	01/01/2006	S
01/01/2002	40	01/01/2009	F

Delivery table of unclaimed units	
Operation start	Quantity
01/01/2000	70
01/01/2002	160

All unclaimed deliveries require an event type and date. Since none of them has a failure date defined, they are all considered suspensions. The event date is then set with regard to the assumptions presented in chapter 3: If a unit has not been claimed yet, the only valid certainty is that it survived the warranty period. Therefore, an unclaimed unit is labeled as suspension at $event\ date = \min(delivery\ date + warranty\ length;\ end\ of\ observation)$.

The reliability analysis is concerned with the life expectancy of each product, but the start of operation date is irrelevant. Therefore, a new field substituting the dates is inserted as $event\ time = event\ date - operation\ start$. At this point, both tables are ready to be concatenated. Finally, the quantity is grouped by event time and type producing the final event table. The steps listed here are more extensively illustrated in APPENDIX B – Functional Diagrams.

4.1.3 Median Rank calculation

Traditionally, at this phase the reliability engineer would have to choose between using the approaches of rank regression or the maximum likelihood estimator (MLE) in order to obtain parameters of a pre-defined theoretical distribution, such as the Weibull. This work uses the first rather than MLE mainly because of three reasons:

I. Solutions given by the MLE are extremely biased at an unknown amount for small sample sizes. [15] suggests its use only when more than five hundred samples are available. Unfortunately, when stratifying the products into clusters of similar material descriptions, they do not always have so many observations, and for simplicity, a common method shared by all estimations is preferred.

II. When ranks (empirical reliability values) are generated, several approximation methods can be further applied. The state-of-the-art is to apply logarithmic transformations to the axes and apply the linear regression to fit parametric distributions. This work takes advantage of the available ranks to also implement and compare the performance of neural network regressions.

III. Since MLE solutions are not based on rank positions, different performance measures are applicable, making the comparison to other approaches difficult.

Today's state-of-the-art for the calculation of empirical reliability indices supports two approaches: median ranks (MR) and Kaplan-Mayer estimates (KM). The second is broadly used by the medial industry but is avoided in this work for two undesired characteristics:

I. When subjected to some censoring patterns, the KM estimates the last failure time as $\hat{F} = 1$, indicating the it is impossible for a unit to survived after it [107]. This do not work along with the idea of generalization of reliability functions outside of the limited observation window.

II. Kaplan-Meyer normally underestimates the true reliability values, which can cause undesired effects in further uses of the results of this study [41].

As extensively presented in chapter 2.3.1, the literature holds many different options for estimating median ranks. This work utilizes the Filliben's [93] estimator defined in equation (2.11), since it is known to produce the smallest deviation from the real reliability values [89]. Since all datasets have censored observations, Johnson's method for adjusting the median rank is also adopted [86].

4.2 ESTIMATION OF RELIABILITY MODELS

4.2.1 Parametric distribution

The modelling of reliability functions based on failure data is commonly executed with parametrical approaches, such as fitting a theoretical distribution to the median rank positions. These distributions are also called life data distributions, due to their effectiveness in modelling such data [39]. One of the most versatile distributions is the Weibull, in a sense that it can fit many different failure behaviors such as infant, random or wear-out phases. Many authors cite it as the most important and useful distribution for modelling reliability [15]. For these reasons, while aiming to adopt a common approach to be used when dealing with different products, an estimation algorithm is developed to fit any batch of failure data into a Weibull distribution.

The estimation algorithm is implemented using rank regression on Y (RRY), since it is recommended method in the literature [39]. It is based on the steps following listed.

I. Apply $\ln(t)$ transformation to the median rank times and $\ln\left(\ln\left(\frac{1}{1-\hat{F}(t)}\right)\right)$ transformation to the median rank values.

II. Plot transformed median rank and use least squares regression to find the best fitting line through the points.

III. Convert LS estimates to Weibull distribution parameters.

The quality of the fit is measured by the Pearson correlation coefficient r, defined in (2.24), which quantifies strength of the linear relationship between two variables [125]. Another useful measure is the mean squared prediction error (MSE), which calculates the average of the sum of squared distance between the median rank values and the predictor's predictions. It is defined by (4.2), where MR_j is the jth median rank and N_{MR} is the number of existing median rank values.

$$MSE = \frac{1}{N_{MR}} \sum_{j=1}^{N_{MR}} \left(MR_j - \widehat{MR}_j\right)^2 \qquad (4.2)$$

4.2.2 ANN for modeling empirical reliability functions

Artificial intelligence is famous for learning and estimating complex relationships on its own and depriving the developer from depths of field expertise at some degree. Artificial neural networks are very powerful in this sense; however, they demand knowledge and work in order to choose the best topology and hyperparameters tuning. A bad topology can imply in too much unnecessary calculation power and time, as well as under- or over-fitting. The process of choosing the right hyper-parameters for this problem is presented in this subchapter.

The data used as input and output has only one dimension each. The first is the event time given in months and the second is the corresponding empirical unreliability index $\hat{F}$ calculated by the median rank. Due to the simplicity of this problem, convolutional or recurrent neural networks are not considered. Instead, a feed-forward ANN (also called multilayer perceptron or MLP for the cases with hidden layers) which has proven to be sufficient is used [18]. For this model and type of data, the structure of the neural network must be given by one input and one output node, with one or more hidden layers of one or more hidden units.

The first claim events usually occur in the first months for any given product. However, depending on how much time it has been on the market, the distribution of event times can vary a lot. Figure 21 shows that the last event times observed in the median rank tables range from 5 to 220 months. Equivalently, Figure 22 displays that some products have range (difference between last and first) of event times 73 times wider than others.

Figure 21: Histogram of last event times in datasets

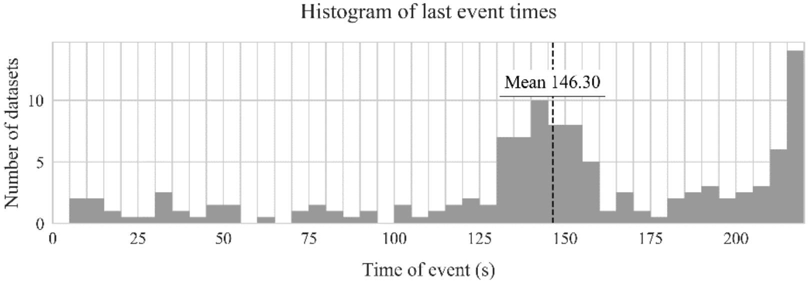
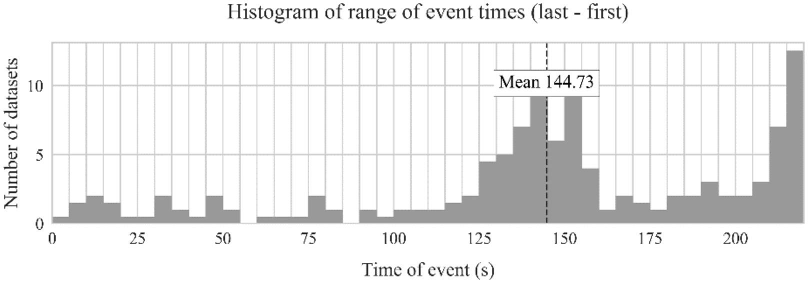


Figure 22: Histogram of range of event times in datasets



[79] show that the advantages of standardizing the input are especially noticeable in smaller networks. Since this is the case and the event times are provided in a big variety of ranges, the inputs given in months will be normalized before being fed to the ANN. Since the labels used to train the network are CDF values, they are always contained in the range from zero to one. For this reason, it was initially found reasonable to use the sigmoid function as the activation function for the output node. Nonetheless, the model encounters much difficult in the modelling of rank tables in which the range of label values are too close to each other, or too close to the limits 0 and 1. Visually demonstrated in the histogram of Figure 23, 74.9% of the tables have initial rank values below 0.002, which can be harmful to the training process. Furthermore, as shown in Figure 24, the difference between the largest and the smallest MR values vary significantly depending on the dataset. While some ranges are as low as 0.00051, others utilize the whole spectrum reaching 0.998 difference. In order to suppress this issue, the neural network is fitted using scaled labels values. The scaling method chosen is the linear interpolation between 0.1 and 0.9.

Figure 23: Histogram of smallest median rank values in datasets

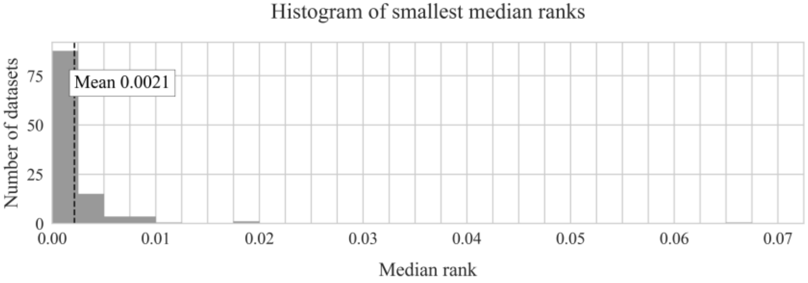
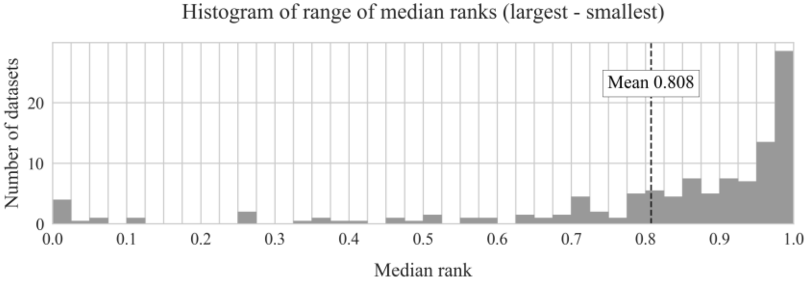


Figure 24: Histogram of range of median rank values in datasets



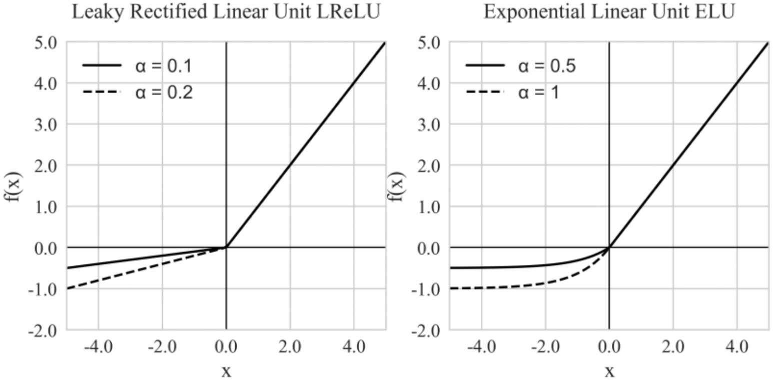
Initially, the state-of-the-art rectified linear units (ReLU) were being used as activation functions of the hidden nodes. However, the network presented performance issues due to dying neurons. This means that neurons become stuck and no gradients can flow backward during training [126]. This can happen because of the large domain region of the ReLU where all outputs and derivatives are null. Many solutions to this behavior are presented in the literature [127] by adapting the ReLU function such as:

$$Leaky\ ReLU = \begin{cases} \alpha x & if\ x \leq 0 \\ x & if\ x > 0 \end{cases} \tag{4.3}$$

$$Exponential\ Linear\ Unit\ (ELU) = \begin{cases} \alpha(\exp(x) - 1) & if\ x \leq 0 \\ x & if\ x > 0 \end{cases} \tag{4.4}$$

These variants add a slight slope in the negative range of values in order to have a non-zero gradient over its entire domain, thereby preventing that the neuron get stuck in this region [128]. The plots of Leaky ReLU and ELU are included in Figure 25. A comparison conducted by [74] showed that the exponential linear unit (ELU) is a good alternative to the ReLU in regard to accuracy and speed, and therefore is adopted.

Figure 25: Plot of Leaky ReLU and ELU



Following the current state-of-the-art guidelines, the Xavier Glorot normalized initializer is used in the initialization of the weights of the network, and the adaptive moment estimation (Adam) is set as the optimizer. The cost function adopted is the mean squared error (MSE). The optimization algorithm stops if the maximum number of epochs is reached. Another feature utilized by the algorithm that helps the optimization is that the training data is shuffled between epochs, helping to avoid getting stuck in suboptimal solutions.

Of course, further hyperparameters still need to be configured prior to the final use of the model. There exist several methods for the optimization of hyperparameters, and the grid search is utilized. This approach is based on an exhaustive testing of a collection of predefined hyperparameters. For the evaluation of the best possible combination, the algorithm is guided by performance metrics. In this case, it is used the minimum mean squared error of predictions over the training dataset and the time necessary to achieve it. The pseudo-code of the prepared grid search algorithm is presented in Figure 26.

Figure 26: Algorithm for grid search of optimum hyperparameters of ANN

```
→ For each set of hyperparameters in collection of sets of
  hyperparameters:
    → Build neural network layout with set of hyperparameters
    → For dataset in collection of datasets:
        → Train neural network with dataset
        → Evaluate and store neural network performance (MSE
          score and time necessary to achieve it)
```

In order to exhibit validity in a bigger picture, each configuration of hyperparameters is tested on a variety of data samples that is representative to the spectrum of event tables encountered in this work's problem. For this purpose, twenty examples are randomly extracted without replacement from the available collection demonstrated in Figure 19. The heatmap of extracted samples is shown in Figure 27.

Figure 27: Heatmap of selected event tables

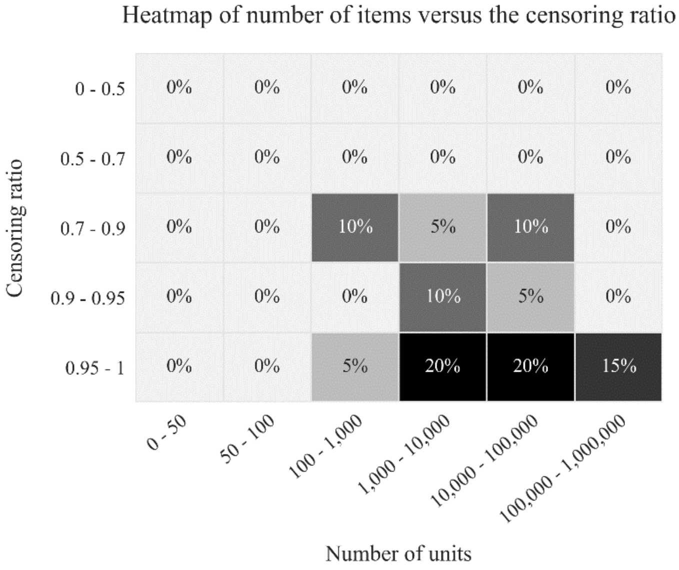


Table 13 contains all the relevant hyperparameters used in the battery of tests. Some of them are fixed to one value only, while others can take up to three different values. A total of 243 different combinations generated are used to fit the twenty sampled datasets.

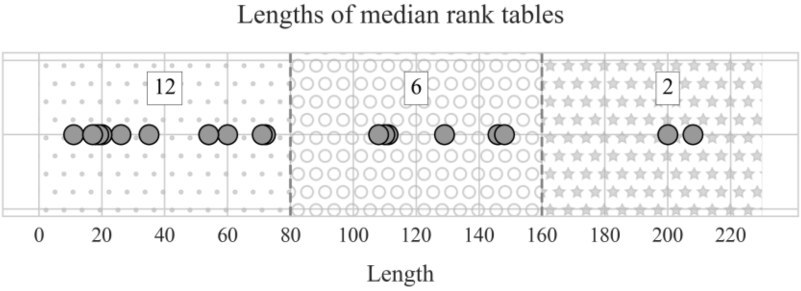
Table 13: Collection of fixed and configurable hyperparameters

Hyperparameter	**Values**	**Count of values**
Size of input and output layers	1	1
Number of hidden layers	{1, 2, 3}	3
Size of hidden layers	{5, 10, 15}	3
Input normalization	{None, Linear, Statistical}	3
Initializer	Glorot uniform	1
Output layer activation function	Sigmoid	1
Hidden layer activation function	ELU	1
Cost function	MSE	1
Optimizer	Adam	1
Learning rate	{0.02, 0.01, 0.005}	3
Batch size	{1, 5, 10}	3
Epochs	400	1
Number of combinations		**243**

Seven processing threads of an Intel i7-4810MQ 2.8GHz were used in parallel to fit all 243 different neural network architectures with 20 datasets in approximately 53 hours.

The comparison of the hyperparameter configurations is made dependently of the length of the median rank (MR) tables, which impacts on the number of training examples per dataset. Therefore, as shown by the shade patterns in Figure 28, the runs are divided into three categories: lengths from 0 to 79, from 80 to 159, and from 160 to 250. These contain 12, 6 and 2 datasets respectively.

Figure 28: Lengths of median rank tables



The most important performance measures in this work is the prediction error and the time necessary to train a network, since big amount of data is involved. The former is defined by the mean squared error (MSE) of all points in the MR table. The different configurations of hyperparameters provided extremely diverse performance results, as shown by Figure 29 and Figure 30.

Figure 29: Boxplot of training times in the grid search for hyperparameter

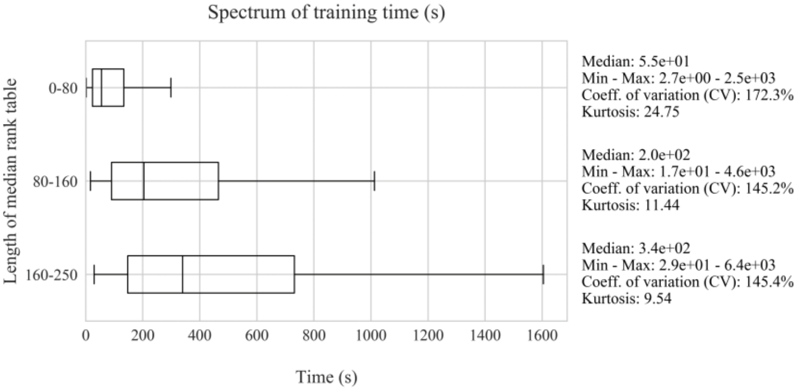
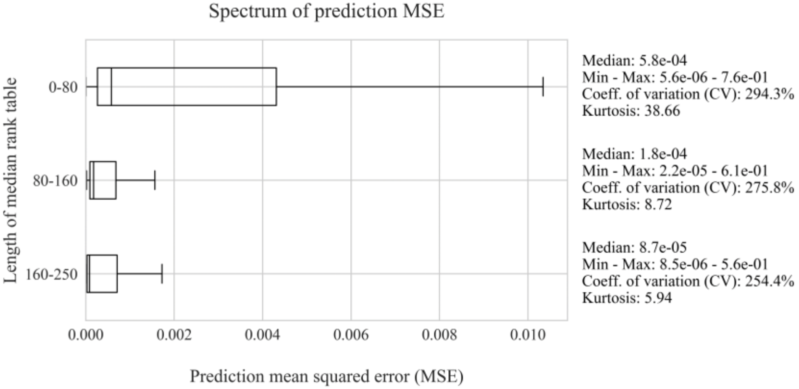


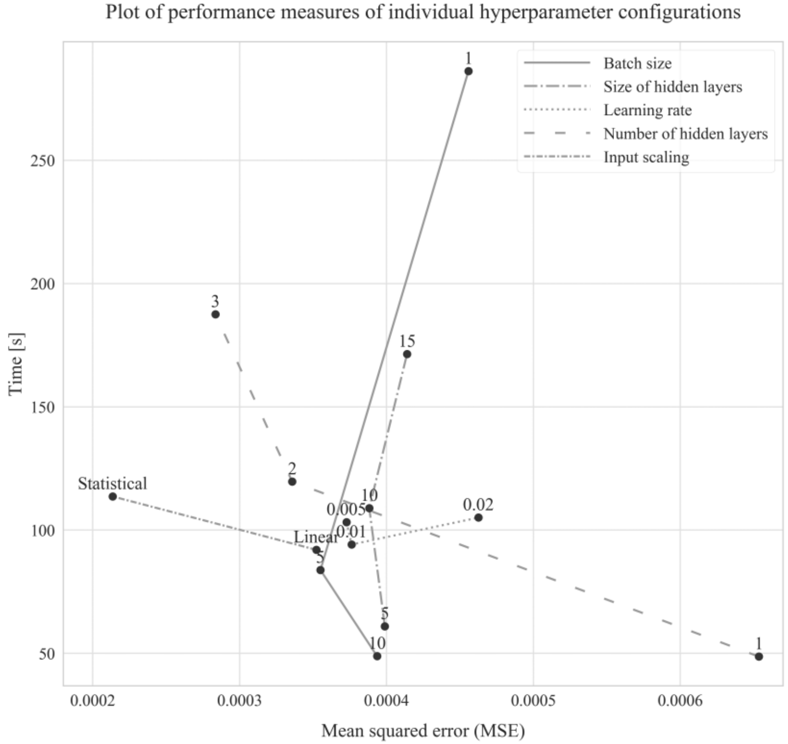
Figure 30: Boxplot of prediction MSE in the grid search for hyperparameter



The training time is substantially reduced in smaller datasets, with a median value of 55 seconds. The group of medium and large datasets presented medians of 200s and 340s respectively. The achieved mean squared error, however, is much lower in larger datasets. The median of the MSE scores of the largest group is 0.000087, which represents an 85% improvement over the smallest group. The group of smaller datasets produced a bit more disperse MSE and training time results due to the larger coefficient of variation.

In order to provide a general glance of the performance achieved in each configuration individually, Figure 31 is presented. The measures were achieved by calculating the median values of all fittings realized with the particular hyperparameter configuration. Consisting of a time and an MSE axis, the best performance results are closer to the origin of the graph. Since the runs without input scaling provided immense MSE results, they were ignored in this plot.

Figure 31: Plot of performance of individual hyperparameter configurations



It is believed that some variables are correlated, which means that the performance of one of them is dependent on the others value. In order to account these situations, some hyperparameters are clustered in pairs in order to be compared against each other. Their performance measurements are presented in heatmaps, which can easily display three-dimension information. In order to enhance readability, all values of a heatmap are divided by their minimum, so that every number is easily comparable to a base value of one. Again, the smaller the value, the better it is.

The main influences in the architecture of the network are the number and size of the hidden layers. Bigger and more numerous hidden layers increase the time consumption proportionally, as seen in Figure 32. The network's best accuracy, shown in Figure 33, is likely to be found in a sweet spot with two or three hidden layers of size 10. Due to speed constraints, the topology with two hidden layers of size 10 is chosen for this work's implementation.

Figure 32: Training time versus number and size of hidden layers

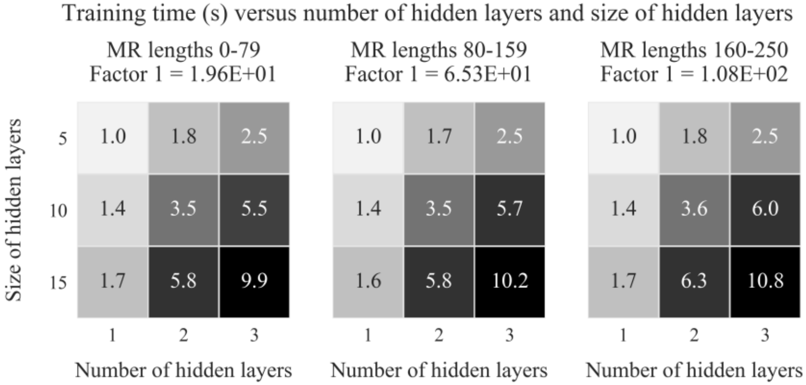
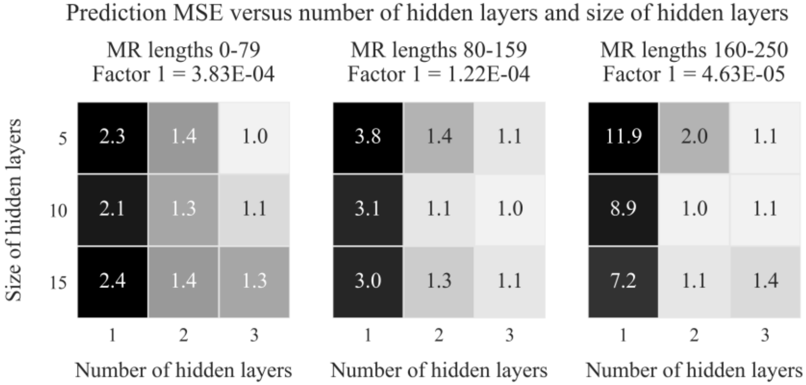


Figure 33: Prediction MSE versus number and size of hidden layers



The network's optimization algorithm is directly affected by the learning rate and the batch size. The first one is commonly found in related works around 0.01 when using the Adam optimizer. Figure 34 shows that this value achieved the shortest training times independently of the batch size, while still maintaining high accuracy.

As explained in chapter 2.2.3, if a batch size of one is chosen, the optimization algorithm is said to be based on the stochastic gradient descent (SGD), and the weights are updated iteratively after each training example is propagated forwards. When the batch has size five or ten, a mini-batch gradient descent (MBGD) is employed and the backward phase for updating the weights is executed only after the batch is processed.

Figure 34 shows that the training time is inversely proportional to the batch size. Based on Figure 35, it is visible that the network struggles to reach reasonable accuracy when the smallest batch size is used with a learning rate of 0.02. Batch sizes of 5 and 10 are relatively similar, and therefore the bigger is chosen due to the time sparing.

Figure 34: Training time versus learning rate and batch size

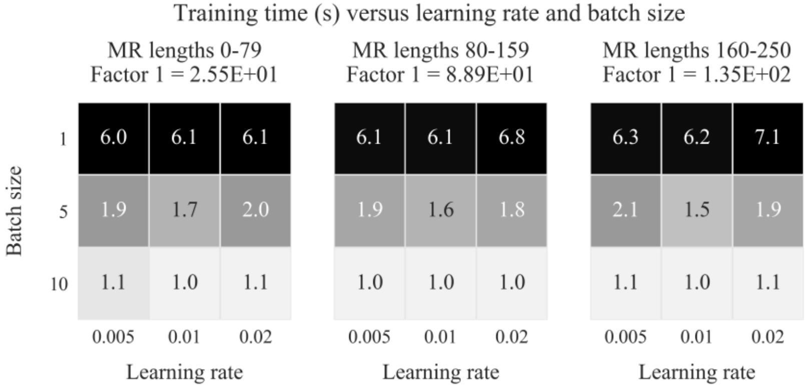
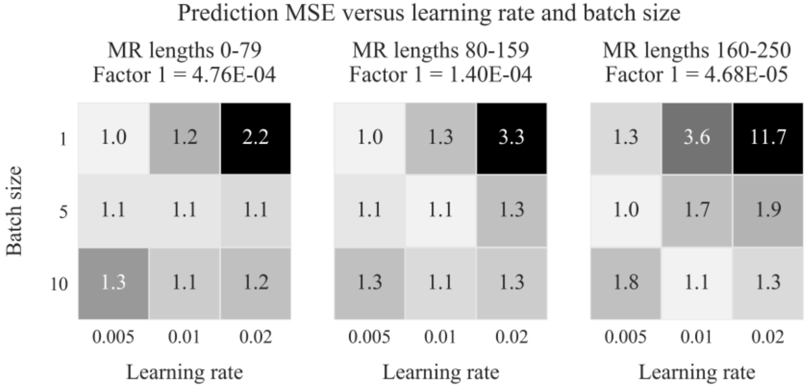


Figure 35: Prediction MSE versus learning rate and batch size



The input normalization is also tested against the batch size in Figure 36 and Figure 37. A characteristic shared across all batch sizes and number of training examples is that the statistical normalization takes longer, while linear normalization is indifferent to no normalization at all. Their accuracies, however, are much more discrepant. Networks trained with unnormalized data achieved in average 35 times worse MSE results than their competitors. This behavior is even more accentuated when the batch size is one.

Figure 36: Training time versus input scaling and batch size

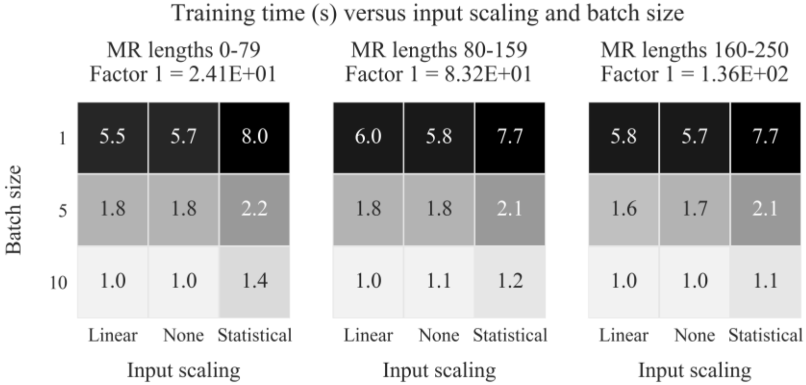
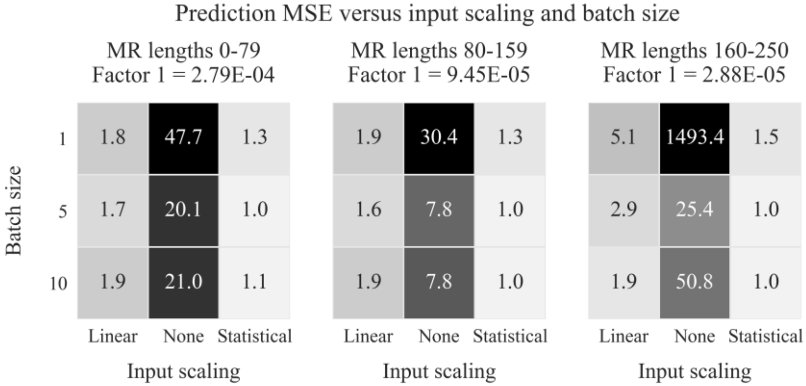


Figure 37: Prediction MSE versus input scaling and batch size



Statistical normalization achieved competitive results in comparison to linear normalization. However, as calculated in Table 14, the gain in accuracy transcends the loss in time.

Table 14: Performance comparison for input normalization (median values)

Input normalization	MSE	MSE [% to Linear]	Time [s]	Time [% to Linear]
Linear	0.000352	0%	92	0%
Statistical	0.000214	-39%	114	+24%

The analysis realized so far aimed on selecting the most appropriate hyperparameters to be used with the type of data addressed in this work, which are listed in Table 15.

Table 15: Collection of chosen hyperparameters

Hyperparameter	Value
Size of input and output layers	1
Number of hidden layers	2
Size of hidden layers	10
Input normalization	Statistical
Initializer	Glorot uniform
Output layer activation function	Sigmoid
Hidden layer activation function	ELU
Cost function	MSE
Optimizer	Adam
Learning rate	0.01
Batch size	10

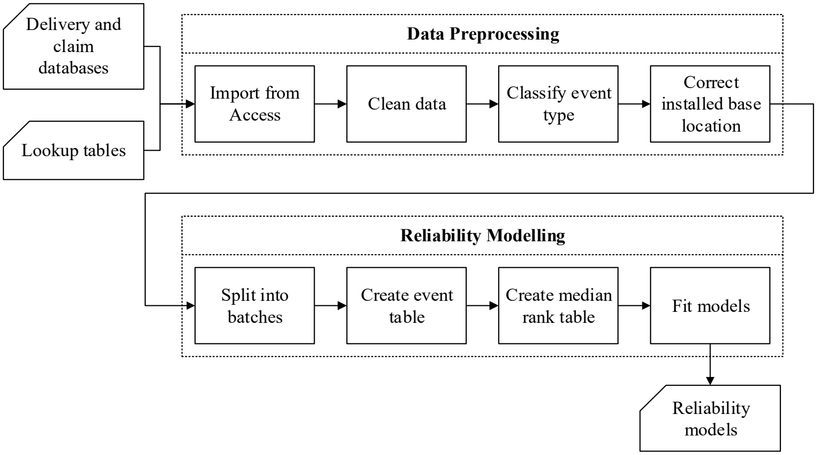
5 IMPLEMENTATION AND RESULTS

5.1 IMPLEMENTATION

This work attempts to reach results in an automatic manner that has not been tested previously in Bosch Rexroth. Although big amount of data was covered so far, this could be increased dramatically if the concept proves to be worth and new product branches are considered. Due to the flexibility, knowledge plurality and processing power required by this work, conventional tools available in the market could not fulfill the entire task. Some tools, however, were employed either indirectly or for support during the development. Examples are Microsoft Power BI, Microsoft Office Excel, ReliaSoft Weibull++ and SAP. A software capable of handling all subjects covered in this work alone was implemented using Python 3.6. Python is a high-level general-purpose programming language that is much known for its agility and power when working with data analysis. A list of used libraries is found in APPENDIX A – Python Libraries.

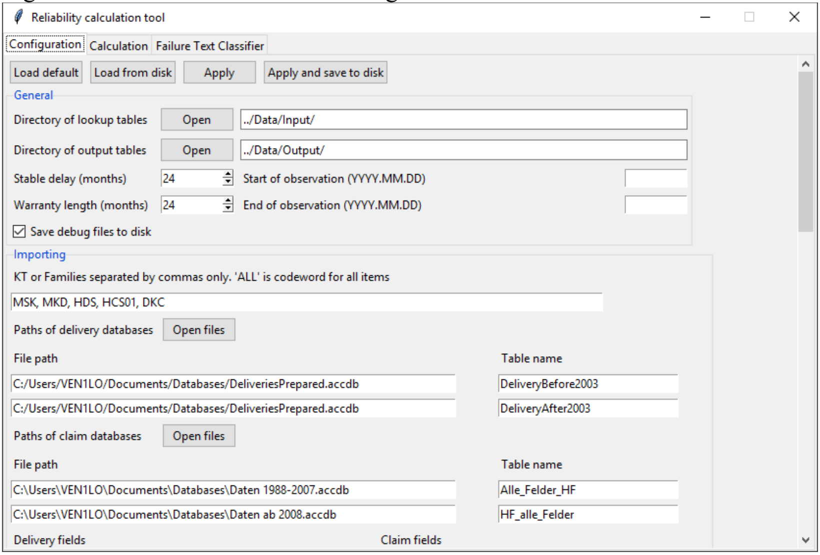
The software follows the façade pattern, which is a design commonly used in object-oriented programming for providing an easy interface of objects that handle calculations more complex underneath. It is divided as shown in Figure 38 into two main blocks: Data Preprocessing and Reliability Modelling. The first is fed with the delivery and claim databases, as well as several lookup tables for the linking of customer and product codes for example. Its outputs are cleaned delivery and claim tables that are directly handled to the second block. The long cascade operation ends with the fitting of the reliability model, which in this case is set to be a parametric Weibull or a Neural Network. The model is then written to the disk, where another front-end tool can further utilize it for several applications.

Figure 38: Data flow



The tool is operated through a GUI written using the built-in library Tkinter. The 'Configuration' tab (Figure 39) provides to the user the freedom of tweaking all parameters related to the functions in order to customize the preprocessing or modelling automatic operations. In the 'Calculation' tab (Figure 40), the user can watch the progress being shown in the console window, allowing for easy debug. All messages are also stored in a local log file. Another console is included in the 'Failure Text Classification' tab (Figure 41), where the user can generate the object responsible for classifying the failure description texts. Since this process takes much time, after the fitting the model is stored in the disk where it can be further loaded to the application.

Figure 39: Software GUI - Tab 'Configuration'



General

Directory of lookup tables: ../Data/Input/

Directory of output tables: ../Data/Output/

Stable delay (months): 24

Start of observation (YYYY.MM.DD):

Warranty length (months): 24

End of observation (YYYY.MM.DD):

☑ Save debug files to disk

Importing

KT or Families separated by commas only. 'ALL' is codeword for all items

MSK, MKD, HDS, HCS01, DKC

Paths of delivery databases

File path	Table name
C:/Users/VEN1LO/Documents/Databases/DeliveriesPrepared.accdb	DeliveryBefore2003
C:/Users/VEN1LO/Documents/Databases/DeliveriesPrepared.accdb	DeliveryAfter2003

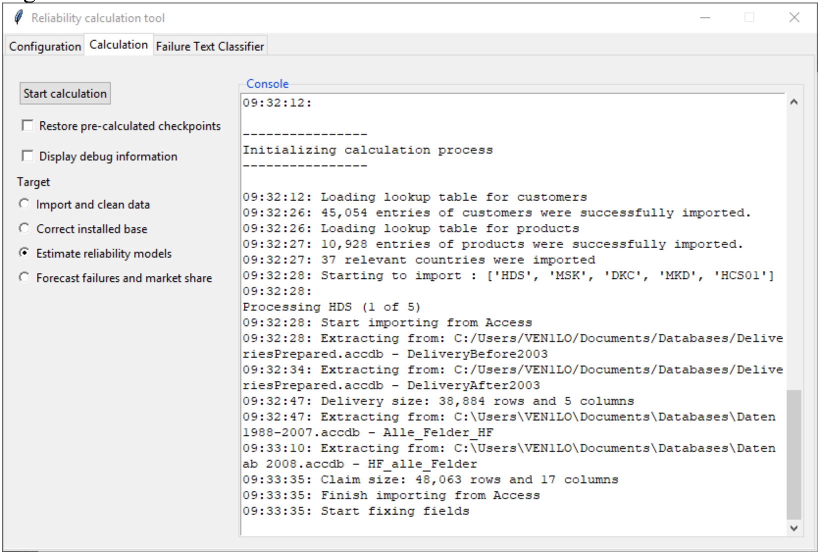
Paths of claim databases

File path	Table name
C:\Users\VEN1LO\Documents\Databases\Daten 1988-2007.accdb	Alle_Felder_HF
C:\Users\VEN1LO\Documents\Databases\Daten ab 2008.accdb	HF_alle_Felder

Delivery fields

Claim fields

Figure 40: Software GUI - Tab 'Calculation'



□ Restore pre-calculated checkpoints

□ Display debug information

Target

○ Import and clean data

○ Correct installed base

● Estimate reliability models

○ Forecast failures and market share

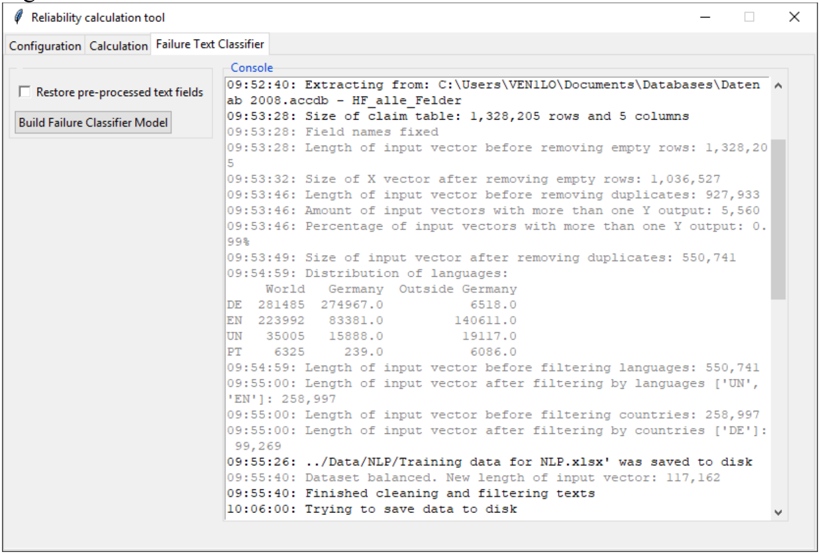
Console

```
09:32:12:

----------------
Initializing calculation process
----------------

09:32:12: Loading lookup table for customers
09:32:26: 45,054 entries of customers were successfully imported.
09:32:26: Loading lookup table for products
09:32:27: 10,928 entries of products were successfully imported.
09:32:27: 37 relevant countries were imported
09:32:28: Starting to import : ['HDS', 'MSK', 'DKC', 'MKD', 'HCS01']
09:32:28:
Processing HDS (1 of 5)
09:32:28: Start importing from Access
09:32:28: Extracting from: C:/Users/VEN1LO/Documents/Databases/Delive
riesPrepared.accdb - DeliveryBefore2003
09:32:34: Extracting from: C:/Users/VEN1LO/Documents/Databases/Delive
riesPrepared.accdb - DeliveryAfter2003
09:32:47: Delivery size: 38,884 rows and 5 columns
09:32:47: Extracting from: C:\Users\VEN1LO\Documents\Databases\Daten
1988-2007.accdb - Alle_Felder_HF
09:33:10: Extracting from: C:\Users\VEN1LO\Documents\Databases\Daten
ab 2008.accdb - HF_alle_Felder
09:33:35: Claim size: 48,063 rows and 17 columns
09:33:35: Finish importing from Access
09:33:35: Start fixing fields
```

Figure 41: Software GUI - Tab 'Failure Text Classifier'



Reliability calculation tool
Configuration
Calculation
Failure Text Classifier
☐ Restore pre-processed text fields
Build Failure Classifier Model
Console

```
09:52:40: Extracting from: C:\Users\VEN1LO\Documents\Databases\Daten
ab 2008.accdb - HF_alle_Felder
09:53:28: Size of claim table: 1,328,205 rows and 5 columns
09:53:28: Field names fixed
09:53:28: Length of input vector before removing empty rows: 1,328,20
5
09:53:32: Size of X vector after removing empty rows: 1,036,527
09:53:46: Length of input vector before removing duplicates: 927,933
09:53:46: Amount of input vectors with more than one Y output: 5,560
09:53:46: Percentage of input vectors with more than one Y output: 0.
99%
09:53:49: Size of input vector after removing duplicates: 550,741
09:54:59: Distribution of languages:
     World   Germany  Outside Germany
DE  281485  274967.0           6518.0
EN  223992   83381.0         140611.0
UN   35005   15888.0          19117.0
PT    6325     239.0           6086.0
09:54:59: Length of input vector before filtering languages: 550,741
09:55:00: Length of input vector after filtering by languages ['UN',
'EN']: 258,997
09:55:00: Length of input vector before filtering countries: 258,997
09:55:00: Length of input vector after filtering by countries ['DE']:
 99,269
09:55:26: ../Data/NLP/Training data for NLP.xlsx' was saved to disk
09:55:40: Dataset balanced. New length of input vector: 117,162
09:55:40: Finished cleaning and filtering texts
10:06:00: Trying to save data to disk
```

5.2 CLASSIFICATION OF EVENT TYPE

The failure classifier developed based on NLP techniques is implemented in the preprocessing part of the software. In total, it scanned and classified almost 500,000 failure claims. As detailed in chapter 3.3, it firstly checks the warranty codes, and if they are not decisive enough, other fields are verified with support of the logistic regression model. The results are always compared to a baseline classification that is solely based on the field V_CODE. In case the requirements for the implemented classifier are not fulfilled, the binary indicator of event type is set based on the existing V_CODE label. Table 16 shows how the classifier performed on the analysis of all data. The requirements were fulfilled in 29.7% of the cases, which were split between the use of the warranty code and the text fields. 34% of these supplied a different label than the existing, indicating that the classification through the V_CODE was faulty. 12% of all claims were classified as failure by the field of warranty code, and half of these provided corrections to the baseline classification. Another 36.5% of the total fulfilled the requirements of the text classifier and were fed to the logistic regression model. Almost half of these predictions supplied confidence above the specified threshold. Approximately eleven percent of the predicted labels satisfied the confidence requirements and supplied a new label that is different from the baseline.

Table 16: Benchmark of the failure classifier

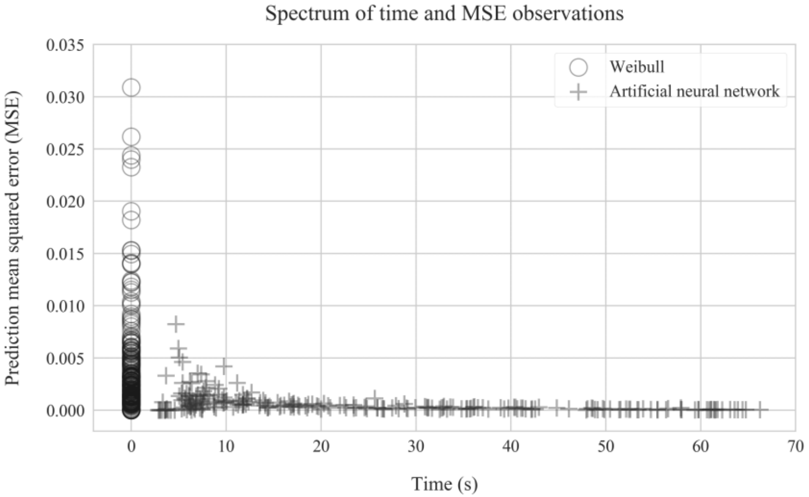
<table>
<tr><td colspan="7"><b>Total: 100%</b></td></tr>
<tr><td colspan="2">12%
Failure due to warranty code</td><td colspan="5">88%
Warranty code not decisive</td></tr>
<tr><td rowspan="2"></td><td rowspan="2"></td><td rowspan="2">13.7%
Empty text fields</td><td rowspan="2">37.8%
Language not supported</td><td colspan="3">36.5%
Predicted</td></tr>
<tr><td>18.8%
Low confidence</td><td colspan="2">17.7%
Enough confidence</td></tr>
<tr><td>6.1%</td><td>5.9%</td><td>13.7%</td><td>37.8%</td><td>18.8%</td><td>13.7%</td><td>4%</td></tr>
<tr><td><b>New value</b></td><td colspan="5">Not changed (identical to analyzing the V_CODE)</td><td><b>New value</b></td></tr>
</table>

5.3 ESTIMATION OF RELIABILITY MODELS

The final part of this work is focused on understanding how the estimation of reliability models through the Weibull fitting (parametric) and the feed-forward neural network (non-parametric) performed. Both approaches were used on the evaluation of all datasets addressed by this work. Similar to the analysis of the selection of the hyperparameters, the performance is measured by the required fitting time and the achieved mean squared error on the prediction of all empirical reliability values of the datasets.

The plot of Figure 42 presents an overview of the spectrum of time and MSE results of all predictions using the two approaches. It is clear that they differ significantly and do not overlap.

Figure 42: Spectrum of time to fit and MSE observations



In order to elaborate a more concise comparison, the datasets are split according to their sizes and censoring ratios. Nine clusters are created with three groups of quantity of entries and three groups of censoring ratios. Their relative quantities are shown in Table 17.

Table 17: Percentage of datasets in each cluster

Size Censoring	50 – 1,000	1,001 – 20,000	20,001 – 1,000,000
0.50 – 0.90	12.1%	9.4%	2.2%
0.90 – 0.95	7.2%	9.9%	3.6%
0.95 – 1.00	6.3%	25.3%	21.1%

Surprisingly, the Weibull fitting takes a relatively constant time independently of the dataset proprieties, as shown in Table 18. In the other hand, Table 19 indicates that ANN models require much more time to fit larger datasets. At the same time, more censored datasets are processed faster.

Table 18: Fitting time (s) of the Weibull model (cluster median values)

Size Censoring	50 – 1,000	1,001 – 20,000	20,001 – 1,000,000
0.50 – 0.90	0.001998	0.01999	0.001998
0.90 – 0.95	0.001998	0.01999	0.002000
0.95 – 1.00	0.001998	0.01998	0.001999

Table 19: Fitting time (s) of the ANN model (cluster median values)

Size Censoring	50 – 1,000	1,001 – 20,000	20,001 – 1,000,000
0.50 – 0.90	9.98	54.54	61.47
0.90 – 0.95	8.27	32.27	59.77
0.95 – 1.00	5.37	11.17	34.81

Table 20 shows that the MSE values achieved by the Weibull models do not appear to depend strongly on the size and censoring ratio of the datasets, repeating the conclusion made after analyzing Table 18. It is noticeable, however, that highly censored datasets perform better if more samples are available. Table 21 clearly shows that models based on artificial neural networks produce smaller MSE results for larger datasets. It also shows that lower censoring ratios allow better results for large datasets.

Table 20: Prediction MSE of the Weibull model (cluster median values)

Size / Censoring	50 – 1,000	1,001 – 20,000	20,001 – 1,000,000
0.50 – 0.90	0.002832	0.001932	0.005462
0.90 – 0.95	0.005099	0.001952	0.001119
0.95 – 1.00	0.003920	0.002833	0.001391

Table 21: Prediction MSE of the ANN model (cluster median values)

Size / Censoring	50 – 1,000	1,001 – 20,000	20,001 – 1,000,000
0.50 – 0.90	0.000765	0.000057	0.000026
0.90 – 0.95	0.001064	0.000204	0.000049
0.95 – 1.00	0.000810	0.000452	0.000199

Models based on the parametric Weibull distribution are exceptionally faster than artificial neural networks to fit any type of dataset, as shown in Table 22. The most similar performances are encountered in small and highly censored datasets, although the parametric method is still more than two thousand times faster. Because ANN suffer from increasing sizes, they are farther defeated in larger and more complete datasets.

Table 22: Time improvement of the Weibull over the ANN model (cluster median values)

Size / Censoring	50 – 1,000	1,001 – 20,000	20,001 – 1,000,000
0.50 – 0.90	4,995 X	25,787 X	30,758 X
0.90 – 0.95	4,139 X	16,144 X	29,890 X
0.95 – 1.00	2,689 X	5,589 X	17,413 X

While ANN models are slow to fit, Table 23 shows that they accomplish much smaller prediction errors. Again, their performance is mostly similar to the parametric approach in small datasets, achieving not more than five times better MSE values. In large datasets, however, this difference increases up to 212 times.

Table 23: MSE improvement of the ANN over the Weibull model (cluster median values)

Size **Censoring**	**50 – 1,000**	**1,001 – 20,000**	**20,001 – 1,000,000**
0.50 – 0.90	3.7 X	33.9 X	212 X
0.90 – 0.95	4.8 X	9.6 X	23 X
0.95 – 1.00	4.8 X	6.3 X	7 X

6 CONCLUSION

This work aimed to support the company Bosch Rexroth by providing insights of product reliability that are useful in the guidance of market strategies. It presented a holistic perspective to the problem of utilizing databases of deliveries and field claims commonly encountered in big companies, and generating reliability models using the most appropriate cleansing and modeling methods. Above all, it provided an extensive research of state-of-the-art approaches to this problem. Moreover, the considerable data volume addressed motivated the research and experiment of alternative approaches involving machine learning.

The cleansing of field data is normally addressed as a manual and laborious process that involves much expertise in order to obtain the best information out of the dirty data. Since the dimension of the problem covered by this work is much bigger than any other encountered in the literature, the author appealed to machine learning algorithms. Based on well-known approaches to sentiment analysis vastly encountered in the literature, an automated classifier algorithm that interprets claim texts was developed. In its core, a logistic regression model obtained 86.2% precision on the classification of the features extracted from a test dataset. Since this is a relatively simple model and offers plenty of room for improvement, the performance is considered good. Nevertheless, this is only part of the classifier and is used if certain conditions are satisfied, as explained in chapter 3.3. This accounted for 36.5% of the claims. In total, 70.3% of the labels were originated entirely from the field V_CODE, which for instance is recognized by the company as not entirely reliable. This can be a big source of uncertainties that is unfortunately not measurable. The text classifier was able to provide corrections to 10.1% of all claims.

Researches published in the literature describe extensively the use of theoretical distributions to model reliability estimates as the state-of-the-art. However, one of their assumptions that could not be fulfilled in this work is that the most appropriate distribution is known. Moreover, the interest of the author in exploring new solutions to well-known problems has led the work to incorporate artificial neural networks as a non-parametric approach to the modelling of the reliability cumulative distribution function. Both parametric and non-parametric methods were employed and compared. While the regression of the data to Weibull distributions is realized thousand times faster than fitting artificial neural networks, the latter achieved several times smaller prediction errors depending on the dataset. At first glance without knowing the application, it is impossible to choose the best method, because different situations offer particular constraints such as limited time or rigorous prediction precision. Although this work already covered a large amount of data, it served as a pilot program to future applications involving even more product families. Therefore, the big constraint on processing time influenced the selection of future modelling approaches using the Weibull distribution. One of the main factors that also accounted for the choice of Weibull is the traditional knowledge available in companies. These have reliability engineers trained to understand statistical distributions since decades, although still lack of experience on machine learning techniques.

As described in the literature review, most of the methods used for modeling incomplete claim data were studied for decades and are well consolidated. Some applications, however, require special adaptations and improvements. Throughout this work, the proposal of utilizing machine learning algorithms for the cleansing of large claim data and for the model estimation of data with unknown shape is what mostly motivated the author. The procedures realized were detailed and justified, and the author hopes that they may serve as base for other researchers in future works.

6.1 FURTHER IMPROVEMENTS AND RESEARCHES

Perfectly calculated reliability models do not account confidence and usefulness if based on weak data. The more corrupted is the data, more numerous and speculative are the assumptions required to work with it. Therefore, it is highly demanded that future works are carried out using databases that are more complete. Especially in cases where suspensions are inevitable due to uncontrolled concerns such as behavior of third parties, the available data must be as concise and reliable as possible. Significant effort was put in verifying which databases should be used and how accurate they are. This issue is extremely common because companies rarely perceive and evaluate the potential available in this data.

Some of the assumptions stablished during the development aimed for simplicity and might have to be reviewed if further studies towards more refined results are desired. These points are grouped and presented in the next subchapters.

6.1.1 Classification of incomplete observations

It was believed so far that every sold product would someday come to a failure, since it cannot run forever. For this reason, a date of failure or of suspension had to be assigned to every unit. Notwithstanding, an unknown number of units are discarded by customers without notice. This usually happens when they are too old or do not suit the application anymore. In these cases, the machines are set aside and might never fail in the future. Such cases should be treated differently, and their classification as truncated data might be more accurate than censored data.

Another hard assumption taken is that all failures occurred in units covered by warranty are reported to the company and stored in the respective database. However, some costumers – especially big companies – sometimes do not concern about repairing cheap equipment, especially if a spare item is already available. The data stored so far does not keep track of that, making these situations unrecognizable. The flaws in this assumption tend to form reliability models with longer life expectancy.

6.1.2 Parametric modelling

In the attempt of generalizing an optimal method for the parametric modelling while not evading the scope of this work, the two-parameter Weibull distribution was chosen. However, datasets with more samples might provide room for more complex distributions such as the three-parameter Weibull or the mixed-Weibull. The former is especially useful for avoiding the assumption of the period respective to the delivery and claim delays. In other words, a fixed time shift related to the delays was assumed, while a three-parameter estimation could evaluate it instead.

Moreover, failure modes of some products might not be modeled by shapes covered by the Weibull spectrum, and different distributions could be addressed. An automatic verification based on goodness-of-fit measures may be enough to choose the most appropriate distribution to each dataset before proceeding with the parameter estimations.

6.1.3 Measurement of error

A cost function in this context is responsible for determining the error between the model and the data used to create the model. The mean squared error (MSE) was the only cost function employed in the estimation of the parametric and nonparametric models, as well as their comparison. Whilst it is widely known and easily implemented, it has a few potential disadvantages.

All distributions were fitted by least-squares regression after applying logarithmic transformations to both axis for achieving a linear relation. This means that the line fitted between the points does not account for the actual shape of the unscaled data. Some authors defend that an adaptation to the cost function of the regression taking the scale in consideration shall be made. The most recommended adaptation is the weighted linear regression, which applies different weights to the distances between the data points and the fitted line.

Furthermore, most products considered in this work are expected to live much longer than the periods analyzed. This means that only the initial tail of the reliability curves related to early years are utilized. Therefore, it is much more important that this section of the curve presents less estimation error than its whole.

REFERENCES

[1] W. Q. Meeker and L. A. Escobar, *Statistical Methods for Reliability Data Using SAS Software*. 1997, p. 10.

[2] W. R. Blischke and D. N. P. Murthy, *Case Studies in Reliability and Maintenance*. Wiley, 2003.

[3] K.-M. Leung, R. M. Elashoff, and A. A. Afifi, "Censoring issues in survival analysis," *Annu. Rev. Public Health*, vol. 18, no. 1, pp. 83–104, May 1997.

[4] S. A. Lewis and T. G. Edwards, "Smart sensors and system health management tools for avionics and mechanical systems," presented at the 16th DASC. AIAA/IEEE Digital Avionics Systems Conference., Irvine, CA, USA, 1997, vol. 2, pp. 8.5-1-8.5–7.

[5] "IEC 60050-192," International Electrotechnical Commission, Geneva, Switzerland, International Standard, 2015.

[6] J. Z. Sikorska, M. Hodkiewicz, and L. Ma, "Prognostic modelling options for remaining useful life estimation by industry," *Mech. Syst. Signal Process.*, vol. 25, no. 5, pp. 1803–1836, Jul. 2011.

[7] C. Yang, C. Yang, and Q. Chen, "Developing predictive models for time to failure estimation," p. 6, 2016.

[8] H. Pham, Ed., *Springer Handbook of Engineering Statistics*. London: Springer, 2006.

[9] T. M. Welte and K. Wang, "Models for lifetime estimation: an overview with focus on applications to wind turbines," *Adv. Manuf.*, vol. 2, no. 1, pp. 79–87, Mar. 2014.

[10] G. Habchi, "An improved method of reliability assessment for suspended tests," *Int. J. Qual. Reliab. Manag.*, vol. 19, no. 4, pp. 454–470, Jun. 2002.

[11] K. Suzuki, "Estimation of Lifetime Parameters from Incomplete Field Data," *Technometrics*, vol. 27, no. 3, pp. 263–271, 1985.

[12] A. Jung, "Wettlauf der Systeme," *Der Spiegel*, vol. 16, 14-Apr-2014.

[13] "Bosch Rexroth AG," *About Bosch Rexroth - Bosch Rexroth AG*, 29-Nov-2018. [Online]. Available: https://www.boschrexroth.com/en/xc/en/company/about-bosch-rexroth/about-bosch-rexroth. [Accessed: 29-Nov-2018].

[14] S. Wu, "Warranty Data Analysis: A Review," *Qual. Reliab. Eng. Int.*, vol. 28, no. 8, pp. 795–805, Dec. 2012.

[15] R. B. Abernethy, *The New Weibull Handbook*, 5th ed. North Palm Beach, Fla.: R.B. Abernethy, 2008.

[16] J. P. Klein and M. L. Moeschberger, *Survival Analysis: Techniques for Censored and Truncated Data*, 2nd ed. New York, US: Springer, 2003.

[17] W. Q. Meeker and L. A. Escobar, *Statistical Methods for Reliability Data*, 1 edition. New York: Wiley-Interscience, 1998.

[18] E. F. Alsina, G. Cabri, and A. Regattieri, “A Neural Network Approach to Find The Cumulative Failure Distribution: Modeling and Experimental Evidence,” *Qual. Reliab. Eng. Int.*, vol. 32, no. 2, pp. 567–579, Mar. 2016.

[19] E. F. Alsina, M. Chica, K. Trawiński, and A. Regattieri, “On the use of machine learning methods to predict component reliability from data-driven industrial case studies,” *Int. J. Adv. Manuf. Technol.*, vol. 94, no. 5–8, pp. 2419–2433, Feb. 2018.

[20] K. D. Majeske, “A non-homogeneous Poisson process predictive model for automobile warranty claims,” *Reliab. Eng. Syst. Saf.*, vol. 92, no. 2, pp. 243–251, Feb. 2007.

[21] B. Rai and N. Singh, “Hazard rate estimation from incomplete and unclean warranty data,” *Reliab. Eng. Syst. Saf.*, vol. 81, no. 1, pp. 79–92, Jul. 2003.

[22] J. F. Lawless, J. D. Kalbfleisch, and S. Blumenthal, “Some Issues in the Collection and Analysis of Field Reliability Data,” in *Survival Analysis: State of the Art*, J. P. Klein and P. K. Goel, Eds. Dordrecht: Springer Netherlands, 1992, pp. 141–152.

[23] S. Wu, “A review on coarse warranty data and analysis,” *Reliab. Eng. Syst. Saf.*, vol. 114, pp. 1–11, 2013.

[24] J. D. Kalbfleisch and R. L. Prentice, *The Statistical Analysis of Failure Time Data, 2nd Edition*, 2nd ed. US: John Wiley, 2002.

[25] L. Chmura, P. H. F. Morshuis, E. Gulski, J. J. Smit, and A. Janssen, “Reliability estimation for populations with limited and heavily censored failure information,” presented at the 2013 IEEE Electrical Insulation Conference (EIC), Ottawa, ON, Canada, 2013, pp. 159–163.

[26] Y. Hong, W. Q. Meeker, and J. D. McCalley, “Prediction of remaining life of power transformers based on left truncated and right censored lifetime data,” *Ann. Appl. Stat.*, vol. 3, no. 2, pp. 857–879, Jun. 2009.

[27] W. R. Blischke and B. P. Iskandar, “Reliability and Warranty Analysis of a Motorcycle Based on Claims Data,” in *Case Studies in Reliability and Maintenance*, Wiley, 2003.

[28] Peter C. Sander, Luis M. Toscano, Steven Luitjens, Valia T. Petkova, Antoine Huijben, and Aarnout C. Brombacher,

"Warranty Data Analysis for Assessing Product Reliability," in *Case Studies in Reliability and Maintenance*, Wiley, 2003.

[29] S. Wu, "Warranty claim analysis considering human factors," *Reliab. Eng. Syst. Saf.*, vol. 96, no. 1, pp. 131–138, Jan. 2011.

[30] J. Jones and J. Hayes, "Investigation of the occurrence of: no-faults-found in electronic equipment," *IEEE Trans. Reliab.*, vol. 50, no. 3, pp. 289–292, Sep. 2001.

[31] M. Ilarslan and L. Y. Ungar, "Mitigating the Impact of False Alarms and No Fault Found Events in Military Systems," *IEEE Instrum. Meas. Mag.*, vol. 19, no. 4, pp. 15–21, Aug. 2016.

[32] H. Qi, S. Ganesan, and M. Pecht, "No-fault-found and intermittent failures in electronic products," *Microelectron. Reliab.*, vol. 48, no. 5, pp. 663–674, May 2008.

[33] M. R. Johnson and I. P. McCarthy, "Product recovery decisions within the context of Extended Producer Responsibility," *J. Eng. Technol. Manag.*, vol. 34, pp. 9–28, Oct. 2014.

[34] S. Wilson, T. Joyce, and E. Lisay, "Reliability estimation from field return data," *Lifetime Data Anal.*, vol. 15, no. 3, pp. 397–410, Sep. 2009.

[35] J. D. Kalbfleisch, J. F. Lawless, and J. A. Robinson, "Methods for the Analysis and Prediction of Warranty Claims: : Vol 33, No 3," vol. 33, no. 3, pp. 273–285, 1991.

[36] J. F. Lawless, "Adjustments for Reporting Delays and the Prediction of Occurred but Not Reported Events," *Can. J. Stat. Rev. Can. Stat.*, vol. 22, no. 1, pp. 15–31, 1994.

[37] L. Kann, "Statistical Failure Prediction with an Account for Prior Information," Doctor Rerum Naturalium, University of Würzburg, Würzburg, Germany, 2018.

[38] L. Wang and K. Suzuki, "Nonparametric Estimation of Lifetime Distribution from Warranty Data without Monthly Unit Sales Information," *J. Reliab. Eng. Assoc. Jpn.*, vol. 23, no. 1, pp. 145–154, Jan. 2001.

[39] "IEC 61649," International Electrotechnical Commission, Geneva, International Standard, 2008.

[40] W. Nelson, *Applied Life Data Analysis*. Wiley, 1982.

[41] R. Manzini, Ed., *Maintenance for industrial systems*. Dordrecht ; New York: Springer, 2010.

[42] M. Rausand and A. Høyland, *System Reliability Theory: Models, Statistical Methods, and Applications*, 2nd ed. Hoboken, NJ: Wiley-Interscience, 2004.

[43] J. F. Lawless, “Statistical Methods in Reliability,” vol. 25, no. 4, p. 13, 1983.

[44] A. Shen, J. Guo, Z. Wang, and W. Jia, “A novel reliability evaluation method on censored data,” *J. Mech. Sci. Technol.*, vol. 31, no. 3, pp. 1105–1117, Mar. 2017.

[45] Y. S. Oh and D. S. Bai, “Field data analyses with additional after-warranty failure data,” *Reliab. Eng. Syst. Saf.*, vol. 72, no. 1, pp. 1–8, Apr. 2001.

[46] S. M. Limon, “Reliability Estimation Considering Customer Usage Rate Profile and Warranty Claims,” Thesis, North Dakota State University, 2014.

[47] A. H. Soukhanov, *Microsoft Encarta College Dictionary: The First Dictionary For The Internet Age*. St. Martin’s Press, 2001.

[48] A. L. Samuel, “Some Studies in Machine Learning Using the Game of Checkers,” p. 21, 1959.

[49] M. E. Celebi and K. Aydin, *Unsupervised Learning Algorithms*. Springer, 2016.

[50] U. S. Shanthamallu, A. Spanias, C. Tepedelenlioglu, and M. Stanley, “A brief survey of machine learning methods and their sensor and IoT applications,” in *2017 8th International Conference on Information, Intelligence, Systems & Applications (IISA)*, Larnaca, 2017, pp. 1–8.

[51] D. S. Vijayarani and J. Ilamathi, “Preprocessing Techniques for Text Mining - An Overview,” *Int. J. Comput. Sci. Commun. Netw.*, vol. 5, no. 1, pp. 7–16, 2015.

[52] K. S. Jones, “Natural Language Processing: A Historical Review,” in *Current Issues in Computational Linguistics: In Honour of Don Walker*, A. Zampolli, N. Calzolari, and M. Palmer, Eds. Dordrecht: Springer Netherlands, 1994, pp. 3–16.

[53] S. R. Joseph, H. Hlomani, K. Letsholo, F. Kaniwa, and K. Sedimo, “Natural Language Processing: A Review,” *Appl. Sci.*, vol. 6, no. 3, p. 10, 2016.

[54] M. M. Mirończuk and J. Protasiewicz, “A recent overview of the state-of-the-art elements of text classification,” *Expert Syst. Appl.*, vol. 106, pp. 36–54, Sep. 2018.

[55] S. Sun, C. Luo, and J. Chen, “A review of natural language processing techniques for opinion mining systems,” *Inf. Fusion*, vol. 36, pp. 10–25, Jul. 2017.

[56] N. Indurkhya and F. J. Damerau, Eds., *Handbook of Natural Language Processing*, 2 edition. Boca Raton, FL: Chapman and Hall/CRC, 2010.

[57] D. W. Castro, E. Souza, D. Vitório, D. Santos, and A. L. I. Oliveira, “Smoothed n-gram based models for tweet language identification: A case study of the Brazilian and European Portuguese national varieties,” *Appl. Soft Comput.*, vol. 61, pp. 1160–1172, Dec. 2017.

[58] R. Xia, F. Xu, J. Yu, Y. Qi, and E. Cambria, “Polarity shift detection, elimination and ensemble: A three-stage model for document-level sentiment analysis,” *Inf. Process. Manag.*, vol. 52, no. 1, pp. 36–45, Jan. 2016.

[59] Y. Mejova and P. Srinivasan, “Exploring Feature Definition and Selection for Sentiment Classifiers,” in *Fifth International AAAI Conference on Weblogs and Social Media*, 2011, p. 4.

[60] D. Kim, D. Seo, S. Cho, and P. Kang, “Multi-co-training for document classification using various document representations: TF–IDF, LDA, and Doc2Vec,” *Inf. Sci.*, vol. 477, pp. 15–29, Mar. 2019.

[61] S. F. Chen and J. Goodman, “An Empirical Study of Smoothing Techniques for Language Modeling,” in *Proceedings of the 34th Annual Meeting on Associationfor Computational Linguistics*, 1996, pp. 310–318.

[62] W. Medhat, A. Hassan, and H. Korashy, “Sentiment analysis algorithms and applications: A survey,” *Ain Shams Eng. J.*, vol. 5, no. 4, pp. 1093–1113, Dec. 2014.

[63] C. Catal and M. Nangir, “A sentiment classification model based on multiple classifiers,” *Appl. Soft Comput.*, vol. 50, pp. 135–141, Jan. 2017.

[64] R. Xia, C. Zong, and S. Li, “Ensemble of feature sets and classification algorithms for sentiment classification,” *Inf. Sci.*, vol. 181, no. 6, pp. 1138–1152, Mar. 2011.

[65] Q. Cheng, P. K. Varshney, and M. K. Arora, “Logistic Regression for Feature Selection and Soft Classification of Remote Sensing Data,” *IEEE Geosci. Remote Sens. Lett.*, vol. 3, no. 4, pp. 491–494, Oct. 2006.

[66] D. W. Hosmer and S. Lemeshow, *Applied Logistic Regression. Solutions Manual to Accompany*, 2nd edition. New York: Wiley-Interscience, 2001.

[67] H.-F. Yu, F.-L. Huang, and C.-J. Lin, “Dual coordinate descent methods for logistic regression and maximum entropy models,” *Mach. Learn.*, vol. 85, no. 1–2, pp. 41–75, Oct. 2011.

[68] R. Rojas, *Neural Networks: A Systematic Introduction*. Berlin Heidelberg: Springer-Verlag, 1996.

[69] B. Müller, J. Reinhardt, and M. T. Strickland, *Neural Networks: An Introduction*. Springer Science & Business Media, 2012.

[70] R. Lippmann, “An introduction to computing with neural nets,” *IEEE ASSP Mag.*, vol. 4, no. 2, pp. 4–22, 1987.

[71] K. Hornik, “Approximation capabilities of multilayer feedforward networks,” *Neural Netw.*, vol. 4, no. 2, pp. 251–257, Jan. 1991.

[72] S. Haykin, *Neural Networks and Learning Machines: A Comprehensive Foundation*, 3 edition. New York: Prentice Hall International, 2008.

[73] V. Nair and G. E. Hinton, “Rectified Linear Units Improve Restricted Boltzmann Machines,” p. 8, 2010.

[74] D. Pedamonti, “Comparison of non-linear activation functions for deep neural networks on MNIST classification task,” Apr. 2018.

[75] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” *Commun. ACM*, vol. 60, no. 6, pp. 84–90, May 2017.

[76] S. S. Haykin, *Neural Networks and Learning Machines*, 3rd ed. New York: Prentice Hall, 2009.

[77] X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks,” in *13th International Conference on Artificial Intelligence and Statistics (AISTATS)*, Sardinia, Italy, 2010, vol. 9, p. 8.

[78] H. Lo, “Deep Networks: Applications, Interpretability, and Optimization,” University of Massachusetts, Boston, 2016.

[79] M. Shanker, M. Y. Hu, and M. S. Hung, “Effect of data standardization on neural network training,” *Omega*, vol. 24, no. 4, pp. 385–397, Aug. 1996.

[80] J. Sola and J. Sevilla, “Importance of input data normalization for the application of neural networks to complex industrial problems,” *IEEE Trans. Nucl. Sci.*, vol. 44, no. 3, pp. 1464–1468, Jun. 1997.

[81] E. Guresen and G. Kayakutlu, “Definition of artificial neural networks with comparison to other networks,” *Procedia Comput. Sci.*, vol. 3, pp. 426–433, 2011.

[82] S. Ruder, “An overview of gradient descent optimization algorithms,” Sep. 2016.

[83] D. P. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization,” Dec. 2014.

[84] S. Bock, J. Goppold, and M. Weiß, “An improvement of the convergence proof of the ADAM-Optimizer,” Apr. 2018.

[85] K. C. Datsiou and M. Overend, “Weibull parameter estimation and goodness-of-fit for glass strength data,” *Struct. Saf.*, vol. 73, pp. 29–41, Jul. 2018.

[86] L. G. Johnson, *The statistical treatment of fatigue experiments*. Amsterdam; New York: Elsevier Pub. Co., 1964.

[87] R. F. Stapelberg, *Handbook of Reliability, Availability, Maintainability and Safety in Engineering Design*. London: Springer, 2009.

[88] S. H. Gonçalves, “Methodology For The Computational Two-Parameter Weibull Distribution,” Cologne University of Applied Sciences, 2015.

[89] J. Jacquelin, “A reliable algorithm for the exact median rank function,” *IEEE Trans. Electr. Insul.*, vol. 28, no. 2, pp. 168–171, Apr. 1993.

[90] A. Bernard and E. C. Bos-Levenbach, “The plotting of observations on probability paper,” *Stat. Neerlandica*, vol. 7, no. 3, pp. 163–173, 1953.

[91] U. Genschel and W. Q. Meeker, “A Comparison of Maximum Likelihood and Median-Rank Regression for Weibull Estimation,” *Qual. Eng.*, vol. 22, no. 4, pp. 236–255, Aug. 2010.

[92] J. C. Fothergill, “Estimating the cumulative probability of failure data points to be plotted on Weibull and other probability paper,” *IEEE Trans. Electr. Insul.*, vol. 25, no. 3, pp. 489–492, Jun. 1990.

[93] J. J. Filliben, “The Probability Plot Correlation Coefficient Test for Normality,” *Technometrics*, vol. 17, no. 1, pp. 111–117, 1975.

[94] M. Cacciari and G. C. Montanari, “Discussion, with reply, on "Estimating the cumulative probability of failure data points to be plotted on Weibull and other probability paper,” *IEEE Trans. Electr. Insul.*, vol. 26, no. 6, pp. 1224–1229, Dec. 1991.

[95] I. J. Davies, “Unbiased estimation of the Weibull scale parameter using linear least squares analysis,” *J. Eur. Ceram. Soc.*, vol. 37, no. 8, pp. 2973–2981, Jul. 2017.

[96] A. N. O’Connor, *Probability Distributions Used in Reliability Engineering*. RIAC, 2011.

[97] C. Forbes, *Statistical Distributions*, 4 edition. Hoboken, N.J: Wiley, 2010.

[98] U. Abbas and G. S. Shah, “Assessment of Weibull Parameter by Five Numerical Methods and Estimation of Wind Speed at Rotterdam, Netherland,” *J. Basic Appl. Sci.*, vol. 12, pp. 245–251, Jun. 2016.

[99] V. I. Kogan, “Life Distribution Models in Engineering Reliability (Nonrepairable Devices),” *IEEE Trans. Energy Convers.*, vol. EC-1, no. 1, pp. 54–60, Mar. 1986.

[100] E. E. Elmahdy, “Modelling Reliability Data with Finite Weibull or Lognormal Mixture Distributions,” *Appl. Math. Inf. Sci.*, vol. 11, no. 4, pp. 1081–1089, Jul. 2017.

[101] A. C. Cohen, “Maximum Likelihood Estimation in the Weibull Distribution Based on Complete and on Censored Samples,” *Technometrics*, vol. 7, no. 4, p. 579, Nov. 1965.

[102] L. A. Ferreira and J. L. Silva, “Parameter estimation for Weibull distribution with right censored data using EM algorithm,” *Eksploat. Niezawodn. - Maint. Reliab.*, vol. 19, no. 2, pp. 310–315, Mar. 2017.

[103] D. Cousineau, “Fitting the three-parameter weibull distribution: review and evaluation of existing and new methods,” *IEEE Trans. Dielectr. Electr. Insul.*, vol. 16, no. 1, pp. 281–288, Feb. 2009.

[104] M. Cacciari, G. Mazzanti, and G. C. Montanari, “Comparison of maximum likelihood unbiasing methods for the estimation of the Weibull parameters,” *IEEE Trans. Dielectr. Electr. Insul.*, vol. 3, no. 1, pp. 18–27, 1996.

[105] K. Fung and A. K. S. Jardine, “Weibull parameter estimation,” *Microelectron. Reliab.*, vol. 22, no. 4, pp. 681–684, Jan. 1982.

[106] E. L. Kaplan and P. Meier, “Nonparametric Estimation from Incomplete Observations,” *J. Am. Stat. Assoc.*, vol. 53, no. 282, p. 457, Jun. 1958.

[107] K. R. Skinner, J. B. Keats, and W. J. Zimmer, “A comparison of three estimators of the Weibull parameters,” *Qual. Reliab. Eng. Int.*, vol. 17, no. 4, pp. 249–256, Jul. 2001.

[108] B. W. Gillespie, “Topics in Kaplan-Meier estimation,” Temple University Graduate Board, 1989.

[109] G. P. Zhang, “Neural Networks for Time-Series Forecasting,” in *Handbook of Natural Computing*, G. Rozenberg, T. Bäck, and J. N. Kok, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 461–477.

[110] A. Z. Al-Garni and A. Jamal, “Artificial neural network application of modeling failure rate for Boeing 737 tires,” *Qual. Reliab. Eng. Int.*, vol. 27, no. 2, pp. 209–219, Mar. 2011.

[111] P. S. Rajpal, K. S. Shishodia, and G. S. Sekhon, “An artificial neural network for modeling reliability, availability and maintainability of a repairable system,” *Reliab. Eng. Syst. Saf.*, vol. 91, no. 7, pp. 809–819, Jul. 2006.

[112] K. Xu, M. Xie, L. C. Tang, and S. L. Ho, “Application of neural networks in forecasting engine systems reliability,” *Appl. Soft Comput.*, vol. 2, no. 4, pp. 255–268, Feb. 2003.

[113] M. Marseguerra, E. Zio, M. Ammaturo, and V. Fontana, “Predicting reliability via neural networks,” in *Annual Reliability and Maintainability Symposium, 2003.*, Tampa, FL, USA, 2003, pp. 196–201.

[114] Z. Tian, “A neural network approach for remaining useful life prediction utilizing both failure and suspension data,” in *2010 Proceedings - Annual Reliability and Maintainability Symposium (RAMS)*, San Jose, CA, USA, 2010, pp. 1–6.

[115] P.-F. Pai and K.-P. Lin, “Application of Hybrid Learning Neural Fuzzy Systems in Reliability Prediction,” *Qual. Reliab. Eng. Int.*, vol. 22, no. 2, pp. 199–211, Mar. 2006.

[116] M. C. Liu, W. Kuo, and T. Sastri, “An exploratory study of a neural network approach for reliability data analysis,” *Qual. Reliab. Eng. Int.*, vol. 11, no. 2, pp. 107–112, 1995.

[117] M. Dehghan and R. Hoseinnezhad, “Estimation of components reliability in petrochemical plants using a neural-Weibull lifetime model,” *Chem. Eng. Commun.*, vol. 196, no. 8, pp. 917–931, 2009.

[118] J. T. Luxhoj and H.-J. Shyur, “Reliability curve fitting for aging helicopter components,” *Reliab. Eng. Syst. Saf.*, vol. 48, no. 3, pp. 229–234, Jan. 1995.

[119] N. Karunanithdi and Y. Malaiy, “Using Neural Networks in Reliability Prediction,” p. 7, 1992.

[120] V. Mudunuru, “Modeling and Survival Analysis of Breast Cancer: A Statistical, Artificial Neural Network, and Decision Tree Approach,” University of South Florida, USA, 2016.

[121] H. N. Dezfouli, M. R. A. Bakar, and H. N. Dezfouli, “Feed forward neural networks models for survival analysis,” in *2012 International Conference on Statistics in Science, Business and Engineering (ICSSBE)*, 2012, pp. 1–5.

[122] “IEEE 100,” IEEE, Standard, Dec. 2000.

[123] A. K. Jain, M. N. Murty, and P. J. Flynn, “Data Clustering: A Review,” *ACM Comput. Surv.*, vol. 31, no. 3, pp. 264–323, Sep. 1999.

[124] G. Klutke, P. C. Kiessler, and M. A. Wortman, “A critical look at the bathtub curve,” *IEEE Trans. Reliab.*, vol. 52, no. 1, pp. 125–129, Mar. 2003.

[125] D. N. P. Murthy and K. A. H. Kobbacy, Eds., *Complex System Maintenance Handbook*. London: Springer, 2008.

[126] A. F. Agarap, “Deep Learning using Rectified Linear Units (ReLU),” Mar. 2018.

[127] L. Trottier, P. Giguère, and B. Chaib-draa, “Parametric Exponential Linear Unit for Deep Convolutional Neural Networks,” May 2016.

[128] A. L. Maas, A. Y. Hannun, and A. Y. Ng, “Rectifier Nonlinearities Improve Neural Network Acoustic Models,” presented at the 30th International Conference on Machine Learning, Georgia, USA, 2013, p. 6.

APPENDIX A – Python Libraries

The following table contains the names of all libraries utilized in the software developed, as well as their version, function and license type.

Table 24: Python libraries utilized

Library	Function	Version	License
collections	Data structure	Built-in	PSF
os	File manipulation	Built-in	PSF
datetime	Time operations	Built-in	PSF
time	Time operations	Built-in	PSF
string	Character manipulation	Built-in	PSF
re	Regular expression operations	Built-in	PSF
tkinter	Graphical interface	Built-in	PSF
queue	Concurrent execution	Built-in	PSF
threading	Concurrent execution	Built-in	PSF
signal	Interprocess communication	Built-in	PSF
logging	Log recording	Built-in	PSF
math	Mathematical operations	Built-in	PSF
numpy	Mathematical operations	1.14.3	BSD
scipy	Mathematical operations (linear regression)	1.1.0	BSD
pandas	Data manipulation	0.23.0	BSD
matplotlib	Plotting	2.2.2	BSD
scikit-learn	Machine learning and file manipulation	0.19.1	BSD
nltk	Natural language toolkit	3.3	Apache 2.0
theano	Artificial neural network	1.0.4	BSD
keras	Artificial neural network	2.2.4	MIT
pycld2	Language identification	0.31	Apache 2.0
pypyodbc	Database manipulation	1.3.4	MIT

APPENDIX B – Functional Diagrams

Figure 43: Correction of installed base

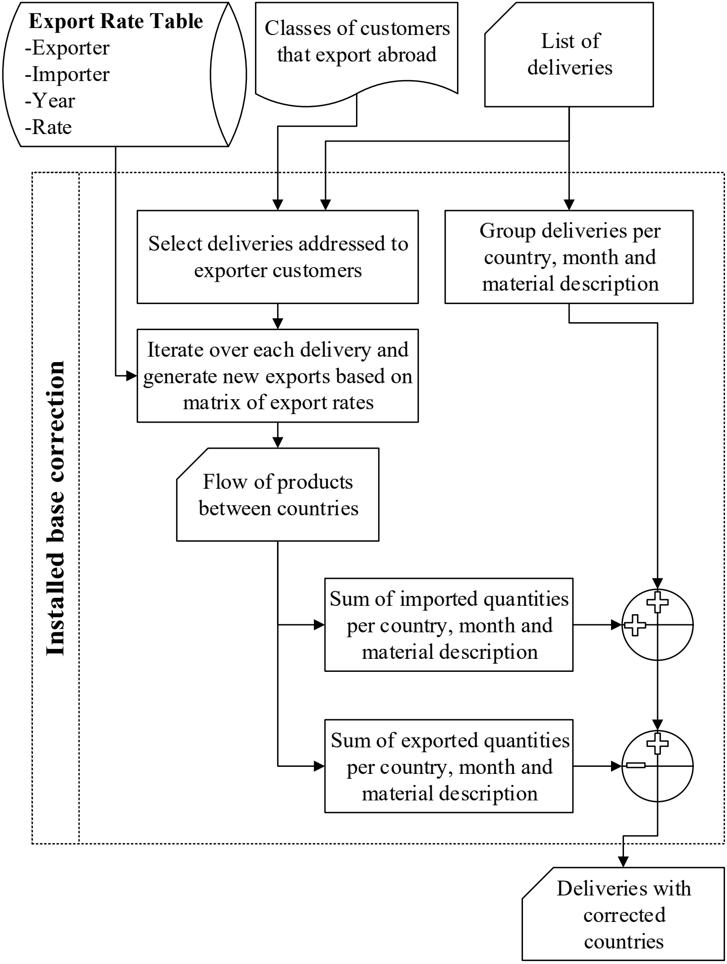
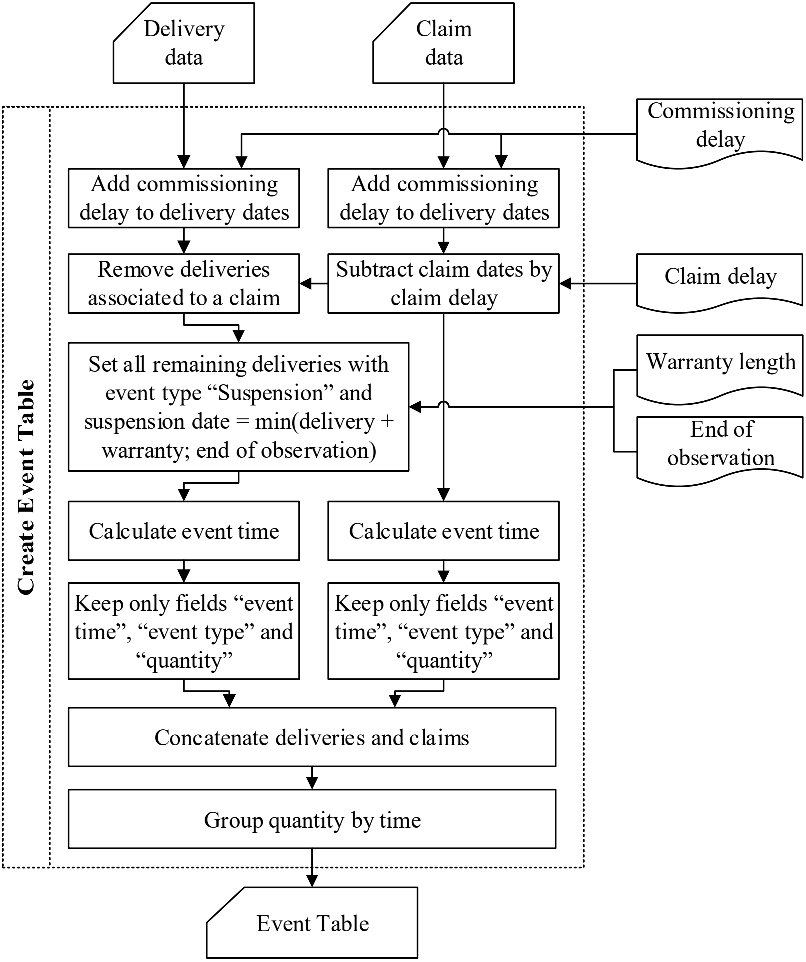

Export Rate Table
-Exporter
-Importer
-Year
-Rate
Classes of customers that export abroad
List of deliveries
Installed base correction
Select deliveries addressed to exporter customers
Group deliveries per country, month and material description
Iterate over each delivery and generate new exports based on matrix of export rates
Flow of products between countries
Sum of imported quantities per country, month and material description
Sum of exported quantities per country, month and material description
Deliveries with corrected countries

Figure 44: Creation of *event table*


Delivery data
Claim data
Commissioning delay
Add commissioning delay to delivery dates
Add commissioning delay to delivery dates
Remove deliveries associated to a claim
Subtract claim dates by claim delay
Claim delay
Set all remaining deliveries with event type "Suspension" and suspension date = min(delivery + warranty; end of observation)
Warranty length
End of observation
Calculate event time
Calculate event time
Keep only fields "event time", "event type" and "quantity"
Keep only fields "event time", "event type" and "quantity"
Concatenate deliveries and claims
Group quantity by time
Create Event Table
Event Table