POLITECNICO DI TORINO

Master's Degree in ICT for Smart Societies

Master's Degree Thesis

Industrial Vehicles' Inventory Management using Data-Driven Stochastic Optimization and Forecasting

Supervisor

Prof. Luca VASSIO

Co-Supervisor

Prof. Marco MELLIA

Candidate

Giorgia GUARNOTTA

December 2022

Abstract

Warehouse optimization is a process that aims at improving the management of time, space, and resources inside a warehouse, minimizing the overall costs yet ensuring a satisfactory quality of service for the customers.

When running an efficient warehouse most effort is on the inventory management, which is a well-known challenge for businesses: on the one hand they have to avoid late product supply, which would result in lost profits, while on the other hand they need to focus on a careful inventory control to mitigate the rise of costs, caused by an excessive accumulation of products.

The purpose of this thesis is that of supporting industrial fleet managers when vehicles undergo maintenance, providing a way to cope with the need of spare parts under realistic situations, represented by the uncertain nature of components demand. This work proposes a Two-stage Stochastic Mixed-Integer Nonlinear Problem that aims at minimising both spare parts inventory costs and vehicle offline periods when the requested items are not immediately available.

The presence of historical data allows to implement a data-driven approach: we use data collected between 2020 and 2022 from industrial vehicles' maintenance history to investigate the spare parts demand distributions and provide an efficient forecast.

First, we investigate the case in which the demand distribution is stationary and does not change over time, providing an optimal target inventory level to maintain for ensuring an efficient warehouse management. Secondly, we study the possibility that the demand may vary over different time periods, thus providing the optimal order quantity per spare part and keeping into account the difference in the shipment time per component. Finally, we also exploit machine learning techniques, introducing the uncertainty of forecast of spare parts demand in the optimization process.

Results show that, through automation and a careful stock control strategy, businesses are able to improve customer satisfaction and reduce their inventory holding costs, especially if compared with manual warehouse management or naive strategies.

Acknowledgements

Non sono brava con i finali. Mettere dei punti è sempre sembrato qualcosa di troppo definitivo e assoluto, e scrivere questi ringraziamenti in un certo senso lo è stato. Rende reale la fine di un periodo della mia vita, che nel bene e nel male mi ha in gran parte definita. Però questo punto va messo, per permettermi di cominciare un nuovo capitolo, e l'unico modo che mi viene in mente per rendere giustizia ad una fine importante e anche un po' sofferta come questa è addolcirla dicendo grazie.

Grazie ai miei relatori, i Prof. Vassio L. e Mellia M., che mi hanno accompagnata, guidata e supportata in questa fase conclusiva del mio percorso accademico, trasmettendomi il desiderio di imparare sempre qualcosa di nuovo. E' stato un piacere affrontare questo percorso con voi, sempre disponibili e pronti ad aiutarmi nei momenti di difficoltà.

Grazie ai miei genitori. A mio padre, che mi ha dato il colore dei suoi occhi, un futuro e tutte le infinite possibilità che ci sono in mezzo. Noi siamo uguali. A mia madre, che mi abbraccia ancora come se fossi la sua bambina, a cui portava un pupazzetto ogni sera tornata da lavoro. Lo sono ancora. Senza di voi e il vostro incessante sostegno, anche a distanza, tutto questo non sarebbe stato possibile. Vi voglio bene. Grazie a mia sorella, che nonostante sia un'adulta ormai, per me è ancora la bimba con la r moscia che facevo spaventare saltando fuori dalle stanze buie. Se potessi tornare indietro, giocherei con te ogni volta che me lo chiedi. Grazie per essere qui con me in questo giorno importante. Ti voglio bene.

Ai miei colleghi, che ho trovato per caso scrivendo randomicamente il primo anno, cercando dei compagni per un laboratorio. Sono contenta di avere affrontato questo percorso con voi, tra Bot di Telegram, costine bruciate e ballerini di dubbia bravura in piscina.

Ai miei amici e alle mie amiche, che anche a distanza hanno tutti trovato qualche momento nelle loro giornate frenetiche per chiedermi come stessi, cosa gli raccontassi di nuovo, facendomi sentire tutto il loro affetto. Vi voglio bene.

In modo particolare, a quei diari in prima media in cui scrivevamo i nostri sogni imbarazzanti, alle confidenze, ai problemi, alle lacrime e ai sorrisi grandi, alla distanza che non ci ha mai separate, ai "caffè?" e ai "Dieci minuti e sono sotto casa

tua", alla complicità che ci rende complementari e ai 14 anni che ci sono in mezzo in cui sei stata la mia lolly.
E ancora, alle mie "ame", alle bevute al solito posto, alle canzoni di High School Musical in macchina a squarciagola, alle citazioni di Aldo Giovanni e Giacomo, alle lacrime asciugate, ai Babbo Natale segreto e a tutte le volte che ci siamo sorrise e volute bene anche a distanza, tramite uno schermo, ma sempre presenti le une per le altre. Siete mie sorelle.

E grazie a Vittorio. Sei stato la mia scialuppa di salvataggio nei momenti in cui pensavo di affondare, la mano che ho stretto quando ho avuto paura, l'ideale di persona a cui mi ispiro e la persona con cui passerei tutte le mie serate guardando serie tv e film strani, che solo noi sembriamo capire. Sei la luce e il mattino.

Table of Contents

List of Tables

List of Figures

Acronyms

OR
: operational research

SKU
: stock keeping units

ADI
: advance demand information

ABC
: always better control

EOQ
: economic order quantity

DLP
: deterministic linear program

MSSP
: multi stage stochastic program

RL
: reinforcement learning

MILP
: mixed integer linear program

KPI
: key performance indicator

CTMC
: continuous time markov chain

SVR
: support vector regressor

RF
: random forest

ANN
: artificial neural network

PM
: preventive maintenance

CAN
: controller area network

SPN
: supsect parameters numbers

CDF
: cumulative distribution function

LR
: linear regressor

GB
: gradient boosting

MP
: mathematical programming

ML
: machine learning

ECDF
: empirical cumulative distribution function

PMF

probability mass function

PDF

probability density function

LP

linear program

NLP

non linear program

DEP

deterministic equivalent program

MSE

mean square error

RMSE

root mean square error

MAE

mean absolute error

AWS

amazon web service

MINLP

mixed integer non linear program

Chapter 1

Introduction

1.1 Motivation and Objectives

Warehouse optimization is an essential process for efficiently running warehouses of all sizes. In particular, it exploits automated solutions to improve the management of time, space and resources inside the warehouse itself, minimizing costs while meeting customers' needs. Being an important step of the whole supply chain process, a systematic warehouse management has significant effects on the business profits, in terms of reducing costs, planning and predicting customers' requests, thus rising their satisfaction, contributing to the overall enterprise's growth.
The focus of this work is specifically on stock management, being one of the practices on which to act for increasing the efficiency of the process. In this context, we collaborate with a company providing telematics services which support fleet managers in scheduling maintenance operations. The proposed approach aims at implementing an automated solution to handle spare parts requests coming from a large company producing vehicles for goods and people transportation. Starting from the provided heterogeneous input data, related to the maintenance history and past usage of the vehicles under study, it is possible to analyse some of the main requirements that motivate this thesis project.

1.1.1 Research Questions

The proposed work aims at solving the following relevant points:

- As soon as industrial vehicles undergo maintenance, it would be better to already have the needed spare parts in the inventory, in order to avoid long waiting times before changing the eventually damaged components. Of course, this is ensured if the orders quantities are placed correctly by the fleet managers, avoiding late supply. Moreover, the quantities of spare parts to order are highly influenced by the lead time that it takes for them to be shipped, which is considered deterministic.

- The uncertainty of the demand for the spare parts must be correctly managed. When demand is uncertain and difficult to forecast, ordering the right quantities of product at the right time is essential.
- For what concerns the monetary costs, the company wishes for an improvement in terms of expenses for the holding of spare parts quantities in the inventory. Notice that at the moment there is no intelligent strategy adopted by them for the management of the inventory.

To this end, in chapter 1 we define the problem and the context in which it takes place; in chapter 2, we provide a review of the current literature that is relevant for this case study; in chapter 3 we introduce the employed dataset with an overview of its characteristics and peculiarities; in chapter 4 we present the relevant theory used to support this thesis project; finally, in chapter 5 we describe in detail the followed methodology and the corresponding results of the work and performances.

1.1.2 Workflow

In particular, the workflow of this project is composed of 3 phases, each of which can be divided into sub-phases, as shown in Figure 1.1.

Figure 1.1: Workflow of the thesis

1. **Data Pre-Processing**, consisting in: an initial description of the exploited tools; data transformation for putting together all the meaningful information contained in the different available datasets; preparation and characterisation of the resulting dataset to understand the behaviour of the data; a critical analysis on the different maintenance cycles of the vehicles under study to get to the spare parts demand distributions.

2. **Forecast**, consisting in the analysis of the adopted training strategy and the Random Forest regression algorithm for specific categories of vehicles with long maintenance cycles.

3. **Optimization Model**, consisting in: an initial description of the tools used to build and solve the model; the statement of the optimization problem and its characteristics; the mathematical formulation of both the stationary and non-stationary optimization models.

1.2 Warehouse Management

Warehouse management includes all the processes that are needed to correctly run the daily operations of a warehouse. Some examples are the organization of the space inside of it to maximize the storage capacity, the management of the stock quantities themselves, the correct choice of the staff, the coordination among suppliers and transportation companies to handle shipment and the subsequent fulfilment of the customers' orders. An effective cooperation among these activities should ensure an increase in productivity and a reduction of costs.

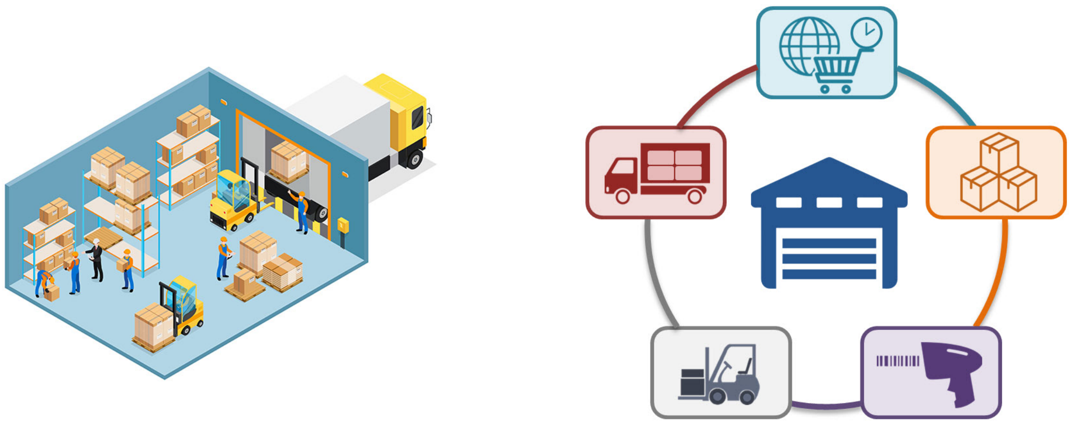


Figure 1.2: Warehouse Management

1.2.1 The importance of Warehouse Management in the Supply Chain process

It is important to notice that any improvement in the management of a warehouse has effects on the whole supply chain process. Supply chain in fact is

> [...] the process generated from the time the customer places an order to the

moment the product or service has been delivered and charged. [1]

In other words, the supply chain refers to the complete flow of a product until it is sold and it involves all those activities in charge of following the whole product life cycle, from the material purchase to manufacture it, up to the arrival of the end product to the customer. The supply chain is also known as the value chain, since products increase their value as they advance through its stages (Figure 1.3). After having obtained raw materials from the designated suppliers, the first stage is related to the manufacturing process itself, in which the producer will transform them into the end product. The second stage is a broader set called logistic involving storage, transportation, and distribution management. The final product is distributed to warehouses and distribution centres, and logistic takes care of the efficient and in-time traveling of the good along the supply chain. Products also need to adhere to some guidelines and some tariffs must be paid. In the final stage the goods reach the customers from the point of sale or directly from the warehouse, transported by delivery drivers across the country.

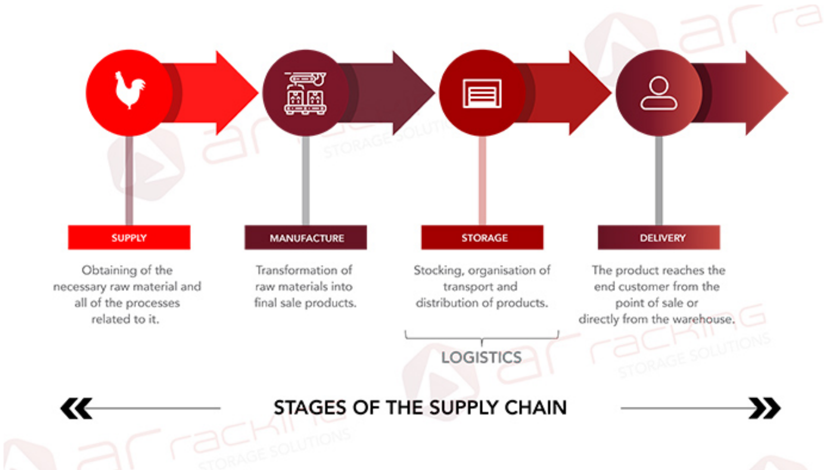


Figure 1.3: Supply Chain Stages
[2]

However, the first definition of supply chain management came out in 1982 thanks to Oliver and Webber [3], who defined it as a technique to reduce stocks in companies belonging to the same supply chain. Thus the term was used specifically to speak of warehouse and inventory management within a supply chain, when the concept actually involves the management of the whole chain.

As a matter of fact, our focus is on the logistic stage, in particular to the part of it comprising management of the warehouse, and it proves the fact that a correct handling of this step influences the whole supply chain, aiming at achieving customer satisfaction through effective order management and optimization practices.

1.2.2 Warehouse Optimization Strategies

When managing a warehouse there are some strategies that result to be useful to bring advantages to the optimization process, such as reducing late deliveries that could make customers unsatisfied and better exploiting the space inside the warehouse by carefully forecasting what to store and what not. Some of the most efficient practices are:

- **Warehouse slotting optimization**, which aims at establishing the best place inside the inventory for each product, allowing faster identification of items which are ready to be distributed. The physical space in the warehouse is optimized by ensuring things are clearly labeled, and items are stocked in order of popularity to make it easy for pickers to find items that are frequently ordered. Since it usually involves a high quantity of products, it is challenging to perform it manually, thus requiring some software systems able to consider enough room for any warehouse operation like product that remain in storage for long or possible returns.

- **Automation**, achieved through some warehouse management softwares that aim at replacing manual checks of correct products reception and shipment. It is a monetary investment, however it lowers the risks related to human errors, by providing automated barcode scanning of goods then uploaded to a server.

- **Shipping process optimization**, with not only the purpose of delivering goods to customers in time, but also that of making sure that they are distributed to the right ones. Through routing automation it could be possible to reduce errors and delays, also by choosing the most suitable shipping carriers by delivery time or costs.

- **Inventory replenishment optimization**, obtained through careful analysis of the available historical data about the customers requests for specific items and forecasting of the demand. This makes the whole process less time-consuming and helps to foresee what will be the quantities needed in specific future instants. Moreover, it is essential to balance the availability of stock quantities to fulfil the demand with the need to prevent overstocking that inevitably leads to higher holding costs.

1.2.3 Difference between Warehouse Management, Inventory Management and Stock Management

Terms like warehouse management, inventory management and stock management are often used interchangeably, but there are some differences.

Warehouse management is a broad term that includes different aspects of warehouse operations, like warehouse organization and design, labor, order fulfillment, warehouse monitoring and reporting.

Inventory management is centered on efficiently ordering, storing, moving, and picking the materials needed to make products or fulfill orders.

Stock management is often used as another term for inventory management, but it is possible to point out the difference between the two, especially for companies involved in manufacturing products. Stock generally refers to finished product ready for sale or distribution, while inventory includes everything in the warehouse: raw materials, materials that are in the process of being built into products and finished products (stock). Stock management is therefore a subset of inventory management that focuses specifically on holding as little stock as possible to save space and costs while still being able to meet customer demand, which is the purpose of this study. For simplicity the two definitions will be used as synonyms.

1.3 Inventory Optimization

Inventory or stock optimization is a supply-chain management method that attempts to remove excess inventory while maintaining the right amount of goods in stock to meet consumer demand and revenue goals, thus balancing demand and supply.

1.3.1 Inventory Optimization Challenges

Inventory is the largest single asset that most companies have. Also Wall Street looks attentively at their inventory handling when making evaluations about businesses' performances. Inventory consumes space, gets damaged, and sometimes becomes obsolete and carrying surplus inventory costs the organisation. Studies show that 70% of correlation exists between overall profits and inventory turns, however a high number of companies still use traditional strategies to cope with customers demand. These consist in companies purchasing high quantities of products to cope with demand spikes, while today inventory optimization is considered a crucial strategy to save working capital by reducing stock quantities without damaging operations and sales. The reasons why it has become a central point are explained by some organisations joining the APQC Open Standards Benchmarking in logistics: they show that more than one-third of their logistics labor is allocated to operating warehouses and keeping the optimal amount of inventory on hand could make a significant difference. In addition, inventory carrying costs can be non-negligible for organizations. Figure 1.4 shows APQC data for inventory carrying costs in 2011, pointing out that median companies spend 10 percent of the annual value of their inventory to carry it. Bottom performers, i.e. those organizations at or below the bottom quartile of performance spend almost 10 percent more than top performers (i.e., organizations at or above the top quartile of performance) to carry their inventory: 15.0 percent compared to 5.8 percent of the inventory value.

Managing inventories has become more and more important also after 2001, when network giant Cisco announced $2.1 billion inventory write down, while in 2006 Wal-Mart began its Inventory Deload Program, after finding that total inventory levels had been rising at a much higher rate with respect to the company's sales growth (almost 90%).

As a matter of fact, an effective inventory optimization program should invest on good design, effective processes, appropriate technology, and regular assessment.

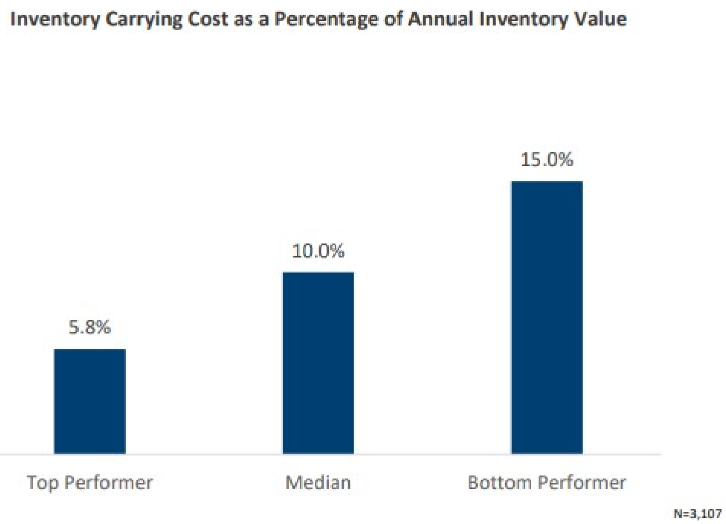


Figure 1.4: Percentage of Inventory Holding Costs [4]

1.3.2 Inventory Optimization Benefits

Inventory optimization can lead to significant economic benefits. IDC Manufacturing Insights, for instance, reported that many organizations were able to reduce inventory levels by up to 25% in one year. In particular, they give an assessment of providers participating in the worldwide supply chain inventory optimization market.

BP Castrol, for instance, which is a manufacturer, distributor, and marketer in the automotive field, turned to inventory optimization because slow moving products had excess inventory while fast moving products were often out-of-stock. Safety stocks were set manually and infrequently adjusted, based largely on personal experience, and in the calculations there was little formal sense of supply and demand uncertainty. They improved demand sensing by generating more robust and reliable forecasts, improving monitoring and safety stocks. In this way BP Castrol reduced inventory levels by 35% in two years and at the same time increased customer satisfaction by 9%.

Also Smiths Medical, a leading supplier of specialized medical devices and equipment for global markets, was able to reduce overstocks and unsatisfied demands exploiting inventory optimization strategies, improving levels of stock availability and, consequently, profits.

1.3.3 Inventory Optimization Characteristics

Inventory optimization models can be classified as deterministic or stochastic. **Deterministic models** are based on the assumption that all parameters and variables associated with an inventory stock are known and that there is no uncertainty associated neither with demand nor with the replenishment of inventory stock. Demand is thus assumed to be known and either static (i.e. constant over an infinite horizon) or dynamic. **Stochastic models**, (or Probabilistic) on the contrary, take into account the more realistic assumption that demand is a random variable that can be forecast or whose distribution may be known. In this case demand can be stationary or not. In this kind of models there can be other sources of uncertainty besides the demand, such as the lead time for the shipment of a product.

Another distinction is made by considering the time horizon of the optimization. For stochastic models it is possible to perform it considering a **Single Stage optimization**, thus minimizing the inventory in just one time window, or through **Multi Stage optimization** that performs minimization over several time periods.

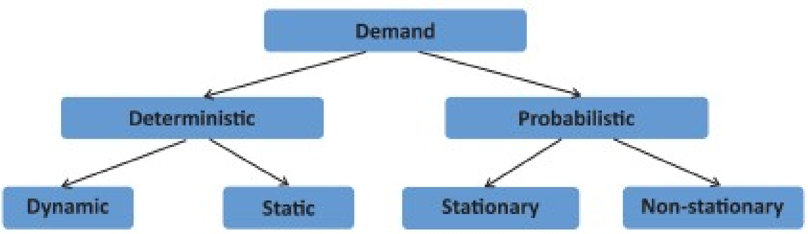


Figure 1.5: Types of demand classification
[5]

1.3.4 Spare Parts Inventory Optimization

Spare parts are common inventory stock items, which are needed to maintain equipment and reduce the consequences of its downtime. According to Gallagher et al. [6] machinery annually consumes spare parts amounting to as much as 2.5% of the purchase price and they might have a useful life of up to 30 years. If spare parts result to be not available when maintenance or repair of certain equipment are needed, businesses may encounter some economic losses. As a consequence, a sufficient availability of components must be ensured through spare parts management, while keeping the costs as low as possible.

Due to some peculiar characteristics of this kind of products, spare parts management can be considered as a special case of inventory stock management:

- **Intermittent demand patterns** that make spare parts demand difficult to forecast due to periods of zero demand followed by others of non-zero one [7].
- **Heterogeneity**, of spare parts and their high number make difficult for companies to handle them inside their inventories. If some enterprises want to devise specific stock control strategies for each kind of spare part, this is made particularly challenging.
- **Obsolescence** of some spare parts, that make necessary the minimization of stocks, with only small quantities per Stock Keeping Units (SKU), to avoid the risks of excessive holding costs.
- **Dependence on equipment usage**, meaning a close relationship between spare parts consumption and maintenance. Thus the usage patterns of equipment and their maintenance strategies are important factors in designing spare parts optimization.[8][9]

For all these reasons, it is crucial to use forecasting strategies together with past data analysis to understand product variability, exploiting previously recorded demand. A good understanding of which products need to be stocked, in what quantities and across what time intervals, is also essential: it helps to understand the safety stock calculations to address sudden fluctuations in demand. Moreover, replenishment strategies are important to understand which quantities are needed to be reordered at what points in time. Also, it is necessary to keep track of the goods which are in transit and not just those which are in stock at the warehouse.

1.3.5 Periodic vs Continuous Review Policies

When taking into account this kind of demand uncertainty and replenishment strategies, different inventory control policies can be analysed, since using the right one can add a lot of value to help in inventory optimization efforts. They can be classified in Periodic Review policies and Continuous Review policies.

- **Periodic Review Policies**: this system is adopted when managers order products at the same time each period. The most common are *(R,Q)*, *(R,S)* and *(R,s,S)* where R indicates the periodicity of the review. Every R time units either a fixed amount Q of items or a variable quantity sufficient to reach the target stock level S is ordered. In (R,s,S) systems instead an order is only performed if the stock position is equal to a threshold s or below.

- **Continuous Review Policies**: in this type of system inventory is continuously monitored. Some common policies are *(s,Q)* and *(s,S)*: whenever the stock position is less than or equal to s, an order of Q units is placed or a variable quantity sufficient to raise the stock level to position S.

1.3.6 Sources of Cost in Inventory Optimization

The kind of policies analysed in Section 1.3.5 can be driven by four sources of costs: ordering, holding, shortage and spoilage.

Ordering cost is assumed to be fixed and does not depend on the number of ordered units, while holding cost represents the the amount of rent a business pays for the storage area where they hold the inventory. These two costs are associated with operational aspects and can be estimated up to a certain level of accuracy.

Shortage costs, also known as stock-out costs, are related with the extra costs incurred when a product demand is not satisfied, causing a delay due to temporary unavailability. Some of the reasons might be emergency shipment costs or disrupted production costs. Spoilage costs are due to the fact that perishable inventory stock can rot or spoil if not sold in time, so controlling inventory to prevent spoilage is essential. These last two costs are more difficult to be estimated.

Chapter 2

Related Literature

2.1 General Analysis

In this section we provide a review of relevant literature considering some difficulties arising from inventory management, such as uncertain demand and lead time, together with some methodologies that cope with them.

Unknown demand complicates inventory management problems. For this reason, it is essential to provide demand information in advance to help controlling stock quantities in the best way possible. Nakade et al. [10] emphasise how production and inventory control using advance demand information (ADI) decrease the amount of products in inventory and backlogs, pointing out the importance of the exploitation of preliminary demand insights. In this context, ADI refers to the piece of information related to the demand that managers get before it actually occurs.

When demand is uncertain and difficult to forecast, ordering the right quantities of product at the right time is essential. Samak-Kulkarni et al. [11] analyse several models to minimize the total annual inventory costs for different items, considering the nature of the demand according to different thresholds for its coefficient of variation. Their study takes into account 288 different items and to simplify the control over the inventory they perform an Always Better Control (ABC) analysis to classify them into three categories according to their usage level (high, medium, low). The first model they analyse is the lot for lot: the system uses the exact shortage quantity (requirement minus available stock) as the order quantity in the case of a material shortage. In this case the holding costs are zero, but this causes shortage costs to rise. The second model is the Economic Order Quantity (EOQ), which is the order quantity that minimizes total inventory holding costs. However, one of the important limitations of the economic order quantity is that it assumes the demand for the company's products is constant over time. The Period Order

Quantity lot-size rule is the third model, and it is based on the same theory as the economic-order quantity. It uses the EOQ formula to calculate an economic time between orders. This is calculated by dividing the EOQ by the demand rate. This produces a time interval for which orders are placed. Instead of ordering the same quantity (EOQ), orders are placed to satisfy requirements for the calculated time interval. The number of orders placed in a year is the same as for an economic order quantity, but the amount ordered each time varies. Thus, the ordering cost is the same but, because the order quantities are determined by actual demand, the holding cost is reduced. The best performing model is however the Wagner-Whitin algorithm that determines the optimal batch size for a product with a dynamic demand considering also capacity constraints.

The work from Axsater [12] analyses one of the most common inventory control problems under normally distributed demand: the purpose is that of minimizing holding and ordering costs of a single-echelon inventory system considering a fill rate constraint. The study aims at determining the reorder point and the order quantity by using an (R,Q) policy, so that the total expected costs are minimized under the fill rate constraint. In this case, the fill rate is the amount of customer demand that can be satisfied with the immediately available stocks, and it can be empirically measured as the average of the correctly served requests with respect to the total number of them. It is important when controlling inventories as it represent the amount of demand that could be satisfied by improving the management of the inventory itself. Two procedures are proposed: one is based on a two-step approach in which the order quantity is estimated through an Operational Research (OR) deterministic model; the other one reformulates the problem for a given fill rate value so that it reduces to a single parameter, thus solving it exploring a grid of values of this parameter and a certain amount of fill rates. At the end, a table for all items is obtained and the single item solutions result using a linear or polynomial interpolation in the table. Results show that this solution is very close to the optimal one, even if savings appear to be larger for low service levels and smaller for high service levels.

Another work focusing on specific optimization approaches to cope with stationary uncertain demand is the one from Perez et al. [13]. It presents an inventory management problem for a make-to-order supply chain, with inventory holding locations at each node. Its aim is that of thus dealing with complex supply chain networks with many distribution points from the point-of-origin to the endpoint, the consumer. It also considers heterogeneous lead times between nodes. In particular, a retailer is subject to an uncertain stationary consumer demand for a single product at each time period, and two sales scenarios are taken into account, i.e. backlogging and lost sales. The daily inventory replenishment requests are modeled and optimized using deterministic linear programming, multi-stage stochastic linear programming, and reinforcement learning, and the obtained performances are compared in terms of both profits and service levels. The deterministic model (DLP) uses either the rolling horizon or shrinking horizon technique to determine optimal re-order quantities for each time period at each node in the supply network, and the demand is modeled as the expected value over the time window. The multi-stage stochastic program (MSSP) exploits a scenario tree and both the shrinking and rolling horizon to choose the optimal reorder quantity at each stage. Finally, the reinforcement learning model (RL) makes re-order decisions according to the overall network state. Even though the application of the work is slightly different from our case study, it is interesting to analyse the outcomes of the different strategies: of the three approaches, stochastic modeling yields the largest increase in profit, whereas reinforcement learning creates more balanced inventory policies that would potentially respond well to network disruptions. However, the deterministic model performs also well in determining reorder policies in a dynamic way, leading to comparable results with respect to reinforcement learning. The average value of lost sales for all the approaches in a single node are summarized in Figure 2.1.

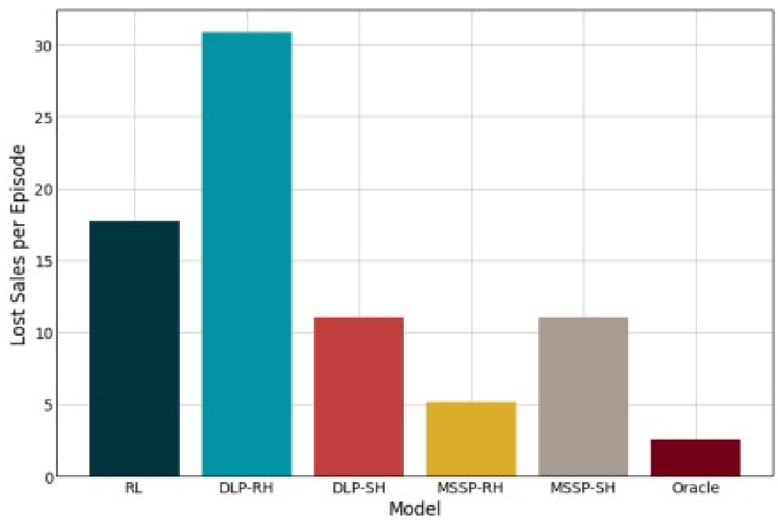


Figure 2.1: Average unfulfilled demand at a single node [13]

In the previously mentioned papers the stochastic demand was stationary. However, several cases can be found in the literature analysing non-stationary demand problems. Armagan Tarim et al. [14] address the single-item non-stationary stochastic demand inventory control problem under a variation of the (R,S) policy. If demand were stationary, the optimal inventory replenishment policy determined using (R,S) would be cyclical, that is, a sufficient amount of order would be placed every fixed number of R periods to raise the inventory position to the level

S. However, under non-stationary stochastic demand assumption this a difficult problem where the two control parameters are non stationary. The actual demand per period is assumed as normally distributed random variable considering forecast values. By means of a piecewise linear approximation to the non-linear cost function, a certainty equivalent mixed integer programming model is built. The accuracy of the approximation can be improved by introducing new breakpoints—each adding a new set of constraints without requiring additional variables—to the piecewise linear approximation. The resultant MIP model gives the approximately optimal solution in terms of the number and timing of the replenishments and the associated order-up-to-levels. Using these (R,S) policy parameters and observing the realized demand, the size of the actual replenishment orders for the periods when stock reviews take place are determined.

Another study that addresses the non-stationarity of the demand, with the additional complexity of the perishability of the managed good, is by Gitae Kim et al. [15]. They address a common single-echelon single-period inventory control problem, i.e. the newsvendor model, extending it to a multi-period one. Instead of minimizing the inventory, the model maximizes the expected profit considering the trade-off between remaining newspapers costs and shortage costs under demand uncertainty. They provide novel knowledge to optimize the trade-off between delivery, transshipment, shortage, and holding costs. In addition, the model supports decision making for allocating inventory of short-life-cycle products under the uncertainty. They first propose the mathematical model for multi-period newsvendor problem with transshipment and non-stationary demand and then develop a multi-stage stochastic programming model to optimize the inventory control policy. The assumption of a non-stationary demand is due to the fact that a stationary one may not be valid in practice because of economic conditions and seasonal effects of the newspapers requests. The uncertainty of data is described by a probability distribution and gradually revealed over the stages in form of a set of scenarios, indicating the possible realization of each random event. The progressive hedging method (Figure 2.2) is used to solve the problem, aiming at solving the multistage stochastic program through scenario decomposition. The experimental results show that the proposed multi-stage stochastic programming model performs better than the EOQ and single-period newsvendor models. Furthermore, this work also provides an interesting sensitivity analysis of the relationships among the initial inventory, the holding, delivery, transshipment and shortage costs, showing that as the holding costs increase, also the delivery and transshipment ones do, while the shortage is stable (Figure 2.3).

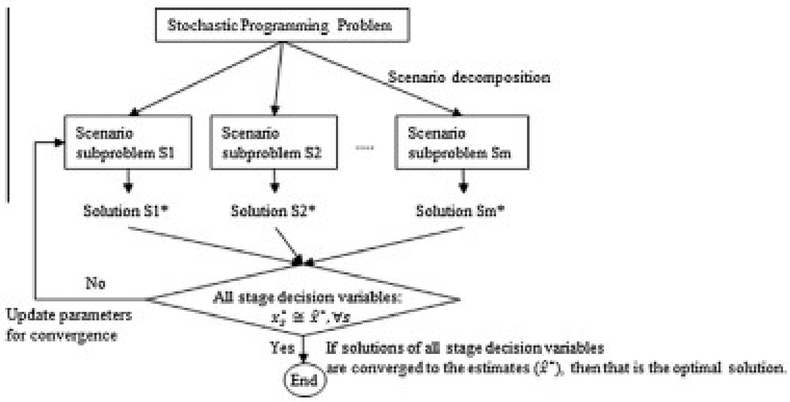


Figure 2.2: Progressive Hedging method
[15]

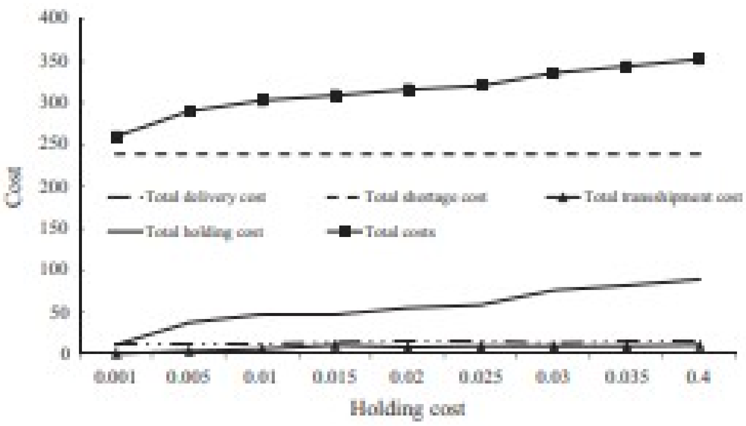


Figure 2.3: Sensitivity analysis of the holding cost
[15]

The perishability of products while minimizing inventory is also considered in Dillon et al. work [16]. The purpose is that of managing inventories in the blood supply chain while minimizing the operational costs and blood shortage or wastage due to outdating, taking into account demand uncertainty. This latter is modeled by considering a general stochastic process that can be solved by an off-the-shelf optimization software like CPLEX. Eight type of bloods are considered, together with the average and standard deviation of their daily demand. The adopted policy is an (R,S) one, where R is the optimal time between reviews and S is the target inventory level used as a reference for defining order quantities. A two-stage stochastic model with recourse is formalized, where the first stage decisions to be made before the uncertainty realization are the R and S parameters, while the second stage ones that are made after the uncertainty disclosure are those referring to the daily operations of the system under each scenario of the model. The resulting model is a Mixed-Integer Linear Program (MILP) representing the deterministic equivalent formulation of the two stage stochastic one. For their case study, after the definition of the demand distribution, 10 demand scenarios are generated for three months and each blood type using a Monte Carlo sampling approach and the distribution is assumed to be identically and independently distributed. To evaluate results, the authors suggest four key performance indicators (KPI), i.e. total cost, because of the cost-oriented management approach that aims at minimizing inventory costs, daily average outdate and average unmet demand, measured in number of units, average age at issue, corresponding to the average age of units transfused in a day, and average inventory on-hand per day. Outcomes are reported in Figure 2.4 for one of the experiments with service level equal to 95% and outdate limiti equal to 5%, and they show that the optimised policy presents improvement of up to 50%.

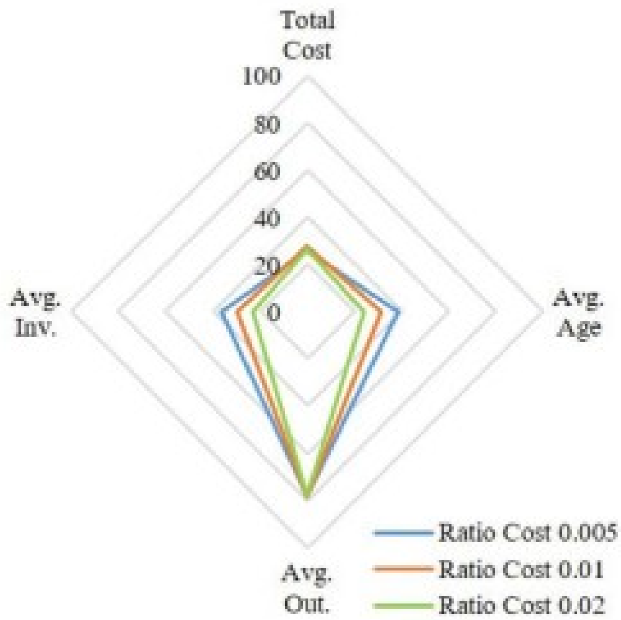


Figure 2.4: Optimised policy with respect to benchmark actual hospital policy in %

[16]

In many cases demand is not the only source of uncertainty, as Kulkarni et al. discuss in [17]. They consider a production-inventory system where the production and demand rates are modulated by a finite state Continuous Time Markov Chain

(CTMC). The exploited order policy involves the placing of an order of size Q as soon as the inventory position falls below a reorder point R, while taking into account stochastic i.i.d. lead times following an exponential distribution. They derive the distribution of the inventory level and study also the sensitivity of the system with respect to other lead times distributions.

2.2 Inventory Optimization of Spare Parts

Spare parts and their corresponding demand are a complex subset of inventory items to cope with when speaking of inventory optimization, because of the high number of managed parts, their intermittent patterns and the risk of stock obsolescence. In industrial contexts the proportion of the stock range that is devoted to spare parts is often considerable as they constitute up to 60% of the total stock value. Thus, small improvements in their management may be translated to substantial cost savings [18].

As customer expectations rise and product complexity increases, an enormous quantity and variety of spare parts is stocked. At the same time, product life-cycles are becoming shorter due to rapid technology advancement, leading to the need of carefully analysing the characteristics of the spare parts in each phase of this cycle, taking into account eventual obsolescence. Hu et al. in [19] provide an analysis of different OR strategies and an overview of spare parts management, pointing out an innovative framework reported in Figure 2.5. Under these conditions advanced inventory management strategies can efficiently reduce the inventory cost while achieving satisfactory service levels.

Li et al. [20] propose a stochastic programming model for the supply chain planning of maintenance, repair and operation (MRO) spare parts (Figure 2.6), becoming more and more relevant for manufacturing enterprises. To do so, they take into account the fact that randomness and uncertainty in storage and production must be correctly modeled because of the specificity of this item category, ensuring availability of components and a minimization of costs. In particular, the focus of their work is that of minimizing production, setup, storage and distribution costs ensuring a good service level. Considering a multi-product multi-period maintenance, they build a model of stochastic programming combined with multi-choice programming, in which the decision maker can model uncertain parameters as random variables and dynamically set multiple choices for some constraint parameters according to the situation. The Lagrange interpolating polynomial approach is used to derive the deterministic equivalent of the model and the equivalent non-linear mixed integer programming model is solved. The results of the numerical examples show the efficiency of the model, through some testing with continuous caster bearings. Moreover, with the aid of the technologies from

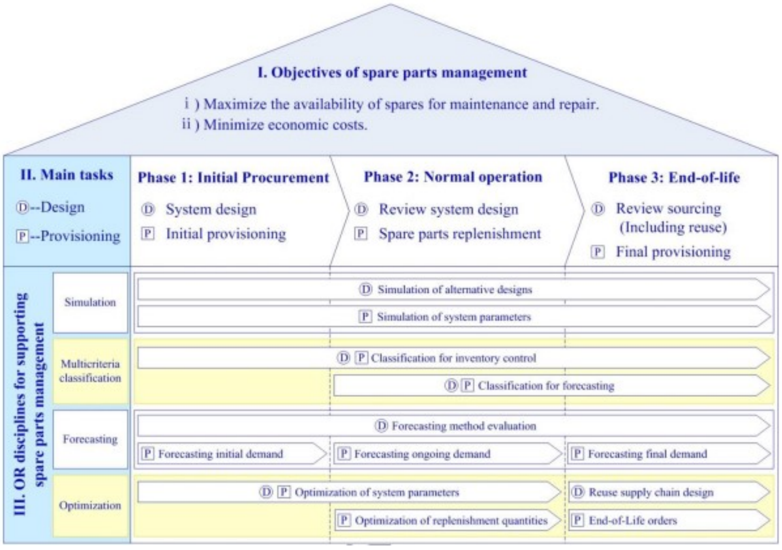


Figure 2.5: Framework for spare parts management [19]

the Internet of Things, mobile Internet and Big Data, information on product operations and user feedback could also be collected to predict the spare parts demand in a more efficient way. The results of the numerical examples validate the feasibility and efficiency of the proposed model. Finally, the model is tested in the supply chain planning of continuous caster (CC) bearings.

Another work by Zhang et al. [21] develops a procedure for establishing stocking rules for a multi-component distribution center that supplies spare parts for an equipment maintenance operation. The aim is that of minimizing inventory costs taking into account service level and replenishment frequency, with particular focus on a preliminary classification of spare parts according to an improved ABC methodology scheme. The latter is more effective since it is sensitive to key attributes of the spare parts with respect to the optimization model. Within each category, they constrain service and order frequency uniformly, and then use various approximations to compute stocking parameters, thus applying heuristics to the (Q,R) policy. Numerical results show that the ABC classification does not introduce large errors if it reflects the model key parameters and that the implemented heuristics for the stocking parameters can be used effectively, as long as the exact formula for service level is employed to adjust the formulas to achieve target performance. These heuristics for setting stocking parameters in multi-item spare parts distribution centers are obtained by approximation of the inventory and service expression to get closed-form equations for both the order quantity and the replenishment points. Moreover, having the different spare parts

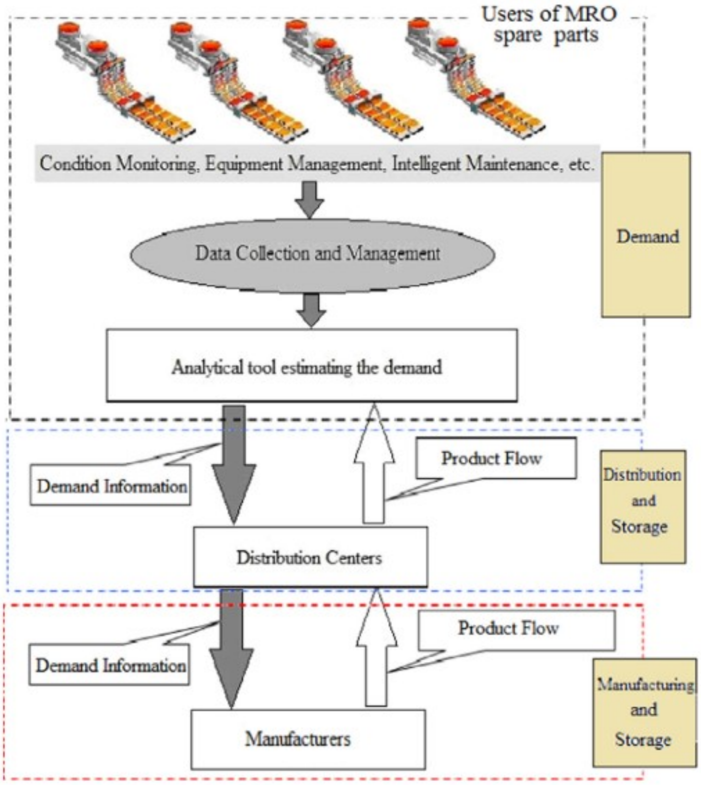


Figure 2.6: Supply chain operations for MRO spare parts
[20]

classification, a uniform service target can be assigned to each part within a given category, reducing the size of the optimization problem. Finally, the exact service is computed for any set of stocking parameters and it is used to guide the search for target service levels useful to achieve the desired one.

Spare parts demand can be indeed non-stationary, especially when considering growth of installed base and new sales. For installed base we refer in particular to a measure of the number of units of a type of product that have been sold and are being used. In this eventuality, due to the challenging task, spare parts inventory control strategies are crucial for coping with the resulting non-stationary behaviour of the demand. The goal is to ensure that timely replacements can be provided to customers while minimizing the overall cost for spare parts inventory control. Jin et al. [22] provide a solution as a non-linear integer program for the aggregate maintenance demand of a product whose installed base grows as a homogeneous Poisson process (Figure 2.7). As the products failures (maintenance) follow an exponential distribution, their mean and variance are derived and through a bisectional search algorithm the optimal setting for the chosen dynamic (Q, R) policy are found. Finally, through a simulation approach, the application of the method in controlling spare parts inventory under service level constraints is carried out, showing effective results also for products with generic distributions of the demand.

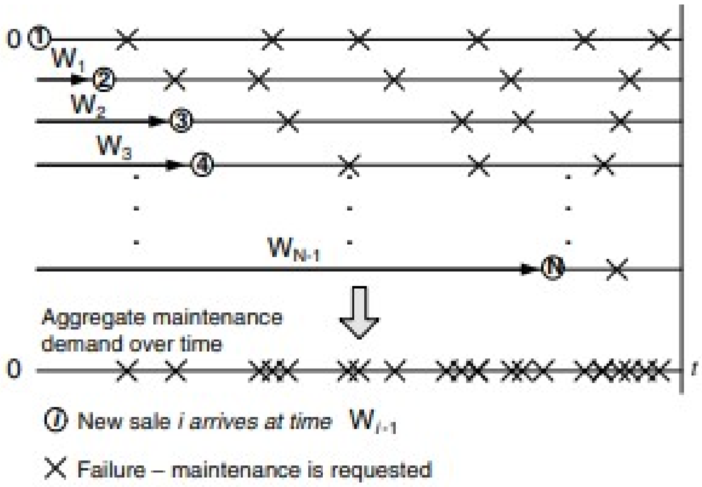


Figure 2.7: Maintenance demand as a Poisson process
[22]

As a matter of fact, Liao et al. [23] present a work involving a non-repairable product whose sale rate is assumed constant and its failure time follows a Weibull distribution. They formulate a mathematical model based on dynamic (Q, R) policy with fixed order quantity and replenishment point, and the problem is solved using a multi-resolution approach. Finally, though numerical examples, the inventory cost is minimized taking into account service level constraints.

Another work that addresses inventory control of spare parts is the one by Aronis et al. [24]. They focus on a specific case study in which a company produces circuit packs as spare parts for telephone switching systems and wants to keep a high service level for its customers. Continuous operation of these systems is essential for the company customers, and when a failure in a circuit pack occurs, it has to be solved by replacing the failed circuit pack with a functioning one, already available. As a consequence, the demand is originated by random failures of the circuit packs. The company under study sees the spare parts stock as an insurance against unexpected events. In this regard, there is uncertainty both with respect to the average failure rate and with respect to the usual fluctuations given an average failure rate, and both uncertainties are addressed separately. The Bayesian method proposed by the authors, instead, tackles them in an integral way and gives a better indication of which service level one may finally expect. This approach is applied to forecas the demand for spare parts, with the objective of more accurate determination of stock levels required to provide a negotiated service level to the users of the equipment. The inventory control policy is a (S-1,S) and it is the same already adopted by the company, however the determination of the appropriate parameters for the policy through a Bayesian approach introduces lower total stocks for the same desired service level.

2.3 Maintenance Forecast

Spare parts demand uncertainty results to be a focal point for all kind of inventory optimization strategies. As a matter of fact, to help the management of the stock quantities and of the customers requests, demand forecast is indeed necessary in certain cases, and when speaking of vehicles' spare parts it is obviously strictly related to the concept of maintenance prediction.

Maintenance prediction can be carried out considering several factors. Perrotta et al. [25] analyse several elements that could influence fuel consumption in trucks and as a consequence its prediction over time, based on truck telematic and road geometry and condition data. Truck data come from sensors installed in recent trucks, and through Machine Learning regression methods the fuel consumption of a large fleet is predicted. In particular, the dataset includes 56 variables, and in order to avoid overfitting only the most significant parameters are kept for the regression analysis. In addition to that, also cross-validation is performed, splitting the data into training and test datasets of 75% and 25% each. The employed algorithms are Support Vector Machine (SVM), Random Forest (RF) and Artificial Neural Network (ANN), and the performances are evaluated through root mean squared error and mean absolute error and then compared. Results show that RF is the technique giving the best performance, even though SVM and ANN present better result in predicting extreme values.

Another work by Rezvanizaniani et al. [26] shows an effective predictive maintenance technique for industrial assests, exploiting a dataset provided by the Prognostic and Health Management Society 2014 Data Challenge. It contains usage and part consumption for three years, two of which are used as training set and the remaining part as test. The the method in built on the probability of failure risk, and its aim is that of establishing the high risk and low risk times of failure for the test data. One of the main challenges is to detect the Preventive Maintenance (PM) in the trainig data, whose pattern based on time and type of maintenance must be detected in the first place. The high-risk time intervals are successively determined through the frequency of the failures at specific times between each preventive maintenance. Finally, the high risk time intervals are predicted also for the testing data. The preliminary processes of the data analysis included the removal of outliers and feature extraction. Pre-Processing functions, included data observation, sorting and irregularities in the usage data, were used to eliminate data that does not have the quality required to accurately determine features. The concept of a bathtub curve was used to extract and select features on how soon after a maintenance a failure would occur during the training data, which was then applied to the testing data to provide an accurate health assessment rating and therefore determine if the asset was at high risk. The other approach is to use the corrective maintenances to evaluate the correlation between corrective measures and the probability of failure afterword. Using the number of failures immediately after maintenance actions in the testing data, the probability of failure after maintenance actions is calculated.

2.4 Previous publications by our research group

The study for this thesis derives from the broader work carried out by the SmartData@Polito research group together with Tierra S.p.A. The former is a center focusing on Big Data technologies, Data Science and Machine Learning approaches, applied to different domains with the aim of providing efficient solutions both to theoretical problems and for helping companies towards applications. The latter, instead, is a company operating in the IoT sector providing telematics solutions for complex issues in terms of management, maintenance and remote diagnostic of equipment. In particular, the domain of study of the following works is related to identification and forecasting of industrial vehicles' usage patterns together with next-maintenance prediction.

One work by Buccafusco et al. [27] aims at automatically define per-vehicle duty levels for industrial vehicles working on construction sites. These duty levels indicate the vehicle state according to how much it has been used, and they are usually set up manually due to the high heterogeneity of usage patterns of the vehicles under analysis, that can change from light to heavy workload. However, this manual strategy is time-consuming and likely to experience errors. To overcome this issues, the authors propose a clustering-based approach to the data collected from a CAN bus data logger installed on a test farm tractor. CAN bus data usually consist of raw time series, sampled, aggregated and transmitted to a central repository, and are mainly related to fuel consumption, route characteristics and vehicle movements. In order to optimize maintenance, the ideal indicators to monitor vehicle usage is the time spent in specific duties, which describe the current state of a vehicle and are classified as long idle, idle, moving/working, light workload and heavy workload. The clustering approach works on a list of 20 Suspect Parameters Numbers (SPNs) acquired from November 7, 2019 to April 15, 2020, which escribe several aspects like the engine speed, fuel rate and so on. First, raw CAN data are cleaned, removing missing data or errors, then after having identified working cycles, the asynchronous CAN messages are aligned and synchronised through linear interpolation and down-sampling, and the most significant and influent SPNs are selected through correlation analysis. At this point, SPN series can be synchronised and segmented into fixed-length intervals, where each segment is described by a specific feature. Finally, the segments are clustered to support the definition of vehicle-specific duty levels, and homogeneous groups are obtained. Results show the effectiveness and truthfulness of the approach, as the number of segments per cluster distribution (Figure 2.8) corresponds to the actual usage of the employed vehicles.

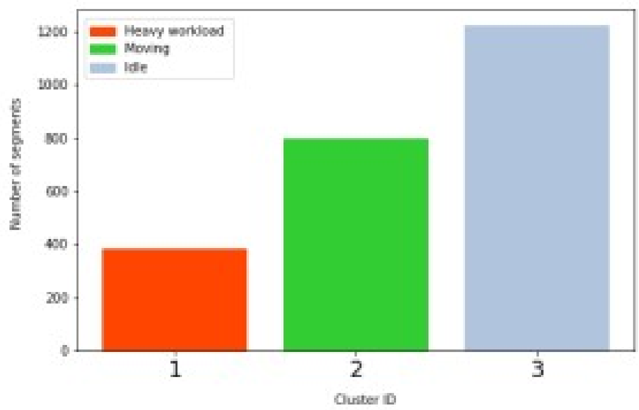


Figure 2.8: Number of segments per cluster
[27]

Another paper that exploits CAN bus data analysis is the one from Markudova et al. [28]. Starting from industrial vehicles' usage indicators like utilization hours, they derive non-stationary time series, and through Machine Learining methods they exploit these historical data and additional features (which describe several vehicles characteristics) to make predictions on future usage. In particular, the aim of this work is that of predicting the daily utilization hours of different models of construction vehicles, exploiting real data collected over 4 years and involving 2239 vehicles over the world. This prediction has the objective of easing maintenance scheduling for site managers. Starting from the collected data, they are prepared for the Machine Learning process though cleaning, normalization, aggregation of the features on a daily basis, enrichment with contextual information and transformation, to tailor input data to a relational data format. Furthermore, vehicles are classified based on the type of construction vehicle and each type is split into several models. Since it is important to discover similarities among the usage patterns, the Cumulative Distribution Functions (CDF) of the number of daily utilization hours per vehicle type are derived. To perform prediction on utilization hours, they train regression models on past vehicle data separately on each vehicle, due to the heterogeneous trend in the usage patterns. The training is performed using both sliding window and expanding window approaches. In addition to that, before proceeding to the prediction step, a statistics-based approach is used to filter relevant and most discriminating features. Finally, some regression algorithms are implemented to perform prediction, such as Linear Regression, Lasso Regression; Support Vector Regression and Gradient Boosting. Results show that the models are effective, with only 15% error in predicting next working day utilization hours.

Mishra et al. in [29] focus instead on the prediction of the remaining time to maintenance of industrial vehicles. This work describes a data-driven application to provide an automatic scheduling of periodic maintenance operations of industrial and construction vehicles, whose schedule is highly influenced by per-vehicle characteristics and usage. Also in this case, the presence of on-board devices allows to monitor several features of the vehicles. For a given vehicles, the purpose is that of implementing regression techniques to predict the remaining days until next maintenance, coping with typical difficulties of working with this kind of vehicles: non-stationarity of utilization time series per vehicle, lack of historical data for newly-added vehicles to the fleet, and vehicle heterogeneity in terms of number, type and frequency of maintenance operations, that make the process time-consuming. All these challenges are faced training a separate regression model per vehicle, which analyses single vehicle usage patterns and time to maintenance, incorporating historical usage levels in the predictive models training linear and non-linear ones, and combining the regressor outcomes obtained on similar vehicles. In particular, different methodologies are issued when vehicles are indicated as new, semi-new or old, due to the different amount of available data. Results from all categories are obtained using LR, SVR, RF and Histogram Based Gradient Boosting (XGB), showing that non-linear algorithms are those performing best.

On a similar track to [29], in a paper from Markudova et al. [30], the research group addresses the learning of per-vehicle predictors to forecast the next-day utilization level (indicated as task A) together with the remaining time until next maintenance (task B). The input data are related to a set of about 2000 construction vehicles of different types. The previous assumptions about the challenges in dealing with large fleets of industrial and construction vehicles hold, leading to the use of data-driven solutions based on Machine Learning techniques able to model the trends of non-stationary patterns of use. For each vehicle category, the two tasks are addressed using univariate but also multivariate models, where the prediction model is enriched with additional contextual features describing different usage patterns. The regression models training is carried out using a portion of the historical data denoted as training window, exploiting both a sliding and an expanding window strategy. Figure 2.9 provides an insight on the parameter tuning for task A and B on old vehicles, in order to find the best configuration. The employed algorithms for regression are LR, SVR, RF and GB. Results show that LR performs better than baseline methods even if it is affected by the usage pattern variability of the vehicles, while RF and GB are good in predicting single vehicle maintenances.

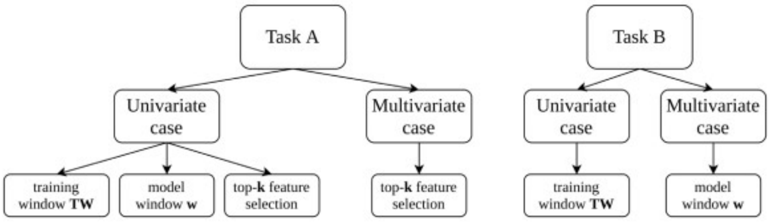


Figure 2.9: Tuned parameters for each task
[30]

2.5 Our Contribution

This thesis project places itself in the literature related to industrial vehicles' spare parts. In particular, starting from OR inventory optimization strategies, it sets the goal of combining MP models with stationary demand with specific inventory control policies that aim at establishing a target equilibrium level for minimizing inventory costs and ensuring customers satisfaction. In addition to that, it also investigates the case of non-stationary demand, exploiting ML techniques to provide forecast of the next maintenance time.

Chapter 3

The Input Dataset

The choice and processing of the dataset play a central role in satisfying the objective of this work. For an effective optimization of the inventory quantities, it is necessary to derive the demand distribution for the required spare parts over different time instants. In order to obtain this kind of input, the main requirements are related to two aspects:

In terms of vehicles, a significant number of them must be available to collect various information. Moreover, historical records about their utilization should be disposable over a sufficiently large time period.

In terms of spare parts, it is necessary to obtain historical maintenance information in order to understand how many tasks are performed and which specific component is substituted according to a specific deadline.

For these reasons, data are gathered from different sources and are then processed and put together in order to satisfy the aforementioned requirements. Information regarding the vehicles utilization over time are obtained thanks to the combined work with a company providing telematics services to industrial vehicles producers, that managed to collect CAN bus data about their usage, such as the traveled distance. Data related to the past maintenance details are instead obtained thanks to the maintenance booklets provided by the company.

3.1 Vehicles' Choice

The input dataset combines information related to a large company producing vehicles for good transportation (trucks) and people transportation (buses) operating in many countries worldwide.

The vehicle choice for the dataset results appropriate since trucks and buses are on-road vehicles, able to travel many kilometers daily, thus providing a great number of usage records. Moreover, the provided dataset contains a sufficiently large number of vehicles and models.

3.2 Characterization of the Dataset

The input data are retrieved from different sources, like CAN bus technology that enables the acquisition, collection, and processing of vehicle usage data, but also from companies historical records about specific maintenance tasks that have been performed over time.

The first dataset of interest is related to the vehicles maintenance history. It contains maintenance information from January 20, 2020 to January 31, 2022, resulting in 8242 rows of data. The most relevant information are related to:

- **Maintenance tasks**: for each vehicle that undergoes maintenance it is possible to check the specific task that has been performed, such as change of the engine oil or substitution of the fuel filter. The task names are associated to a univocal task id and also the number of performed tasks for each maintenance are indicated.

- **Maintenance deadlines**: these are thresholds according to which vehicles undergo maintenance. They refer to a specific vehicle model and they can be expressed in terms of traveled kilometers, engine hours or seconds elapsed from the last maintenance. They represent a focus point since each maintenance task is related to a specific deadline.

- **Dealers**: vehicles go to the dealer they visit more often. These dealers are registered according to their names and univocal id, and they are in charge of specifying the date in which the performed maintenance has been confirmed. Each maintenance booking has a specific id and the date in which the dealer confirms that the maintenance has been done.

When the deadlines are reached, maintenance information is saved in some maintenance booklets that keep track of all the performed interventions for each vehicle model. The models names contained in these booklets are directly linked to the ones contained in the maintenance history records.

The second dataset is the one related to vehicles' traveled kilometers. It contains information retrieved from October 1, 2021 to September 30, 2022 related to the cumulative kilometers traveled by 51980 vehicles. This leads to a total of 9712592 rows of data. For each record there is a univocal vehicle id, with the corresponding crossed km up to the specified date. This information is relevant since it allows to check when a specific vehicle reaches a deadline in km, meaning that it is time for it to perform a maintenance task.

Besides the previously mentioned maintenance booklets, in order to put together all the needed information about the vehicles, it is necessary to check also an additional file that contains all the vehicles ids with the correspondent model name, model id and maintenance booklet name. This is needed in order to check exactly which vehicles have specific maintenance deadlines according to their models.

3.3 Dataset Analysis and Data Cleaning

3.3.1 The Maintenance History Dataset

Starting from the maintenance history dataset, it is possible to perform some preliminary analysis. First of all, since it contains performed maintenance information for every kind of task, it is necessary to retrieve only those which refer to actual spare parts substitutions. Table 3.1 lists all the performed tasks in the period of time under study. Among the 12 of them, only 4 refer to spare parts maintenance, i.e. Element Oil Filter, Element Fuel Filter Lower and Upper, Element Air Cleaner Out.

Task name	Task id
Chassis Grease	73
Engine Oil	66
Element Oil Filter	62
Element Fuel Filter Lower	64
Element Fuel Filter Upper	63
Element Air Cleaner Out	65
Grease Wheel Bearing	72
Power steering Oil	71
Brake Fluid	70
Clutch Fluid	69
Differential Oil	68
Transmission Oil	67

Table 3.1: List of maintenance tasks

Moreover, all the maintenance records refer to multiple dealers across different countries, and each vehicle usually undergoes maintenance to the one they visited more often. Thus, to perform an analysis of the requested spare parts by a specific warehouse, it is also possible to filter according to a single garage. Considering each garage, we can count the number of required maintenance over the time horizon, both in terms of spare parts only and of total performed tasks (Figure 3.1). This is useful to have an insight on the dealers for which we have more records as we want to derive a time series of the requested components. Notice that the names of the dealers in the x axis of the figure are not shown for confidentiality reasons.

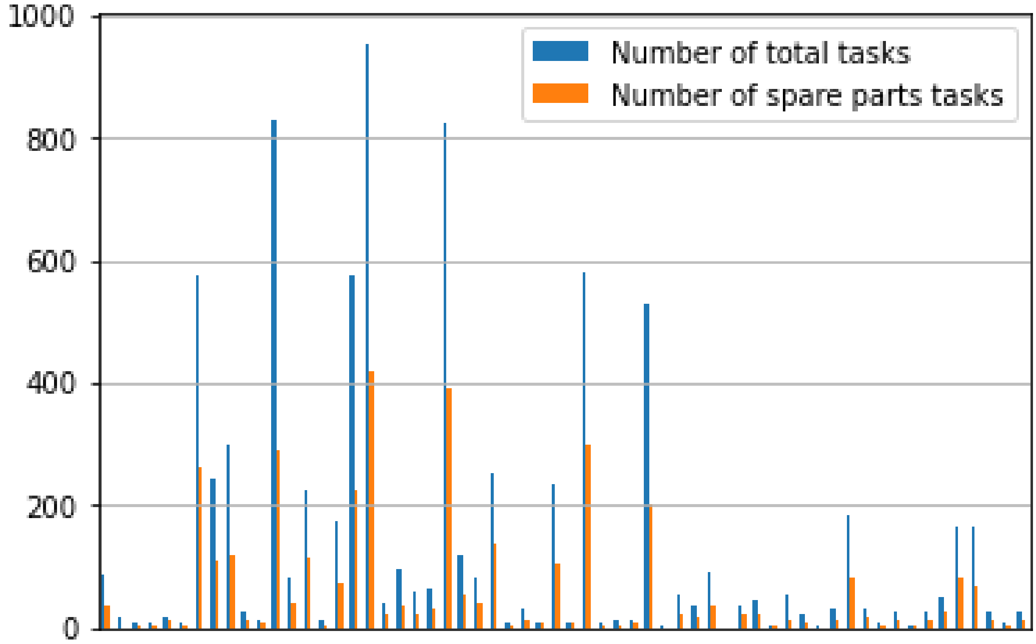
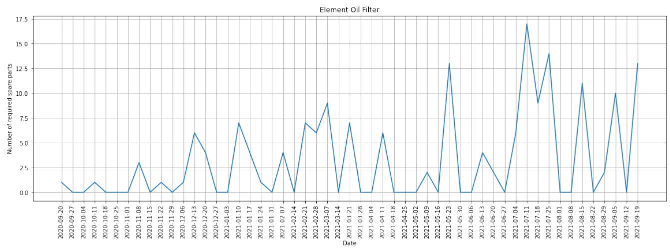
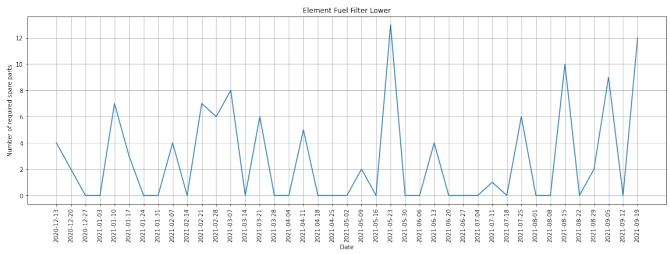


Figure 3.1: Total number of maintenance tasks over the chosen time period for each dealer

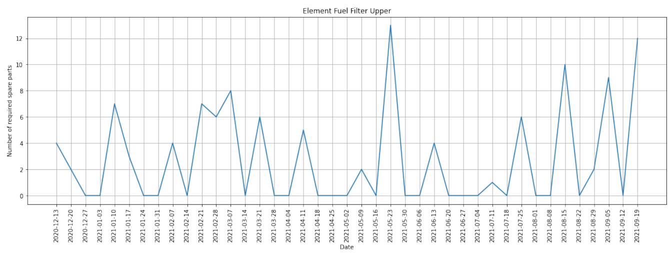
Choosing for instance a specific dealer which registers 955 overall maintenance tasks and 417 component ones, it is possible to plot the weekly time series for each required component (Figure 3.2). Notice that the four figures do not show any particular trend or seasonal variation, with values that range between 0 and 16 components per week. The two fuel filters display the same behaviour as they are usually substituted at the same time.



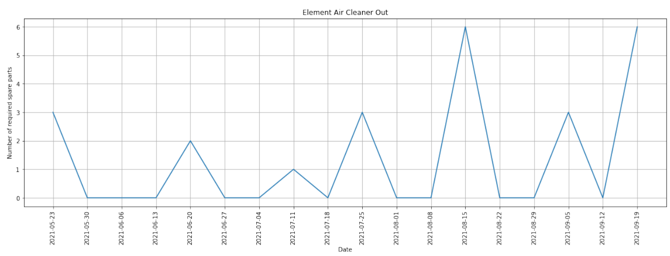
(a) Element Oil Filter



(b) Element Fuel Filter Lower



(c) Element Fuel Filter Upper



(d) Element Air Cleaner Out

Figure 3.2: Weekly Time Series for Spare Parts

3.3.2 The Cumulative Odometer Dataset

This dataset contains vehicles' cumulative traveled kilometers on a daily basis. After a first check on the correctness of the data, it is noticeable that some dates are missing, maybe due to the fact that the vehicle was left unused for some days and no data was collected. For this reason, for each vehicle id, dates are ordered in chronological order and the missing ones are detected and counted, in order to have an insight of the amount of missing data and the usage pattern of the vehicles: some vehicles show almost the entire year of missing data, making them useless for performing any analysis. For this reason, they are removed. In addition to that, some vehicles appear to have 0 cumulative traveled km in between some dates, and since this may be due to some errors in the data collection, they are also removed from the dataset to avoid imprecise results. Furthermore, the remaining missing data about the cumulative traveled km for each missing date are filled with the immediately previous record, as it is likely that the total amount of kilometers has not increased since the last utilization date. The changes in the dataset dimensions after this cleaning phase are reported in Table 3.2.

	Original Dataset	Cleaned Dataset
# of records	9712592	9370842
# of vehicles	51980	45515
# of vehicles with > 300 missing data	45	0
# of vehicles with 0 cumulative km	6420	0

Table 3.2: Comparison before and after dataset cleaning

As already mentioned, one of the deadlines for performing maintenance is related to the amount of travelled road. However, for the vehicles under study, the starting amount of it is not equal to zero. For this reason, the best choice is that of evaluating the daily travelled km by each of them, thus performing a difference between the cumulative travelled amount of two consecutive dates. In this way it is possible to compare this feature to the chosen maintenance threshold.

Nevertheless, another observation must be made as in the first dataset mentioned in Section 3.3.1 the deadlines are associated to the vehicle model and not to the vehicle id, which is not present. Despite this, the maintenance history is still useful to extract the model names and check the corresponding maintenance booklet. The latter contains, in fact, the thresholds associated to each spare part request for all the models in the file. As a result, every vehicle model has a different target for each component. The correspondance is summerised in Table 3.3, where for confidentiality reasons they are reported with a letter.

In this way, by choosing a specific target, it is possible to consider all the models

Vehicle Model	Oil Filter	Fuel Filter Low	Fuel Filter Up	Air Cleaner
A	20000	20000	20000	60000
B	10000	20000	20000	60000
C	20000	20000	20000	60000
D	20000	20000	20000	40000
E	10000	20000	20000	60000
F	20000	20000	20000	40000
G	20000	20000	20000	60000
H	20000	20000	20000	40000
I	20000	20000	20000	40000
L	10000	20000	20000	60000
M	10000	20000	20000	60000
N	10000	20000	20000	50000
O	20000	20000	20000	40000
P	20000	20000	20000	40000

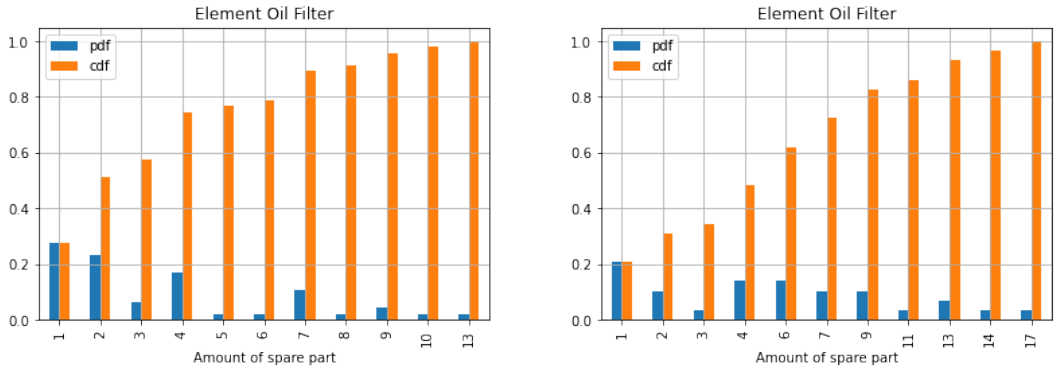
Table 3.3: Deadlines in kilometers for each vehicle model and spare part

that undergo maintenance as soon as the selected deadline is reached. Finally, with the support of the file mentioned at the end of Section 3.2, we are able to link the specific vehicles with the corresponding model. As a matter of fact, selecting a threshold value (and consequently a specific spare part), we can select all the distinct models that perform maintenance in correspondence of that deadline and filter the odometer dataset according to them. The result is a dataset containing the daily traveled kilometers for each selected vehicle, that will be used to predict their future maintenance in time, and evaluate the total required amount for the component under analysis.

3.3.3 Spare Parts Demand Characterization

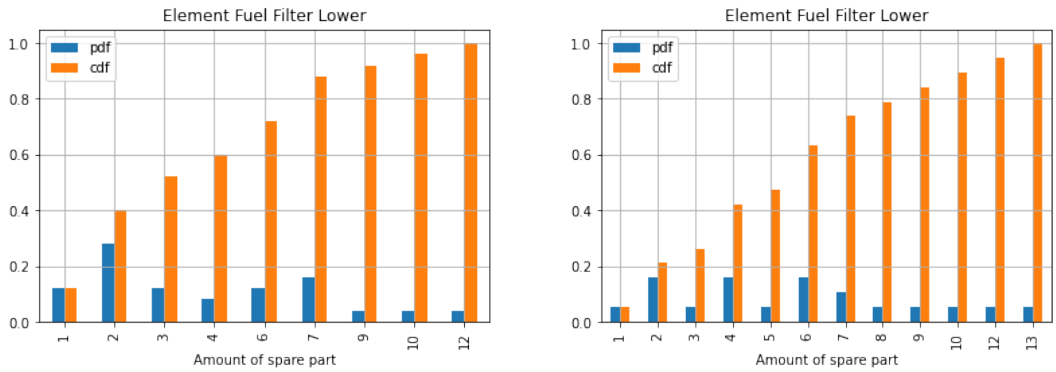
The objective of this dataset analysis is that of finally obtaining the distribution of each component's demand. The result will be provided as input for the inventory optimization model, which is in charge of minimizing the inventory costs of a specific warehouse yet satisfying the demand. Through these kind of data, different scenarios of demand can be gathered, each with a specific quantity of component and probability of occur.

In a preliminary phase of the work, we use the maintenance historical data related to the specific performed tasks over time. Considering the same single dealer as in Section 3.3.1, for each component we evaluate the ECDF either on a daily and on a weekly basis (Figures 3.3, 3.4, 3.5). Through these plots it is possible to understand that the higher the number of requested components, the smaller the probability of occurrence of that specific scenario. This is true in particular for components like the Oil Filter and the Fuel Filter, and it is confirmed by Table 3.3 in which the deadlines for maintenance are shorter, thus hypothetically leading to more frequent requests of that specific spare part. Their behaviours on a daily and weekly basis are similar, and we can deduce that in the majority of the cases it is more likely to have small demands of products. A slightly different behaviour can be noticed in the Air Cleaner distribution, that on a weekly basis shows a uniform-like behaviour, thus leading to a set of scenarios that are more or less equally probable.



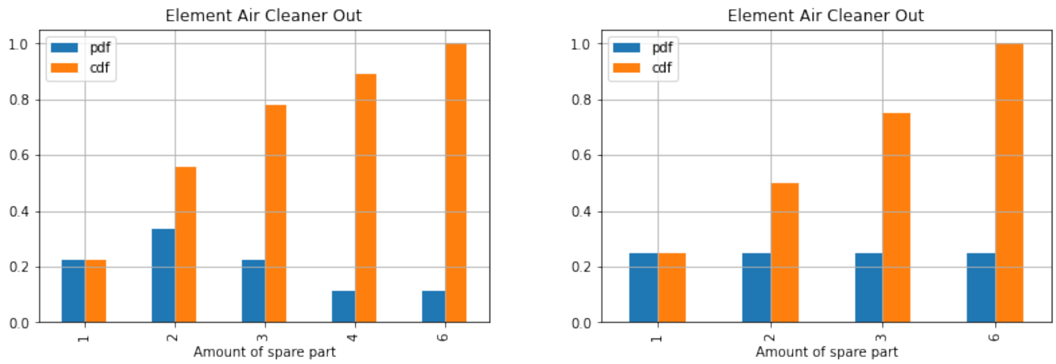
(a) Daily **(b)** Weekly

Figure 3.3: Distribution of Element Oil Filter demand



(a) Daily **(b)** Weekly

Figure 3.4: Distribution of Element Fuel Filter demand



(a) Daily **(b)** Weekly

Figure 3.5: Distribution of Element Air Cleaner Out demand

Chapter 4

Relevant Theory

In this section we will present the theory that appears to be relevant to this thesis project.

In particular, this work is composed of two main blocks, each belonging to a specific macro study area: the definition and implementation of an optimization model for the inventory cost minimization, and the prediction of the future required components, necessary to provide an input to the mentioned model. For this reason, we first focus on the definition of OR optimization model, and on the differences between deterministic and stochastic approach, with particular attention to the latter. Afterwards, we will present some notions of probability theory to model the uncertainty, analysing some commonly known distributions useful for our work. Finally, for coping with the prediction stage of this project, we will describe the regression method both in general terms and in relation to the employed method.

4.1 Optimization Models

An optimization problem is

> "A computational problem in which the object is to find the best of all possible solutions. More formally, find a solution in the feasible region which has the minimum (or maximum) value of the objective function [31]".

Thus, with the term optimization we refer to a discipline within applied mathematics that deals with optimization models, their mathematical properties, and the development and implementation of algorithms. This concept is strictly related to the definition of Operational Research, representing one of the main techniques that support the different phases of an OR approach.

Operational Research concretely originated during World War II, when the British military management contacted a group of scientists to apply a scientific approach in the study of military operations to win the battle. The main objective was to allocate scarce resources in an effective manner to various military operations and to the activities within each operation. The application of OR in the military domain spread interest in it to other government departments and industry.
OR is defined as

> "[...] the application of scientific methods for solving complex problems arising within the management of complex systems of people, machines, materials and money in industry, finance, government and defence. [...] The aim is to aid decision makers in determining scientifically their policies and actions [32]".

One of the main characteristics of OR is thus dealing with complex and real problems, applying advanced analytical modeling and solution methods to the study of operations, to help making better decisions. In other words, OR represents the study of optimal resource allocation.

The main phases of the Operational Research approach are depicted in Figure 4.1 and they consist of specifying and formulating the problem, constructing a suitable mathematical model and deriving a solution from it, testing and modifying the model, and finally implementing the model solution in the real problem situation.

4.1.1 Definitions

Modeling is essential in Operational Research and when speaking of formulating an optimization problem it means performing a translation of the key characteristics of the real problem into mathematical equations and variables. To do so, we need to deal with limitations and restrictions on our freedom to make decisions yet finding

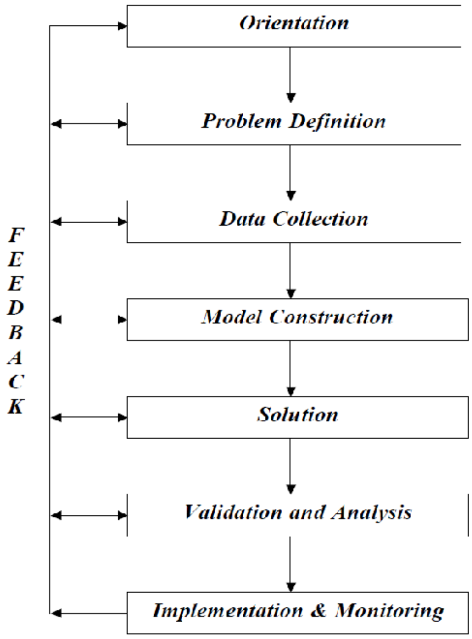


Figure 4.1: Phases of a classical OR approach

the optimal decisions with respect to one or more attributes of the problem itself. An optimization model consists of four main elements:

- **Objective Function**: the objective function, often denoted *f*, reflects a single quantity to be either maximized or minimized. It is a measure used to evaluate the goodness of a solution and to choose the best one among them. It is generally related to economic attributes like cost minimization, revenue maximization, or minimization of deviations.

- **Decision Variables**: the decision variables reflect aspects of the problem that can be controlled. This can include both variables that directly modify the status of the system, as well as variables indirectly influenced by the choice of other ones. Every decision variable in the formulation should either directly influence the objective function or influence another decision variable that affects the objective function. A precise assignment of values to the decision variables is a solution.

- **Parameters**: the parameters are the constant quantities/entities that describe the system under study.

- **Constraints**: the problem constraints represent any kind of limitation on the values that the decision variables take. They are the main way to link parameters and decision variables, ensuring the feasibility of a solution. The most intuitive types of constraints are those which directly and obviously limit the choices to be made (implicit constraints), while the second type of constraint is required to ensure consistency among the decision variables (explicit constraints).

An assignment of values to decision variables that satisfies all the implicit and explicit constraints is called a **feasible solution**. On the contrary, an **infeasible solution** is an assignment of values that violates at least one of the constraints. An **optimal solution** is a feasible solution such that there exists no other feasible solution having a better objective function value.
As a consequence, a model is called feasible if it has at least one feasible solution, infeasible if there exists no feasible solution, and unbounded if there is no precise optimal solution but the value of the objective function can grow or decrease to infinite values.

In order to find a solution, optimization algorithms are used, consisting in a finite step by step procedure with the scope of finding the optimal solution of an optimization problem. Among the optimization algorithms that can be used, we can list exact ones, that at the end of the procedure ensure the optimality of the solution, heuristic ones, that do not give this guarantee, and approximation ones that ensure a maximum gap from the optimality. This distinction is made because computational time needed by an algorithm to return a solution is a key aspect and must be kept as reasonable as possible, which is not always the case for exact solutions.

4.1.2 Mathematical Programming Models

Mathematical programming is one of the most important techniques available for quantitative decision making. Its purpose is that of finding an optimal solution for allocation of limited resources to perform competing activities and the optimality is defined with respect to important performance evaluation criteria, such as cost, time, and profit. To describe the problem under study, mathematical programming uses a compact mathematical model defined as in (4.1) and the formulation includes the objective function and the explicit and implicit constraints.

$$\begin{aligned} \min \quad & f(x_1, ...x_n) \\ s.t. \quad & g_j(x_1, ..., x_n) = 0 \quad \forall j \in \{1, ..., m\} \\ & x_i \quad \forall i \in \{1, ..., n\} \end{aligned} \tag{4.1}$$

Figure 4.2 summarizes the different type of MP and we will analyse their characteristics in the following.

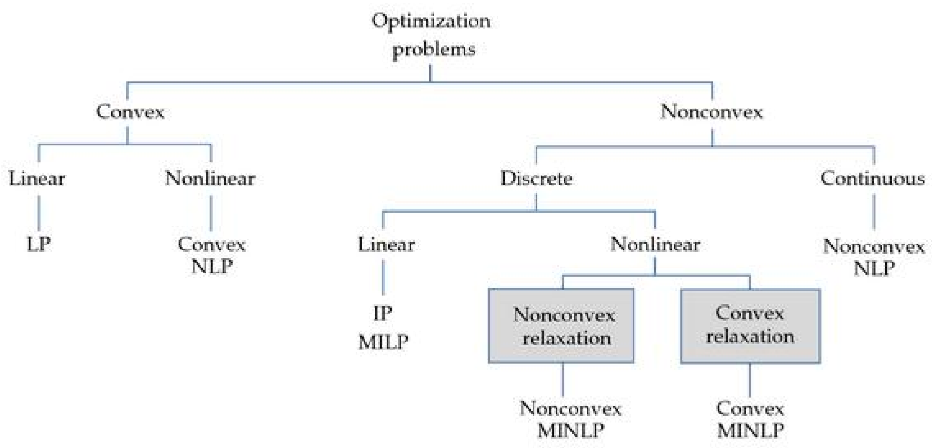


Figure 4.2: Problem types related to optimization problems [18]

Continuous and Integer Models

A first distinction among MP models involves the kind of variables that are present in the model. We refer to **continuous** models when the decision variables under study are defined in the continuous domain, i.e. their cardinality is infinite. (4.2)

$$\begin{aligned} \min \quad & f(x_1, ...x_n) \\ s.t. \quad & g_j(x_1, ..., x_n) = 0 \quad \forall j \in \{1, ..., m\} \\ & x_i \geq 0 \quad \forall i \in \{1, ..., n\} \end{aligned} \tag{4.2}$$

On the contrary, **integer** models contain only discrete (integers) variables, including Boolean ones. It is common to cope with Mixed Integer models, that imply the use of both continuous and integer variables. (4.3)

$$\begin{aligned} \min \quad & f(x_1, ...x_n, y_1, ..., y_q) \\ s.t. \quad & g_j(x_1, ..., x_n, y_1, ..., y_q) = 0 \quad \forall j \in \{1, ..., m\} \\ & x_i \geq 0 \quad \forall i \in \{1, ..., n\} \\ & y_k \in \mathbb{Z} \quad \forall k \in \{1, ..., q\} \end{aligned} \tag{4.3}$$

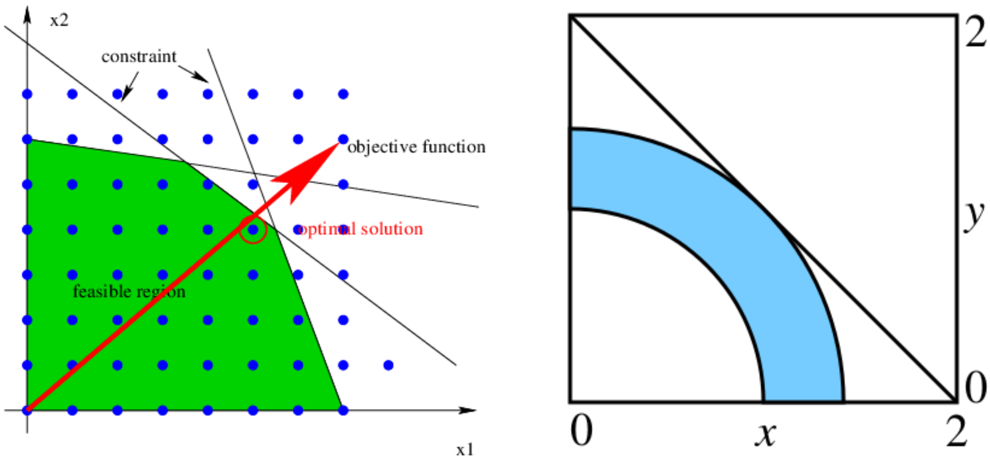
Linear and Non Linear Models

Another difference involves the concept of linearity or not of the studied problem. It is called, in fact, linear programming problem (LP) if both the objective function and the constraints are linear. It is called, instead, a nonlinear programming problem (NLP) if the objective function is nonlinear and/or the feasible region is determined by nonlinear constraints [33] .

Feasibility

According to the type of models under study, one significant difference relies in the shape of the feasibility region, i.e. the set of all possible points of an optimization problem that satisfy the problem's constraints.

For linear programming problems, in fact, the feasibility region changes if considering only continuous decision variables or integer ones too. In general, the feasibility region of a linear continuous problem is a convex and closed region, while for integer ones it is a reticular non convex set of points called convex hull. In the absence of integer constraints the feasible set is thus the entire green region in Figure 4.3a, but with integer constraints it is the set of blue dots.
Due to the fact that the type of constraints influences the feasibility region, its shape changes also for NLP models (Figure 4.3b).



(a) Linear models **(b)** Non-linear models

Figure 4.3: Feasibility Regions
[34]

Several methods are available for solving non-convex problems. One approach is to use special formulations of linear programming problems. Another method involves the use of branch-and-bound techniques, where the program is divided into subclasses to be solved with convex (minimization problem) or linear approximations that form a lower bound on the overall cost within the subdivision. With subsequent divisions, at some point an actual solution will be obtained whose cost is equal to the best lower bound obtained for any of the approximate solutions.

4.1.3 Deterministic and Stochastic Models

A further distinction in optimization models is related to the nature of the input data.
Many problems rely on the assumption that their input parameters are deterministic and exact values, known a-priori and without any uncertainty. Nevertheless, for modeling realistic issues, the lack of knowledge of some data must be taken into account, together with the possibility that these may become known during the decision making process evolution. For this reasons, the problem formulation slightly changes in case of stochastic models, as shown in (4.4), where ω is a random process.

$$\begin{aligned} \min \quad & f(x_1, ...x_n, \omega) \\ s.t. \quad & g_j(x_1, ..., x_n, \omega) = 0 \quad \forall j \in \{1, ..., m\} \\ & x_i \geq 0 \quad \forall i \in \{1, ..., n\} \end{aligned} \tag{4.4}$$

4.1.4 Uncertainty Characterization

When modeling uncertainty, there can be mainly two different sources that generate it [35]:

- **Exogenous**, when decisions cannot influence the stochastic process. Exogenous uncertain parameters are realized at a known stage in the problem irrespective of the values of the decision variables. For example, demand is generally considered to be independent of any decisions in process industries, and hence, it is regarded as an exogenous uncertain parameter.

- **Endogenous**, when decisions impact endogenous uncertain parameters and they may cause alteration of the probability distribution by making one possibility more likely than the other.

Randomicity is usually modeled in form of probability distributions of the random variables under study, obtained through historical data or predictions.

Random Variables

Given an outcome ω of a random experiment, and the set of all the possible outcomes usually denoted as Ω, these outcomes can be combined into subsets $\mathcal{A}$ of Ω called events. For each of these events $A \in \mathcal{A}$ there is a probability measure that tells the probability with which the event belonging to the subset occurs (P). The triplet (Ω, $\mathcal{A}$, P) is called probability space. Random variables are functions that assign a numerical value to each possible outcome over the sample space Ω [36].

When a probability measure has been specified on the sample space of an experiment, we can determine probabilities associated with the possible values of each random variable X. Let C be a subset such that $\{X \in C\}$ is an event, and let $P(X \in C)$ denote the probability that the value of X will belong to the subset C. Then $P(X \in C)$ is equal to the probability that the outcome ω of the experiment will be such that $X(\omega) \in C$.
There are two major classes of distributions and random variables: discrete and continuous.

We say that a random variable X has a discrete distribution or that X is a discrete random variable if X can take only a finite number k of different values. If a random variable X has a discrete distribution, the **probability mass function** (pmf) of X is defined as the function f such that for every real number x, $f(x) = P(X = x)$. In this case, the probability of each subset C can be determined as in (4.5)

$$P(X \in C) = \sum_{x_i \in C} f(x_i) \tag{4.5}$$

On the contrary, a random variable X has a continuous distribution if there exists a function f such that for every interval of real numbers, the probability that X takes a value in the specified interval is the integral of f over that interval (4.6).

$$P(a \leq X \leq b) = \int_a^b f(x)\,\mathrm{d}x \tag{4.6}$$

If X has a continuous distribution, the function f is called the **probability density function** (pdf) of X.

Independently from the discrete or continuous characterization of a distribution, it can be defined in terms of cumulative distribution function (cdf). The cdf of a random variable X is the function $F(x) = P(X \leq x)$ for $-\infty \leq x \leq +\infty$.

Some relevant metrics to describe random variables are the expected value and the variance, that will result useful in this work.

4.1.5 Poisson Process

Among the common-known probability distributions, of particular interest for this work are those distribution that are related to a Poisson Process.
A Poisson process [37] is a model for a series of discrete events where the average time between events is known, but the exact timing of events is random, meaning that they are randomly spaced in time. Furthermore, the arrival of an event is independent of the event before, leading to the statement that the waiting time between events is memoryless.

A Poisson process must meet the following criteria:

- Events must be independent from one another and the occurrence of one event must not influence the probability of another event to happen.
- The average rate (in terms of event per time period) is constant.
- Two events cannot be simultaneous, meaning that each sub-interval in a Poisson process can be seen as either a success or a failure (Bernoulli).

Poisson distribution

As mentioned, a Poisson process describes randomly occurring events, and the Poisson distribution is used to model the number of such events occurring in a fixed time period. As a matter of fact, the Poisson distribution pdf gives the probability of observing a certain number of events in a time period given the length of the period and the average events per time. This factor given by events/time * time period can be summarised into a single parameter λ, the rate parameter. Being $\lambda > 0$, a random variable X has a Poisson distribution with mean λ if the pdf of X is as in (4.7).

$$f(x|\lambda) = \begin{cases} \frac{e^{-\lambda}\lambda^x}{x!} & \text{for x = 0, 1, 2, ...} \\ 0 & \text{otherwise} \end{cases} \tag{4.7}$$

We can think of λ as the expected number of events in the interval, thus it represents the mean of the random variable and also its variance. As we change the rate parameter we change the probability of seeing different numbers of events in one interval as shown in Figure 4.4.

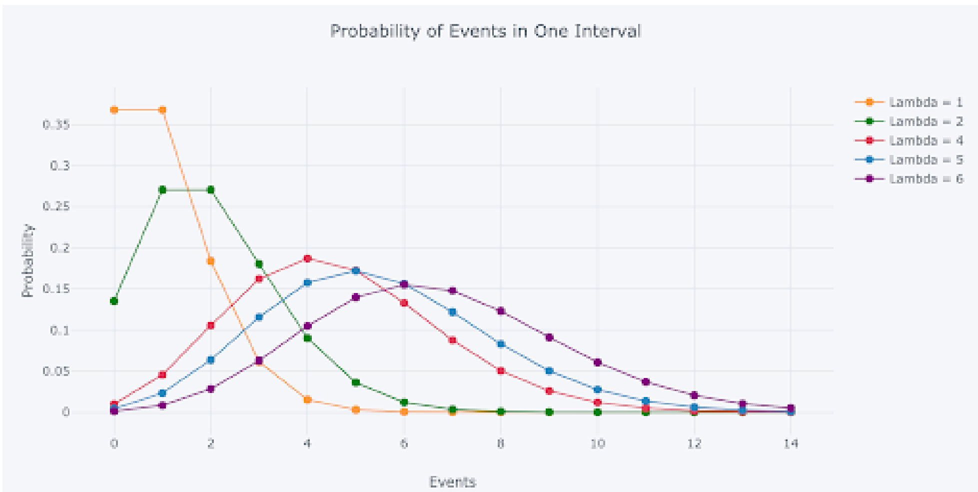


Figure 4.4: Poisson pdf at varying of rate λ [38]

Exponential distribution

An additional step of the Poisson process involves figuring out the waiting time before the following Poisson event, usually called the inter-arrival time. The time to wait between events can be described as a decaying exponential, since the probability of waiting a given amount of time between successive events decreases exponentially as time increases [39]. Evidence is given in Figure 4.5.

A continuous random variable X is said to have an exponential distribution with parameter $\lambda > 0$ if its pdf is given by (4.8).

$$f_X(x) = \begin{cases} \lambda e^{-\lambda x} & \text{x} > 0 \\ 0 & \text{otherwise} \end{cases} \tag{4.8}$$

The probability of waiting less than a specific amount of time t is given in (4.9).

$$P(T \leq t) = (1 - e^{-\frac{events}{time}t}) \tag{4.9}$$

The waiting times between events are memoryless, so the time between two events has no effect on the time between any other events. This memorylessness is also known as the Markov property.

The mean and variance of exponential distributions are, respectively, $\mathbb{E}(X) = \frac{1}{\lambda}$ and $Var(X) = \frac{1}{\lambda^2}$.

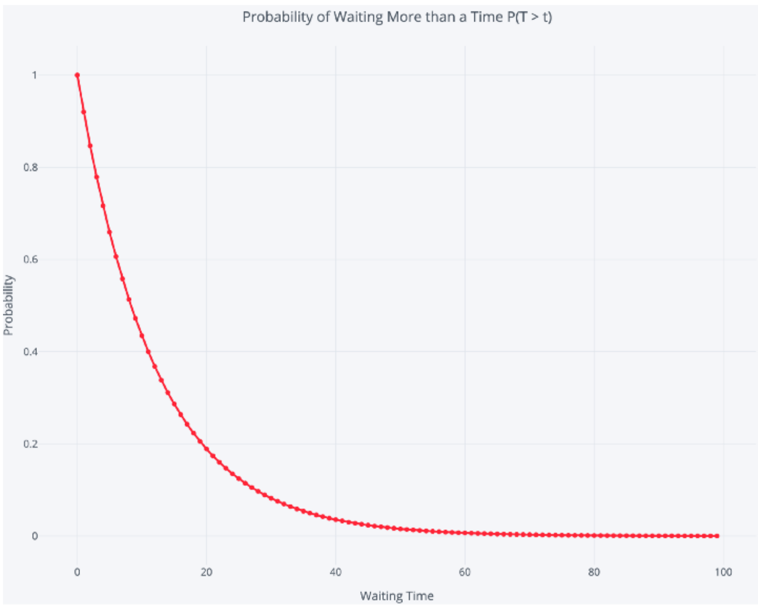


Figure 4.5: Exponential probability of waiting time between successive events [38]

4.1.6 Stochastic Programming

Stochastic programs' aim is that of finding a solution to problems involving some uncertain data, that can be thus represented as random variables [40]. It is essential to provide a probabilistic description of such phenomena, using statistics and expertise to learn the most about uncertain parameters and represent them in the form of probability distributions and densities. Indeed the true value of these variables will be known with certainty only after their realization, which is usually too late to make decisions if we want to optimize the problem under study in an efficient and effective way.

For this reason, the temporal horizon can be divided into **information stages**, which are points in time where decisions are made during a process. The definition of these stages depend from the process structure or can also be a modeling approximation, however it only makes sense to distinguish two points in time as different stages if something relevant is observed in between (Figure 4.6).

Figure 4.6: Information stages

As a result, the set of decisions is divided into two groups (if the information stages are two):

- *first-stage decisions*, which are those decisions that must be taken before the experiment.
- *second-steg decisions*, which are those decisions that must be taken after the experiment.

Considering ω an outcome of the random experiment, and $\xi(\omega)$ the values of the the random variables after the realization of the experiment, first-stage decisions are represented by the vector x, and the second-stage ones are represented by the vector $y(\omega, x)$. The sequence of events and decisions becomes $x \to \xi(\omega) \to y(\omega, x)$. Furthermore, if the uncertainty to be realized can be described by means of a distribution, it is straightforward to derive different scenarios of realization, each with a specific probability. The result of such procedure is the scenario tree depicted in Figure 4.7.

Notice that stochastic programs can be two-stage or multi-stage ones: if there are only two stages then the problem corresponds to a two-stage stochastic program, while in a multistage stochastic program the uncertainty is revealed sequentially in multiple stages, and the decision-maker can take corrective action over this sequence.

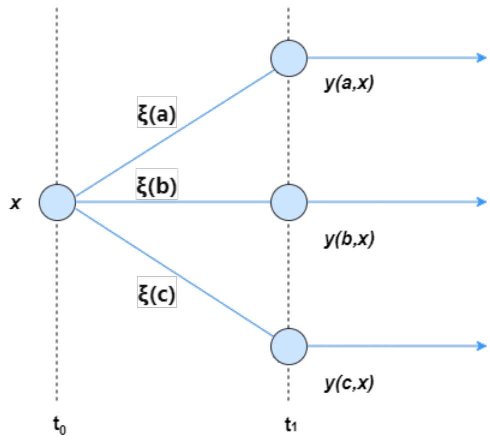


Figure 4.7: Scenario Tree

Finally, when making decisions under uncertainty, we must take into account two properties:

- **Robustness**, which is a first stage property, and it indicates that the model must be capable of bearing random events that will eventually happen.
- **Flexibility**, which is a second stage property, and gives the possibility to the model to adapt to various events occurrences.

Two-Stage Stochastic Programming

Starting from the classical formulation of a random program as in (4.10)

$$\begin{aligned} \min \quad & c^T x \\ s.t. \quad & Ax \leq b \\ & Tx \leq h \\ & x \in X \end{aligned} \tag{4.10}$$

the classical two-stage problem with recourse [41] is the one of finding (4.11)

$$\begin{aligned} \min \quad & c^T x + \mathbb{E}_\xi[Q(x,\xi)] \\ s.t. \quad & Ax \leq b \\ & x \in X \\ \text{where} \quad & Q(x,\xi) := \min \quad q^T y \\ & s.t. \quad Tx + Wy \leq h \\ & y \in Y \end{aligned} \tag{4.11}$$

The first-stage decisions of the two-stage program are represented by the vector x, and corresponding to x there are the first-stage vectors and matrices c , b , and A. In the second stage, a number of random events may realize, and for each realization the second-stage problem data Q, h, T become known. Then, the second-stage decision y must be taken, and they are different according to the realization of the event. W is the recourse matrix.

Random constraints are modeled as "soft" constraints, and their violation is accepted even though the cost of this violation will influence the choice of x through the recourse matrix W.
There can be different type of recourse: random recourse, in which the matrix W depends on uncertainty, fixed recourse, in which W does not depend on uncertainty, simple recourse in which the matrix is defined as W=[I,-I].

Deterministic Equivalent Programming

Notice also that the objective function in (4.11) contains both a deterministic term ($c^T x$) and the expected value of the second-stage objective ($\mathbb{E}_\xi[Q(x, \xi)]$), where for each event realization y is the solution of a linear program. For this reason, it is possible to write the formulation as a deterministic equivalent program (DEP) (4.12), where s is the single scenario that occurs and p_s is the probability for that scenario to happen.

$$\begin{aligned} \min \quad & c^T x + \sum_{s \in S} p_s q_s^T y_s \\ s.t. \quad & Ax \le b \\ & x \in X \\ & T_s x + W y_s \le h_s \quad \forall s \in S \\ & y_s \in Y, \quad \forall s \in S \end{aligned} \tag{4.12}$$

The deterministic equivalent is *scenario-dependent*, and it is in general much bigger in terms of dimensions of variables and constraints.

4.2 Regression Methods

In statistical modeling, regression [42] is a technique useful to infer the relationship between a dependent variable y and p independent variables $\mathbf{x} = [x_1|\ldots|x_p]$. The dependent variable is also known as response variable or outcome, and the independent variables as predictors, explanatory variables, or covariates.

The term regression was coined by Francis Galton in the 19th century to describe a biological phenomenon, in particular he stated that the heights of descendants of tall ancestors tend to regress down towards a normal average [43]. His work, only related to biological phenomena, was later extended to a more general statistical context.

Regression falls into the category of **supervised** problems. For each observation of the predictor measurements $x_i, i = 1, ..., p$ there is an associated response measurement y_i.The model is fitted according to the response to the predictors, with the aim of accurately forecasting future observations (prediction) or better understanding the relationship between the response and the predictors (inference).

Another distinction relies in the used variables, that can be quantitative or qualitative. The former take only continuous numerical values, while the latter can fall into e set of different classes or categories. Regression problems are **quantitative** ones, while classification problems are often referred to as qualitative ones [44].

4.2.1 Random Forest Regressor

Random Forest [45] uses ensemble learning, which is a technique that combines many classifiers to provide solutions to complex problems. In particular, it combines the predictions of multiple decision trees, which constitute the building block of Random Forest.

A decision tree is a decision support technique that forms a tree-like structure. It is composed of a root node, decision nodes and leaf nodes (Figure 4.8). This kind of algorithm divides a training dataset into branches, which further segregate into other branches. This sequence continues until a leaf node is reached. The nodes in the decision tree represent attributes that are used for predicting the outcome. The main difference of Random Forest algorithm with respect to decision tree is the fact that establishing root nodes and segregating nodes is done in a random way, exploiting a method called *bagging* to issue predictions.

For Random Forest *classifiers*, this technique uses different samples of training data rather than just one sample. The decision trees produce different outputs, depending on the training data fed to the random forest algorithm, these outputs will be ranked, and the output chosen by the mode of the decision trees becomes the final output of the system (Figure 4.9a).

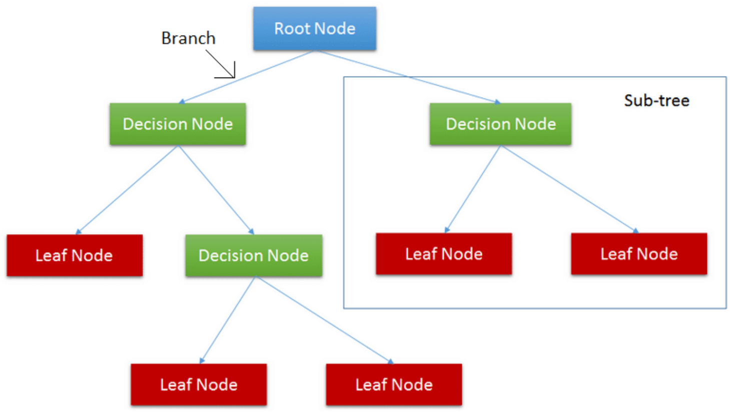
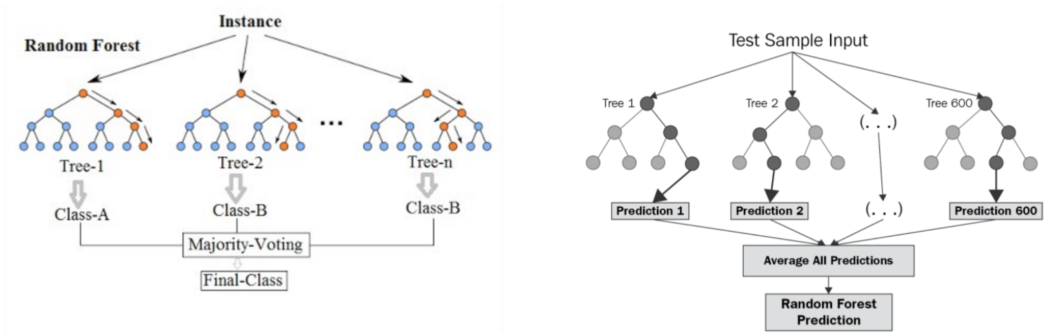


Figure 4.8: Decision Tree structure
[46]

In case of Random Forest *regressors*, instead, features and independent variables are passed to the model and the result of the regression will be the mean prediction of the individual trees (Figure 4.9b).



(a) Classification **(b)** Regression

Figure 4.9: Random Forest Algorithms

An important last step involves its final performance testing. Considering Random Forest as a regression algorithm, its results can be assessed using all the available regression metrics. The most commonly used among the many are the Mean Squared Error (MSE), the coefficient of determination R2, Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE).

In particular, the MSE is the average of the summation of the squared difference between the actual output value and the predicted output value, and it should be kept as small as possible (4.13). The RMSE is just the square root of the MSE.

The MAE is instead the mean of the absolute values of the individual prediction error over all instances in the test set (4.14).

The R2 score is a number between 0 and 1 that measures how well a statistical model predicts an outcome. The outcome is represented by the model's dependent variable. The better a model is at making predictions, the closer its value will be to 1. In the equation (4.15) RSS stands for the sum of squared residuals and TSS for the total sum of squares

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{4.13}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{4.14}$$

$$R^2 = 1 - \frac{RSS}{TSS} \tag{4.15}$$

The advantages of using a Random Forest are the fact that it is more accurate than a single decision tree, it provides an effective way to handle missing data, solves the issue of overfitting in decision trees and can produce predictions without hyper-parameter tuning.

Chapter 5

Methodology and Results

In this chapter we will present the step-by-step process that we followed to satisfy the requirements of this thesis project. In particular, we will analyse in detail how data are processed and analysed; afterwards, we will introduce the implementation of both the forecast model and the optimization one; finally, results will be compared and critically analysed to give evidence of the usefulness of the work.

5.1 Data Pre-Processing

As described in Chapter 3, the input data used for this work come from different sources and are put together to satisfy the main requirements of having a significant number of vehicles' data and enough maintenance task information about specific spare parts' deadlines. All useful data are provided by Tierra S.p.A, a company providing telematics services to industrial vehicles producers. In particular, the needed files are:

- Vehicles maintenance history, containing information from January 20, 2020 to January 31, 2022, for a total of 8242 records about performed maintenance tasks with the corresponding deadline.
- Maintenance booklets, containing information about performed interventions for each vehicle model and whose models names are directly linked to the previous data source.
- Vehicles' traveled kilometers, containing information from October 1, 2021 to September 30, 2022 related to the day-by-day cumulative km of 51980 vehicles. It gathers together 9712592 records.
- A vehicle-to-model file that links the vehicles ids with the corresponding company model, to check which vehicles have specific deadlines according to

the model.

5.1.1 Instruments

The majority of the input data files are directly provided in CSV format from the company. However, due to its high dimensions, the dataset containing the vehicles traveled kilometers can be retrieved exploiting Amazon Web Service (Figure 5.1).

Figure 5.1: Amazon Web Service, Amazon S3, Amazon Athena and Amazon SageMaker logos ©

Amazon Web Service (AWS) [47] is an online platform that provides scalable and cost-effective cloud computing solutions. It was launched in the early 2000s as a subsidiary of Amazon and it provides on-demand cloud computing platforms and APIs to individuals, companies, and governments, on a pay-as-you-go pricing model with no upfront cost. It is adopted worldwide and offers over 200 fully featured services from compute power and database storage to content delivery, helping companies to scale and grow. Amazon Web services offers flexibility since only the actually used services are paid for, moreover enterprises using AWS can reduce capital expenditure of building their own private IT infrastructure.

The AWS global infrastructure is divided into geographical regions, which are further divided into separate availability zones, physically isolated from each other, and providing business continuity for the infrastructure as in a distributed system. If one zone fails to function, the infrastructure in other availability zones remains operational. These availability zones are connected by AWS's own high-speed fiber-optic network.

There are three cloud computing models available on AWS:

- *Infrastructure as a Service (IaaS)*: basic building block fo cloud IT, highly flexible. It provides access to data storage space, networking features and computer hardware, and provides management controls over the IT resources to the developer.

- *Platform as a Service (PaaS)*: AWS manages the underlying infrastructure, and this solution helps the developer to be more efficient as there is no need to worry about capacity planning, software maintenance, resource procurement etc.

- *Software as a Service (SaaS)*: complete product that usually runs on a browser. It refers mainly to end-user applications, and it is run and managed by the service provider, thus the user only has to care about the most suitable application of the software.

AWS can be accessed through a web-based interface called AWS Management Console, signing in with an AWS account. In our case, the company is already equipped with one, and from the console home page it is possible to see all the provided services.
In the following we will analyse some of the ones exploited in this work.

Amazon S3

Amazon S3 or Amazon Simple Storage Service [48] is a service offered by AWS that provides object storage through a web service interface, ensuring scalability, high availability, and low latency with high durability. The basic storage unit is an *object* identified by a unique key, and objects are immutable and organized into buckets. Buckets are containers with their own set of policies and configurations, enabling users to have more control over their data. An object in S3 can be between 1 byte and 5TB. If an object is larger than 5TB, it must be divided into chunks prior to uploading. When uploading, Amazon S3 allows a maximum of 5GB in a single upload operation; hence, objects larger than 5GB must be uploaded via the S3 multipart upload API. Theoretically AWS S3 is supposed to have infinite storage space, making it infinitely scalable for all kinds of use cases. Furthermore, users are charged according to the S3 storage they hold.

Amazon Athena

Amazon Athena [49] is an interactive query service that allows to analyse Amazon S3 data through standard SQL. Athena is serverless, so there is no infrastructure to manage, and easy to use, since it is necessary only to point at data in Amazon S3, define the schema and start querying. Moreover, the only cost to pay is for the used queries. Most results are delivered within seconds. Athena uses Presto, an open source SQL query engine, with ANSI SQL support and works with a variety of standard data formats, including CSV, JSON, ORC, Avro, and Parquet. It is ideal for interactive querying and can also handle complex analysis, including large joins, window functions, and arrays.

Amazon SageMaker

Amazon SageMaker [50] is a fully managed service that allows to build, train, and deploy ML models quickly, giving complete access, control and visibility into all the steps. It provides all the components used for machine learning in a single toolset to make the process faster and cheaper. It is easy and quick to upload data, create notebooks, train and tune models all in one place, and all the activities involved in the ML development can be performed within the unified SageMaker Studio visual interface. For what concerns our interests, Amazon SageMaker provides one-click Jupyter notebooks (Figure 5.2) that allows to start working in seconds. The underlying compute resources are elastic, allowing to dial up or down the available resources and the changes take place automatically in the background without interrupting the work. It also enables one-click sharing of notebooks, making it easier to collaborate with others, who will get the same notebook saved in the same place.

Figure 5.2: Jupyter logo ©

The aforementioned services are useful for our thesis because Athena is able to map all the S3 objects into tables, allowing to analyse them through simple queries. In particular, data can be retrieved from Athena and elaborated with the help of some tools and libraries (Figure 5.3):

- **Pandas** [51] is a fast, powerful and flexible open source software library for data manipulation and analysis, built on top of Python programming language. In particular, it offers data structures and operations for manipulating numerical tables and time series.

- **NumPy** [52] is an open source package for scientific purpose in Python. It offers comprehensive mathematical functions, random number generators, linear algebra routines, vectorization, indexing and more.

- **Seaborn** [53] is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.

- **Matplotlib** [54] is a comprehensive library for creating static, animated, and interactive visualizations in Python. It is possible to create publication quality plots, make interactive figures that can zoom, pan, update, customize visual style and layout and export many file formats.
- **SciPy** [55] provides algorithms for optimization, integration, interpolation, eigenvalue problems, algebraic equations, differential equations, statistics and many other classes of problems. It extends NumPy providing additional tools for array computing and provides specialized data structures, such as sparse matrices and k-dimensional trees.
- **SciKit-Learn** [56] is an open source, simple and efficient tool for predictive data analysis, built on NumPy, SciPy and matplotlib. It features various classification, regression and clustering algorithms, dimensionality reduction, model selection and pre-processing.
- **Awswrangler** [57] is an open source Python library that enables to focus on the transformation step of extract, transform and load (ETL) data pipelines by using familiar Pandas transformation commands and relying on abstracted functions to handle the extraction and load steps. It reduces the time it takes to aggregate and prepare data for machine learning.

Figure 5.3: Logos of the used libraries ©

The input data about the vehicles' odometer information are stored in a database called "politecnico_db" on Athena, from which it is possible to retrieve the file "patchwork_odometer" containing all the useful records for our purposes. In order to transfer these data on a Jupyter Notebook in Amazon SageMaker, we can perform a single query thanks to the awswrangler library that allows us to manipulate it and transform it into a pandas DataFrame. In particular, DataFrames permit to organize data into table-like structures with named columns, making it easier to work with them. The employed query is reported in the following:

```
sql = '''SELECT *
FROM "politecnico_db"."patchworkb_odometer"''' #<-- write here your SQL (PrestoDB) query

df = wr.athena.read_sql_query(sql, database = "politecnico_db", ctas_approach = False, s3_output='s3://peruser-athena-result/datalake-admin-cctutorial/') #run the previous query in Athena and return a pandas dataframe
```

athena_query.py

5.1.2 Data Transformation

The retrieved data are loaded as DataFrames in pandas in order to carefully analyse their attributes. The first dataset, related to the maintenance history, is composed as in Table 5.1.

Attribute	Type	Description
`id_book`	`int64`	Indicates the specific booking id
`id`	`int64`	Indicates the id of the dealer
`confirmed_date`	`string`	Indicates the date in which the performed maintenance has been confirmed
`garage`	`string`	Indicates the dealer's name
`NAME`	`string`	Indicates the model's name
`num_tasks`	`int64`	Indicates the number of tasks performed in that date
`deadline_km`	`float64`	Indicates the maintenance deadline in terms of traveled km
`deadline_time`	`float64`	Indicates the maintenance deadline in terms of seconds elapsed
`deadline_engine_hours`	`float64`	Indicates the maintenance deadline in terms of hours the vehicle was on
`name`	`string`	Indicates the name of the specific performed maintenance task
`id_task`	`int64`	Indicates the id of the performed maintenance task

Table 5.1: Attributes description of maintenance history

After analysing the dataset composition, only the columns related to the *confirmed date*, the *garage*, the *model's name*, the *deadline in km*, the *name* and the *id* of the tasks are kept, since they represent the useful ones for our purposes. Notice that the confirmed date attribute is changed into datetime format for an easier manipulation.

The second dataset related to the vehicles' traveled kilometers is composed as summarised in Table 5.2.

Attribute	Type	Description
date	string	Indicates the specific day corresponding to the registered traveled km
cumulated_raw_odometer	float64	Indicates the cumulative value of the traveled km, incremented day by day
unit_uuid	int64	Indicates the id of the vehicle whose traveled km are registered

Table 5.2: Attributes description of vehicles' cumulative odometer values

This dataset contains vehicles' cumulative traveled kilometers on a daily basis, and also in this case the date is changed into a datetime format. After this, dates are ordered chronologically for each vehicle id, and due to the presence of some errors and missing data, a first cleaning step is performed, as explained in Section 3.3.2. The best choice turns out to be the one of slightly modifying the dataset to obtain only the vehicles' daily amount of traveled km from the cumulative ones. This is done by performing a difference between the cumulative travelled amount of two consecutive dates, allowing to compare the obtained values to the chosen spare parts maintenance threshold, which in our case is the deadline in km.

As shown in Table 3.3 of Section 3.3.2, for each vehicle model and spare part, retrieved from the maintenance history, there is a specific km threshold associated, obtained looking at the maintenance booklets provided by the company. As a result, every vehicle model has a different target for each component, and by choosing a specific target it is possible to consider all the models undergoing maintenance as soon as the selected deadline is reached.

To link the specific vehicle with the corresponding model, the third needed file is a model-to-vehicle mapping organized as explained in Table 5.3.

As a matter of fact, in consideration of all the available data, the procedure to obtain the final dataset to be prepared for the forecasting step is the following:

- The maintenance history dataset is filtered to keep only the 4 spare parts under study and the models that have different components maintenance deadlines according to the booklets.
- We choose the targets in km, and for each target we consider only the vehicles that undergo maintenance according to that reached threshold.

Attribute	Type	Description
vehicle_id	int64	Indicates the vehicle id corresponding to the "unit_uuid" in the odometer file
model_name	string	Indicates the name of the model corresponding to the specific vehicle
id	int64	Indicates the id of the specific kind of model
book_name	string	Indicates the name of the booklet to which the model belongs

Table 5.3: Attributes description of vehicles to model file

- Looking at the file summarized in Table 5.3 we keep all the vehicles ids corresponding to the chosen models.
- The vehicles' odometer dataset is filtered according to the vehicles' ids extracted from the previous point. The dataset is already cleaned from errors and missing data. For example, the resulting vehicles considering each target are reported in Table 5.4.

Target maintenance deadline [km]	# vehicles
10000	340
20000	12768
40000	6341
60000	6438

Table 5.4: Number of vehicles per target maintenance deadline after the cleaning

The result is a dataset containing the daily traveled kilometers for each selected vehicle (Figure 5.4), ready to be prepared for the forecast step. Notice that the daily traveled km are consistent with the cumulative odometer value that increases correctly day by day.

5.1.3 Dataset Preparation

Starting from the dataset obtained in Section 5.1.2 the main purpose is that of building the training dataset for allowing the chosen ML algorithm to perform predictions.

In order to prepare the dataset in the most effective way to achieve our purposes, it is useful to present the main variables of interest in relation with the objectives

	date	cumulated_raw_odometer	uuid	daily_km
0	2021-10-01	239315.0	11775	0.0
1	2021-10-02	239360.0	11775	45.0
2	2021-10-03	241618.0	11775	2258.0
3	2021-10-04	242814.0	11775	1196.0
4	2021-10-05	243880.0	11775	1066.0

Figure 5.4: Example of daily traveled km dataframe

to reach. We start from the statement of the problem presented by Markudova et al. in [30], where the aim was that of predicting next-day utilization hours and remaining days until maintenance according to specific utilization hours cycles. We adapt it to our use case introducing the traveled km. Taking into account a single vehicle v indicated by the `uuid` in Figure 5.4, we consider:

- T^v[km] as the target maintenance deadline in kilometers for each vehicle under study belonging to a specific model. It indicates the traveled km by each vehicles before having to perform maintenance.
- $U^v(t)$[km] as the time series of **daily** traveled km by each vehicle. It corresponds to the column `daily_km` in Figure 5.4.
- $L^v(t)$[km] as the series of km left to the next maintenance of the vehicle. It will correspond to the column `km_to_maint`.
- $D^v(t)$[days] as the time series of the days left to the next maintenance of the vehicle. It will correspond to the column `days_to_maint`.

The first two information are already available in the current dataset, however some further preparation is needed to obtain the remaining ones, i.e. the km and days left to the next maintenance. To do so, the perform the following steps:

- We initialize an empty list called *days_per_cycle* to keep track of the days left to the next maintenance target for all the cycles that each vehicle performs. This is useful to understand the **frequency of maintenance**.
- We also initially set the km left to maintenance $L^v(t)$[km] to the deadline target T^v[km], and the $D^v(t)$[days] to -1.
- for all the dates inside the vehicle's dataframe, so day-by-day, we check if the difference between the remaining km to maintenance and the daily traveled km ($U^v(t)$[km]) is negative, meaning that the target T^v[km] is reached, and

we can store the days of the maintenance cycle, the days left to maintenance, and reset the values of $D^v(t)$[days] to -1 and $L^v(t)$[km] to the deadline target T^v[km].

- if the previous condition is not respected, we simply increment the count of the days left to maintenance, we decrement the km left to maintenance of the daily traveled amount ($U^v(t)$[km]), and store in the new dataframe the date, the vehicle id, the daily traveled km, and the decremented new km left to maintenance $L^v(t)$[km].

For further clarity, the code in the following shows how to achieve the desired results:

```python

def prepare_df(df_vehicle, target_km):
    days_per_cycle = []
    km_to_maintenance = target_km
    days_count = -1
    data = defaultdict(list)

    for _, row in df_vehicle.iterrows():
        # If cycle has finished, restart counter
        if km_to_maintenance - row['daily_km'] < 0:
            days_per_cycle.append(days_count)
            data['days_to_maint'].extend(range(days_count, -1, -1)
)
            km_to_maintenance = target_km
            days_count = -1

        date_today = pd.to_datetime(row["date"])
        usage_until_now = df_vehicle[df_vehicle["date"] <
date_today]["daily_km"].mean()
        data["uuid"].append(row['uuid'])
        data["avg_util"].append(usage_until_now)
        data["date"].append(row['date'])
        data['daily_km'].append(row['daily_km'])
        km_to_maintenance -= row['daily_km']
        data['km_to_maint'].append(km_to_maintenance)
        days_count += 1
    data['days_to_maint'].extend([None] * (days_count + 1))
    dataset = pd.DataFrame(data)

    return dataset, days_per_cycle
```

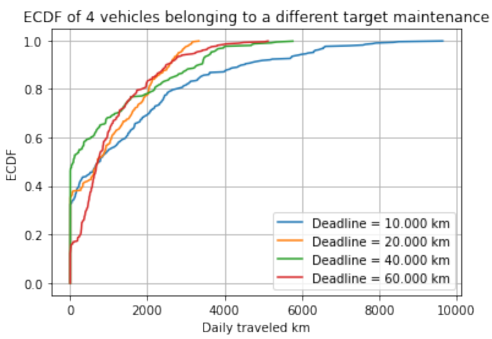
In the end, the resulting dataframe will be as shown in Figure 5.5. Notice that also the average traveled km up to the specified day are added for more insights.

	uuid	avg_util	date	daily_km	km_to_maint	days_to_maint
0	11775	0.000000	2021-10-02	45.0	19955.0	14.0
1	11775	22.500000	2021-10-03	2258.0	17697.0	13.0
2	11775	767.666667	2021-10-04	1196.0	16501.0	12.0
3	11775	874.750000	2021-10-05	1066.0	15435.0	11.0
4	11775	913.000000	2021-10-06	357.0	15078.0	10.0

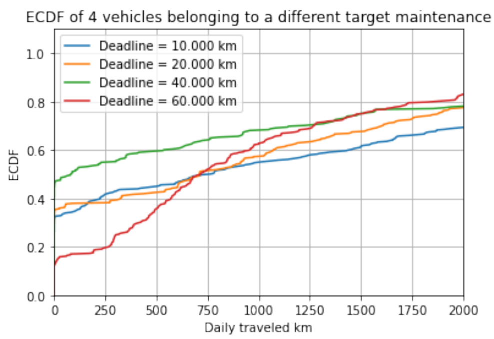
Figure 5.5: Result of dataset preparation

5.1.4 Data Characterization

After having obtained all the relevant information for the vehicles under study, it is useful to perform some analysis to understand the characteristics and behaviour of some data. In particular, we consider 4 target maintenance deadlines T^v: 10000, 20000, 40000 and 60000 km, which are the ones under study for the vehicles resulting from Section 5.1.2.
Then, for each target deadline, we consider one representative vehicle id and the corresponding records over an entire year.
Taking into account the daily traveled km, it is possible to derive the empirical cumulative distribution function (ECDF) for all four cases (Figure 5.6).



(a) Original

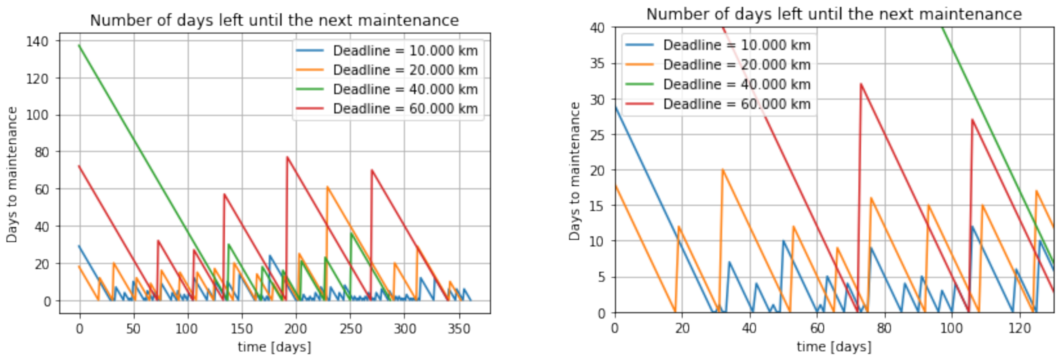


(b) Zoomed

Figure 5.6: ECDF of daily traveled km $U^v(t)$ for each target maintenance deadline

Notice that the behaviour of the vehicles for all the targets larger than 20000 km is more or less the same, opposite to the vehicle with target equal to 10000 km, whose plot highlights some very high amounts of daily traveled km with a probability that is not negligible. Since it is quite unlikely for a vehicle to drive more than 4000 km a-day, these values may be considered as outliers due to errors in the data collection and thus can be removed. For what concerns the reasonable values, the plots highlight the tendency of more or less all the vehicles to travel less than 2000 km per day, which is still a quite high amount of distance.

This last observation leads to the necessity to understand how frequent are the maintenance cycles for the vehicles. For this reason, Figure 5.7 shows the number of days left until the next maintenance for the 4 vehicles. It is evident that for all the



(a) Original **(b)** Zoomed

Figure 5.7: Number of days left until next maintenance

4 vehicles maintenances are quite frequent, with few days per maintenance cycle, which is a reasonable outcome due to the fact that as we have seen they travel a lot of km daily. Rationally, as the target maintenance deadline increases, also the number of days in each maintenance cycle does, however for a smaller threshold it could be more useful to understand the **rate** with maintenances happening for the vehicles.

Notice that in the end we want to obtain the distribution of the required spare parts to derive the scenarios for our optimization model. If maintenances are quite frequent they may generate some stationary demand scenarios, since the shipment time for a specific spare part may be much bigger than the number of days between one maintenance and the other. For this reason, from now on we will differentiate two possibilities:

- **Short maintenance cycles** with respect to the considered lead time, characterised by a relatively small amount of days in-between target km deadlines, that make maintenances more frequent. This will correspond to stationary demand.

- **Long maintenance cycles** with respect to the considered lead time, characterised by a higher number of remaining days until the next maintenance. This will correspond to non-stationary demand.

5.1.5 Short Maintenance Cycles

Vehicles with short maintenance cycles are those which feature small amount of days left to the next maintenance. For each target maintenance deadline and for each related vehicle we collect the number of days to maintenance inside a list containing all the performed cycles.

Notice that, after having removed the outliers as said in Section 5.1.4, it can be useful to evaluate the mean value and the standard deviation of the maintenance rate for all the vehicles under study, in order to derive a distribution of the mean value as a measure of the similarity of the vehicles' behaviour. The result is shown in Figure 5.8.

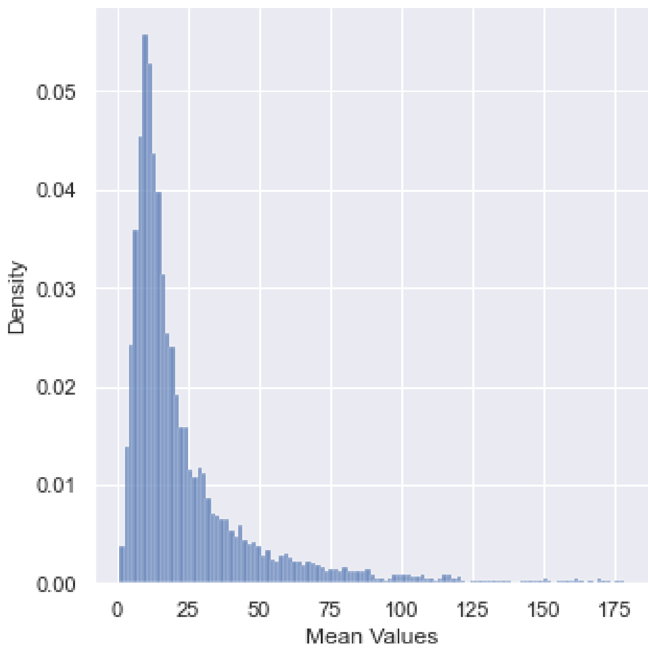


Figure 5.8: Distribution of the mean values of the maintenance cycles

The majority of the average days per cycle are more or less concentrated in correspondence values that are smaller than 25 days per maintenance cycle, with a distribution that can be fitted to a decaying exponential one. For more in-depth information, the distribution of the whole days in each maintenance cycle, for all the vehicles under study, is presented in Figure 5.9. As a matter of fact, the behaviour is confirmed to be that of an exponential distribution, with very few large values indicating the number of days in each cycle.

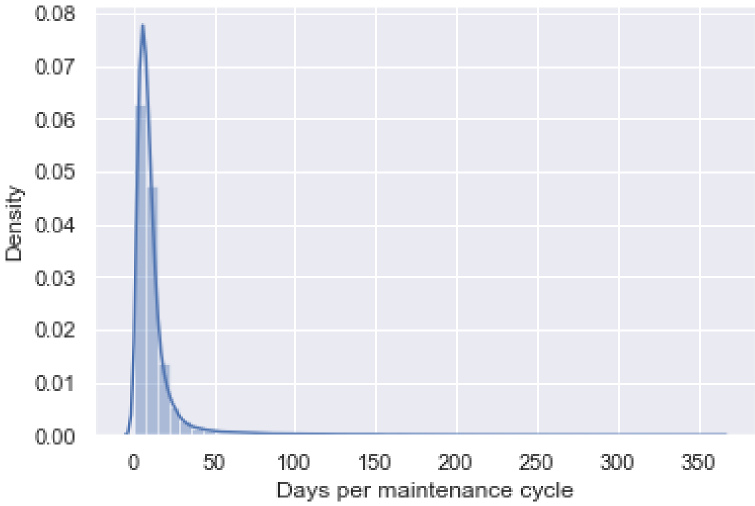


Figure 5.9: Distribution of the days per maintenance cycle

Distribution of the number of vehicles - Stationary Demand

The exponential distribution is often concerned with the amount of time until some specific event occurs. As explained in Section 4.1.5 the time to wait between events can be described as a decaying exponential, since the probability of waiting a given amount of time between successive events decreases exponentially as time increases. This definition is suited with the outcome of our analysis, since the days to wait until the occurrence of next-maintenance event follow an exponential decay.

Considering 12000 vehicles in total, corresponding to our example of target maintenance deadline equal to 20000 km, they all show a similar behaviour and follow the same distribution of **inter-time maintenance**, as we can call the days in-between maintenances. From the inter-time maintenance exponential distribution we can derive a mean of μ units of time.

Assuming that these times are independent, meaning that the time between events is not affected by the one between previous events, the process can be considered **Poisson**. In this way it is possible to obtain the number of events per unit time, which is exactly what we want to derive to understand how many vehicles undergo maintenance (and thus how many spare parts are needed) in the unit of time. Thus the number of events per unit time will follow a Poisson distribution with mean $\lambda = \frac{1}{\mu}$. If we want to understand how many vehicles will undergo maintenance in a specific time interval, we can consider to ideally place orders once every two weeks or once every four weeks. In this cases, applying the formulations for the Poisson distribution in (5.1) we can evaluate the probability of having a certain number k of vehicles under maintenance in the chosen unit of time.

As a result we obtain the plots in Figure 5.10 for different values of λ obtained from Table 5.5. Notice that $\lambda = 1$ refers to maintenances every 2 weeks and $\lambda = 2$ to maintenances every 4 weeks.

$$P(k\ events\ in\ interval) = \frac{e^{-\lambda}\lambda^k}{k!} \tag{5.1}$$

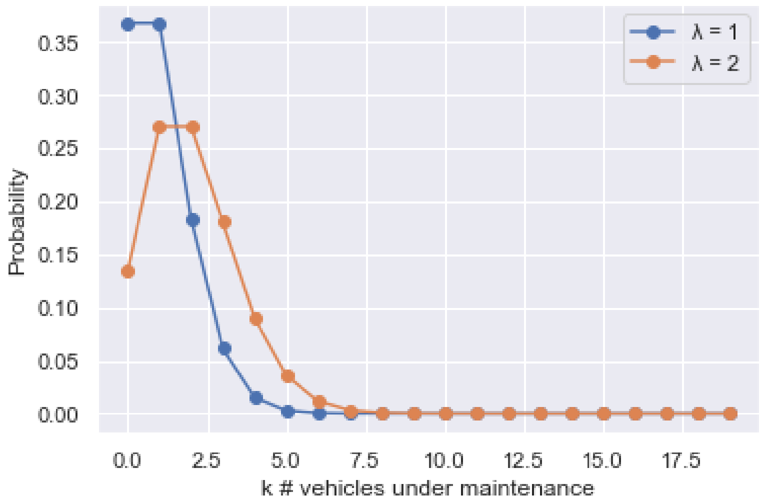


Figure 5.10: Distribution of the number of vehicles k that undergo maintenance every 2 and 4 weeks

	μ	λ
Interval of 2 weeks	0.93	1.07
Interval of 4 weeks	0.47	2.14

Table 5.5: Values of the mean μ of the exponential inter-maintenance time and the rate λ of the poisson distribution

Once we have obtained the distribution of the number of vehicles undergoing maintenance, we can easily derive the desired number of scenarios for our stationary demand, each with its own probability value, by defining an interval of values and extracting the corresponding probability.

5.1.6 Long Maintenance Cycles

Vehicles with long maintenance cycles are those which feature a great amount of days left to the next maintenance. Usually this kind of behaviour can be seen in vehicles whose target maintenance deadline is higher, for instance 60000 km, which contributes to make the next maintenance further in time. If maintenances are far away in time it can be more useful to exploit predictions for the next due one, in order to then consider the errors and build up the different non-stationary demand situations.

In this specific case, we can think of considering only the vehicles whose days to maintenance are around 150. In order to further prepare the dataset for the machine learning process, it is possible to enrich the available data with additional information about the historical daily traveled km in the past instant within a window time interval of size 10 days. Once this last step is performed, it is possible to proceed with the forecasting process.

5.2 Forecast

After the dataset processing, we have distinguished two cases of maintenance cycles: short and long ones. For vehicles with long maintenance cycles it is useful to perform a forecast by exploiting machine learning techniques. To do so, we take advantage of the work by Markudova et al. in [30], and in particular we focus on the methodology followed to predict the remaining time to next maintenance as in their *task B*. In the following, we will present the authors' choices for the training and regression phases of the process, and we will explain the differences adopted in our thesis project with respect to theirs.

5.2.1 Training of the model

For training the model under study the first thing to do is to choose the training set. It must be used to teach prediction models how to extract features that are relevant to specific goals. After that, the trained model can be evaluated using a new set of data, called test set, which must not be used during the training.

The authors employ the strategy of training the regression models on a portion of the historical data that come before a chosen day t. The portion is chosen by using a training window equal to a certain value in order to set the dimension of the previous set of data to use. All the subsequent days ($t+1$) are used to test the trained model. Since the training phase requires a long period, they choose to keep the training window fixed and equal to 70% of the whole set.

For our work, we start from the dataset in Figure 5.11 and we perform the training on all the available vehicles except one, that will be used to perform the prediction. Specifically, for example, the vehicle whose id is 3213 is excluded from the training set.

	uuid	daily_km	km_to_maint	days_to_maint	U(t-1)	U(t-2)	U(t-3)	U(t-4)	U(t-5)	U(t-6)	U(t-7)	U(t-8)	U(t-9)
0	3213	19.0	19885.0	160.0	12.0	11.0	14.0	2.0	10.0	20.0	9.0	8.0	10.0
1	3213	15.0	19870.0	159.0	19.0	12.0	11.0	14.0	2.0	10.0	20.0	9.0	8.0
2	3213	16.0	19854.0	158.0	15.0	19.0	12.0	11.0	14.0	2.0	10.0	20.0	9.0
3	3213	9.0	19845.0	157.0	16.0	15.0	19.0	12.0	11.0	14.0	2.0	10.0	20.0
4	3213	203.0	19642.0	156.0	9.0	16.0	15.0	19.0	12.0	11.0	14.0	2.0	10.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
21407	22820	794.0	3944.0	4.0	649.0	495.0	529.0	679.0	479.0	1066.0	659.0	654.0	743.0
21408	22820	997.0	2947.0	3.0	794.0	649.0	495.0	529.0	679.0	479.0	1066.0	659.0	654.0
21409	22820	790.0	2157.0	2.0	997.0	794.0	649.0	495.0	529.0	679.0	479.0	1066.0	659.0
21410	22820	657.0	1500.0	1.0	790.0	997.0	794.0	649.0	495.0	529.0	679.0	479.0	1066.0
21411	22820	860.0	640.0	0.0	657.0	790.0	997.0	794.0	649.0	495.0	529.0	679.0	479.0

Figure 5.11: Dataset of vehicles for performing the training

5.2.2 Regression Algorithm

The objective is that of applying regression to predict the time until the next due maintenance. The authors train and test both linear and non-linear regression models, each category showing advantages and disadvantages: linear models have smaller complexity but are not ideal if the purpose is that of modeling non-stationary trends. For this reason, they exploit four regression algorithms, i.e. a Linear Regressor, a Support Vector Regressor, a Random Forest Regressor and a Gradient Boosting. For each of them they perform a grid search to find the optimal hyper-parameter configuration that manages to fit the input data distribution.

Considering the fact that the Random Forest one performed best for the task, we will focus on this specific algorithm. Hyperparameter tuning is a useful step to optimize the Random Forest model. For this specific algorithm, hyperparameters include the number of decision trees in the forest, the maximum number of features considered by each tree when splitting a node, the maximum number of levels in each decision tree, the minimum number of data points placed in a node before the node is split and others. These can all be tuned when using SciKit-Learn, which implements a set of sensible default hyperparameters for all models, even if they are not guaranteed to be optimal for a problem. For this reason, it can be useful to perform a Grid Search by explicitly specify every combination to find out the optimal one. In general, the most important settings are the number of trees in the forest and the number of features considered for splitting at each leaf node.

For our case study, we perform regression both exploiting grid search to find their optimal configuration and without grid search, using the default sensible values ensured by SciKit-Learn. Due to the high computational time of the grid search execution, and to the small differences in terms of errors between the two strategies, we choose to use the deafult ones.

5.2.3 Results

The Random Forest model is finally trained and tested as indicated in Section 5.2.1. The prediction results are shown in Figure 5.12 and also some meaningful metrics are evaluated to support the considerations. In particular, both the MSE and the R2 value are calculated as shown in Section 4.2.1. The MSE is equal to 391.27, which is a reasonable value considering the fact that the R2 score is equal to 0.80, thus quite close to the ideal value of 1. R2 is a measure of goodness of fit, and it represents the proportion of variance in the dependent variable that is explained by the model. Thus, by looking at the plot, we can see that the results are coherent, since the observations are quite close to the model's predictions, thus they are close to the line of best fit.

Starting from the prediction results, it could be possible to evaluate for each chosen time interval how many vehicles undergo maintenance, calculating their expected value and standard deviation from which we derive the non-stationary demand. For this thesis work, we focused mainly on the short cycles of maintenance, as the provided dataset contained few data about long maintenance ones, although we still present a model for non-stationary demand management that could be used with other datasets.

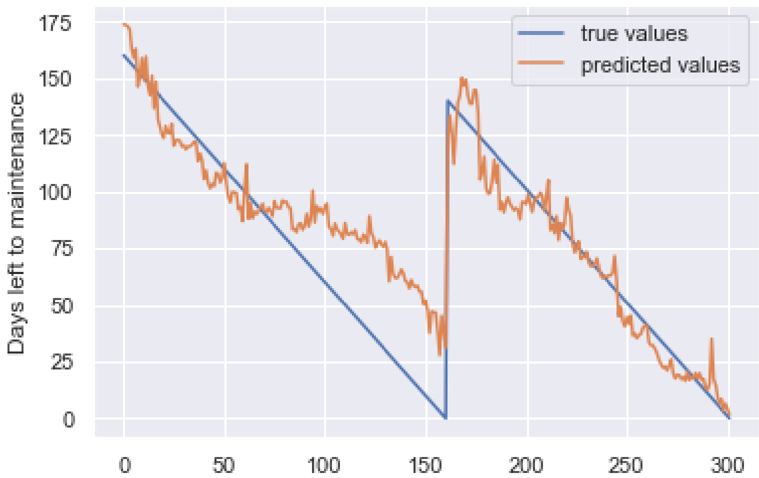


Figure 5.12: Prediction line vs true values

5.3 The Optimization Model

As stated earlier, the final phase involves the optimization of the overall management of the inventory in order to meet all the requirements related to low costs and high customers' satisfaction.

Taking into account the requirements mentioned in Section 1.1.1, and due to the necessity of dealing with a complex and realistic problem, it is straightforward to think of implementing an OR stochastic model. Our objective is essentially that of finding an optimal solution for the allocation of resources, taking into account objectives, constraints and input parameters.

5.3.1 Instruments

Optimization models can be formulated and solved with the help of several useful tools, from libraries to different kind of solvers. In the following we list the instruments chosen to build up and solve our mathematical program (Figure 5.13):

- **Pyomo** [58], a Python-based, open-source optimization modeling language with a diverse set of optimization capabilities, allowing to load data from many sources. Using Pyomo, a user can describe optimization model in a flexible way by specifying decision variables, constraints, and an optimization objective.

- **Couenne** [59], an Open Source algorithm for solving Mixed-Integer Nonlinear Programming (MINLP) problems. It implements linearization, bound reduction, and branching methods within a Branch&Bound framework. It is distributed on Coin-OR under the Eclipse Public License (EPL).

Everything is implemented exploiting Jupyter Notebook, allowing to run the optimization models also on SageMaker.

Figure 5.13: Logos of the used tools and solver ©

5.3.2 Problem Statement

Managing inventories in the industrial vehicles domain is a challenging task, most of all for the uncertain nature of the spare parts demand, which often shows an intermittent pattern.
In this section we present a Stochastic Programming model for minimizing the inventory holding costs of spare parts inventory and at the same time avoiding vehicle offline periods when the requested items are not immediately available. The choice of adopting this kind of model is mainly driven by the necessity of considering the lack of knowledge of the demand, which is a quantity that may become known only during the decision making process evolution. Spare parts demand is indeed represented as a random variable and its true value becomes known only after its realization. For this reason, the formulation introduced and explained in Section 4.1.6 is adopted.

In this problem the spare parts will be ordered on a regular basis, indicatively once every one or two weeks, and they will be received after the lead time related to each of them. This characteristic makes the lead time non negligible.

The inventory dimensions are of course limited, since a single warehouse cannot store an infinite number of components. Thus, in our model we must also take into account the fact that a maximum storage bound has to be respected. Together with that, since the model is meant to hold independently from the moment in time in which the program runs, also any quantity of product already stored in the warehouse must be specified.

Moreover, since the objective is that of minimizing the inventory holding costs, it is important to know how much it will be paid for a specific spare part to be stored in the warehouse. The holding costs are specific for each spare part, because some components may have some small costs for the storage while others may be quite expensive.
To model, instead, the minimization of the offline period, we introduce a shortage cost that can be seen as a penalty faced when a product's demand is not satisfied. Also in this case this is component-dependent. The optimization is performed by considering an holding/shortage cost ratio, since we do not have a precise available value for them but just a proportion.

Finally, due to its uncertain nature, we consider the demand distribution for each spare part and from it we derive a set of possible realization scenarios to be given as input to our model. In the following we will analyse in detail the mathematical model formulation for the two cases under study, i.e. stationary and non-stationary demand.

5.3.3 Stationary Spare Parts Demand

The first case under study is the one in which the demand distribution for the components does not change over time, meaning that in between the placement of the orders and their reception the demand is always stationary.

To model the problem at hand, we propose a Mixed-Integer Non Linear (MINLP) Two-Stage stochastic model whose decision variables are grouped into first-stage decisions that must be made prior to the realization of the uncertainty (usually called "here-and-now") and second-stage decisions (so-called "wait-and-see") that are the result of the realization of the uncertainty. The advantage of this type of framework is that we can represent any stochastic process, as long as it can be approximated by a finite number of realisations.

In this context, we want to make first-stage decisions that must hold for all the considered scenarios such that the costs associated with all variables are optimized. For the problem under study, the first-stage decisions are represented by the quantity to order for each spare part, that will be delivered after its own lead time, and by a target inventory level for each component that represent the ideal quantity to keep in the inventory to ensure the demand satisfaction. The idea of adopting a target level comes from the inventory control policies already present in the literature, and we can easily adapt it to our use case using it as a "safe stock" to keep to guarantee an equilibrium at steady-state.

Given those decisions, the second-stage ones are those referring to the daily operation of the system, under the scenarios considered in the model. In fact, they are scenario-dependent.

Notice that the implemented mathematical model must cover different time instants as the optimal quantity to find is not delivered immediately but only after the respective lead time, which may cover an arbitrary number of time steps in the time horizon.

We will start by giving the mathematical formulation of the problem under study, listing its parameters and decision variables, followed by the objective function and constraints definitions and explanation.

Sets

First of all, we need to define the employed sets to index variables and parameters:

- Number of spare parts $c \in C$
- Number of scenarios $\xi \in \Xi$
- Time period $t \in T$

The time period is needed because we are performing an optimization over multiple time instants, due to the non-negligible lead time that will cover different time steps for each component.

Parameters

Parameters are those data that are provided as input to the model. They are usually obtained from the company or derived from statistical analysis:

- Inventory holding cost $h_c \quad \forall c \in C$.
- Shortage cost $w_c \quad \forall c \in C$
- Lead time $l_c \quad \forall c \in C$
- Initial inventory quantity $i_c \quad \forall c \in C$
- Maximum inventory capacity M
- Demand for spare part $D_{c,\xi} \quad \forall c \in C, \quad \forall \xi \in \Xi$
- Demand probability for each scenario $P_{c,\xi} \quad \forall c \in C, \quad \forall \xi \in \Xi$

As previously mentioned, the inventory holding and shortage costs are component-dependent, as also the lead time which is specific for each spare part. We also need to define the initial state of the inventory at the start of the optimization. All the quantities mentioned up to now are deterministic and known. The demand values and probabilities are obtained from the distribution of the spare parts demand, as seen in Section 5.1.5.

Decision variables

Decision variables are divided into first stage and second stage ones, according to whether they depend on the scenario or not.

- Quantity to order for a specific component that will arrive after the shipment time $Q_{c,t-l_c} \quad \forall c \in C, \quad \forall t \in T$.
- Target level of inventory for a specific component $S_c \quad \forall c \in C$.
- Inventory quantity for a component at a specific time instant $I_{t,c,\xi} \quad \forall t \in T, \quad \forall c \in C \quad \forall \xi \in \Xi$.
- Unsatisfied demand for a component at a specific time instant $F_{t,c,\xi} \quad \forall t \in T, \quad \forall c \in C \quad \forall \xi \in \Xi$.

- Binary variable to indicate if there is shortage or not $Z_{t,c,\xi} \quad \forall t \in T, \quad \forall c \in C \quad \forall \xi \in \Xi$.

As a matter of fact the first stage decision variables are $Q_{c,t-l_c}$ and S_c, while the second stage ones are $I_{t,c,\xi}$, $F_{t,c,\xi}$ and $Z_{t,c,\xi}$. The latter is needed just to force the model to put the shortage equal to zero if the inventory is greater than zero. In particular, it is equal to 0 if there is shortage, and equal to 1 otherwise.

Model

The goal of the implemented model is that of finding the optimal quantity to order and to keep in stock so that the inventory holding costs are minimized as long as the customers' satisfaction is ensured, meaning that we must also minimise the shortage costs. To do so, we derive a two stage optimization with two main objectives:

$$\min_{Q_{c,t-l_c}} \quad \sum_{\xi}\sum_{c} P_{c,\xi}[\sum_{t} h_c I_{t,c,\xi} + w_c F_{t,c,\xi}] \tag{5.2}$$

$$\min_{S_c} \quad |\sum_{\xi}\sum_{c} P_{c,\xi} I_{t,c,\xi} - i_c| \tag{5.3}$$

$$\text{s.t.} \quad I_{t,c,\xi} = (I_{t-1,c,\xi} + Q_{c,t-l_c} - D_{c,\xi})Z_{t,c,\xi} \qquad \forall t \in T,\ \forall c \in C,\ \forall \xi \in \Xi \tag{5.4}$$

$$F_{t,c,\xi} = (D_{c,\xi} - I_{t-1,c,\xi} - Q_{c,t-l_c})(1 - Z_{t,c,\xi}) \qquad \forall t \in T,\ \forall c \in C,\ \forall \xi \in \Xi \tag{5.5}$$

$$\text{where} \quad I_{t-1,c,\xi} = (I_{t-2,c,\xi} + Q_{c,t-l_c-1} - \mathbb{E}(D_{c,\xi}))Z_{t-1,c,\xi} \tag{5.6}$$

$$M \geq \sum_{c} I_{t,c,\xi} \qquad \forall t \in T,\ \forall \xi \in \Xi \tag{5.7}$$

$$Q_{c,t-l_c} \geq 0 \qquad \forall t \in T,\ \forall c \in C \tag{5.8}$$

$$S_c \geq 0 \qquad \forall c \in C \tag{5.9}$$

$$I_{t,c,\xi} \geq 0 \qquad \forall t \in T,\ \forall c \in C,\ \forall \xi \in \Xi \tag{5.10}$$

$$F_{t,c,\xi} \geq 0 \qquad \forall t \in T,\ \forall c \in C,\ \forall \xi \in \Xi \tag{5.11}$$

$$Z_{t,c,\xi} \in \{0,1\} \qquad \forall t \in T,\ \forall c \in C,\ \forall \xi \in \Xi \tag{5.12}$$

Expression (5.2) consists of the first objective function to be minimised. It involves the holding and shortage costs and it is formulated as the deterministic equivalent of the stochastic problem as it involves the probabilities for each scenario to occur. This is done because the resulting second stage decision variables are scenario dependent, and we want to minimize our problem for all the possible scenarios that could occur. Notice that the function must be optimized for all the involved time instants, as we want to ensure that for all the days in between the reception of the ordered quantity $Q_{c,t-l_c}$ the inventory costs are minimized. For this reason, the optimization follows a sort of iterative procedure in which at each time instant we evaluate the constraints (5.4) and (5.5): the first states that if shortage does not happen, then the inventory value at the current time instant is equal to the inventory state at the previous one, plus any eventual quantity of product that was ordered l_c lead times before, minus the satisfied demand for that specific occurring scenario; the second, instead, tells us that if there is a shortage the unmet demand $F_{t,c,\xi}$ must be equal to the realized demand in that scenario minus any eventual stock still present in the inventory, which involves also the possibility that previously made orders have arrived.

Constraint (5.7) is needed to ensure that the maximum inventory capacity is not exceeded.

Notice that, as stated in (5.6), since we are optimizing the variables at each time instant, the previous inventory state is itself a variable that is obtained by going backwards to all the previous time instants that have been considered. In particular, $I_{t-1,c,\xi}$ should be the state of the inventory at the previous time instant, however due to the uncertain nature of the demand, we are not sure about which scenario is occurring in that specific moment in time, and it is not said that the demand scenario we are considering at time t is the same of that at time t-1. For this reason, the previous inventory state is always evaluated by taking into account the **expected value of the demand** over all the possible occurring scenarios.

This necessary assumption leads to the importance of establishing a target inventory level S_c in our model. This variable represents a quantity that we wish to reach such that the demand is always satisfied even if it could oscillate around the target value. To ensure this equilibrium, as we are consideing the expected value of the demand at each time instant, S_c should be the same at the end of every time period. Thus our new objective becomes that of finding the optimal value of the target in order to respect this condition.

To find out which is the optimal target value that allows to reach an equilibrium situation, we suppose that also the quantity ordered and arrived at every time instant must be the same. This comes from the assumption that, if we consider the expected value of the demand and we have reached an equilibrium situation, then in the past we must have made always the same decisions about the quantity to order.

We devise a specific strategy to obtain the final result of our optimization:

1. First of all, we exploit the first optimization in (5.2) to find the optimal value of $Q_{c,t-l_c}$, which represents a quantity that we want to order right now but that will be delivered only l_c times afterwards.

2. We set the values of the quantities arrived at the previous time instants equal to an arbitrary value (for example zero).

3. We proceed with the first stage of the optimization, evaluating the optimal $Q_{c,t-l_c}$ value, which will be of course highly influenced by the fact that the previous quantities have been set to a low value.

4. After the first optimization, the obtained value is shifted and assigned to the value of $Q_{c,t-l_c+1}$, which is the quantity assumed to be received at the previous time instant. The optimization is repeated.

5. Steps 3 and 4 are repeated until the values of the purchased quantities are equal among them, reaching a sort of convergence after the first oscillations to adjust the ideal quantities to buy at every time step.

6. Once the quantities are the same, we keep them fixed and we continue the optimization with the second stage (5.3), incrementing or decrementing the target value at each iteration according to how close the expected value of the inventory at the current time instant and the initial inventory quantity are. The procedure for solving this second step is explained in Algorithm 1.

As long as the target value for each component is not found, we evaluate the expected value of the inventory after each optimization and we calculate the difference between its value and the initial inventory one. If the difference at the current iteration is smaller than the previous one, it means that there is still the possibility to improve the model, thus we check if the expected value of the inventory is greater or smaller than the initial inventory: in the first case, the target value is incremented and the initial inventory is put equal to it, in the second case it is decremented. If, instead, the new difference in the inventory quantity is greater than the previous one, it means that we have reached in the previous iteration the reasonable target value, and if the difference in the inventory is reasonably small we can set S_c to the optimal found value.

Algorithm 1 Optimization of the target value S_c

$S_c \leftarrow 0$
$c\_found \leftarrow$ False
Initialize $\mathbb{E}(I_{t,c,\xi})$
$previous\_inventory\_difference \leftarrow 0$
while all($c\_found =$ False) **do**
    Optimize the model
    Evaluate $new\_inventory\_difference = |\mathbb{E}I_{t,c,\xi} - i_c(t)|$
    if $new\_inventory\_difference \geq previous\_inventory\_difference$ **then**
        if $new\_inventory\_difference \leq 0.5$ **then**
            $S_c \leftarrow i_c(t-1)$
            $c\_found =$ True
        else
            if $\mathbb{E}I_{t,c,\xi} > i_c(t)$ **then**
                $S_c+ = 1$
            else if $\mathbb{E}I_{t,c,\xi} < i_c(t)$ **then**
                $S_c- = 1$
            end if
            $previous\_inventory\_difference = |\mathbb{E}I_{t,c,\xi} - i_c(t)|$
            $i_c = S_c$
        end if
    end if
end while

The main advantage of this optimization model is that it does not need to be solved multiple times as long as the demand remains stationary over the time horizon. Until the demand holds, the procedure must not be repeated. As a result, if it is convenient to buy stocks because the holding costs are smaller than the shortage ones, the target will be incremented and the company will be covered against the unlucky event in which the realized demand is slightly different from the expected one. On the contrary, if the penalty costs are smaller than the inventory ones, the target inventory will remain equal to zero. It is a reasonable outcome since the costs will be minimized due to the inconvenience of keeping stocks in the inventory, and the company will just have to order the optimal quantity $Q_{c,t-l_c}$. Figure 5.14 shows how the values of the objective function and of the target inventory level change as the ratio between the holding and the shortage costs changes too. We keep the inventory holding cost fixed and equal to 1 and we just change the proportion with respect to the shortage, varying from 0.01 to 2.0. As expected, the objective function grows incrementally as the penalty cost does, while the target value performs a step increment as it suddenly changes from 0 to the optimal value found.

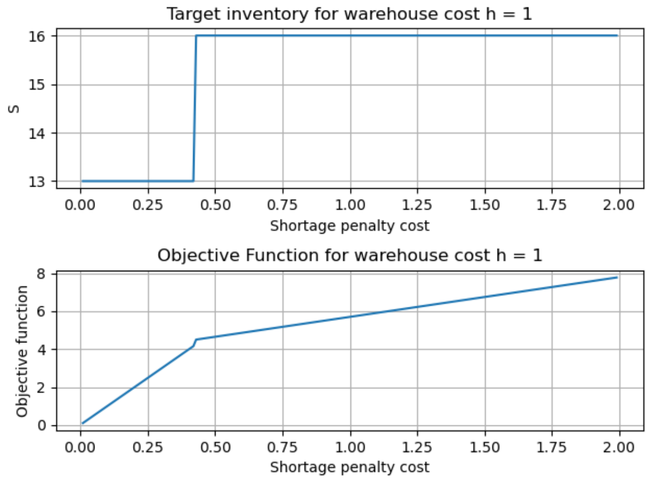


Figure 5.14: Effects of the holding/shortage ratio on the target S_c and the objective function

5.3.4 Non-Stationary Spare Parts Demand

The second case under study is the one involving the non-stationary nature of the demand for the spare parts. As anticipated in Section 5.1.6 this kind of behaviour can be derived from the prediction step of this thesis project: for every time instant we will have a different demand for the spare quantities, each with their own uncertainty, and as a consequence the quantity to order will change at every interval of the time horizon we are considering. Due to this change in the assumption, also the mathematical model must be modified to cope with it.

We still propose a Two-Stage stochastic model whose decision variables are grouped into first-stage decisions that must be made prior to the realization of the uncertainty and second-stage decisions that are the result of the realization of the uncertainty. In this context, however, it makes no sense to consider a target inventory level to reach, as the equilibrium condition ensured by the stationarity of the demand does not hold anymore. As a consequence, the target would not be the same at every time step, thus we just keep the quantity to order to for each component $Q_{c,t-l_c}$ as first stage decision variable. Also in this case the mathematical model covers different time instants.
We will present now the formulation of the problem under study, listing its parameters and decision variables, followed by the objective function and constraints definitions and explanation.

Sets

First of all, we need to define the employed sets to index variables and parameters:

- Number of spare parts $c \in C$
- Number of scenarios $\xi \in \Xi$
- Time period $t \in T$

The time period is needed because we are performing an optimization over multiple time instants, due to the non-negligible lead time that will cover different time steps for each component.

Parameters

Parameters are those data that are provided as input to the model. They are usually obtained from the company or derived from statistical analysis:

- Inventory holding cost $h_c \quad \forall c \in C$.
- Shortage cost $w_c \quad \forall c \in C$
- Lead time $l_c \quad \forall c \in C$
- Initial inventory quantity $i_c \quad \forall c \in C$
- Maximum inventory capacity M
- Demand for spare part $D_{c,\xi} \quad \forall c \in C, \quad \forall \xi \in \Xi$
- Demand probability for each scenario $P_{c,\xi} \quad \forall c \in C, \quad \forall \xi \in \Xi$

As previously mentioned, the inventory holding and shortage costs are component-dependent, as also the lead time which is specific for each spare part. We also need to define the initial state of the inventory at the start of the optimization.

Decision variables

Decision variables are divided into first stage and second stage ones, according to whether they depend on the scenario or not.

- Quantity to order for a specific component that will arrive after the shipment time $Q_{c,t-l_c} \quad \forall c \in C, \quad \forall t \in T$.
- Inventory quantity for a component at a specific time instant $I_{t,c,\xi} \quad \forall t \in T, \quad \forall c \in C \quad \forall \xi \in \Xi$.

- Unsatisfied demand for a component at a specific time instant $F_{t,c,\xi} \quad \forall t \in T, \quad \forall c \in C \quad \forall \xi \in \Xi$.
- Binary variable to indicate if there is shortage or not $Z_{t,c,\xi} \quad \forall t \in T, \quad \forall c \in C \quad \forall \xi \in \Xi$.

As a matter of fact the first stage decision variable is just $Q_{c,t-l_c}$, while the second stage ones remain $I_{t,c,\xi}$, $F_{t,c,\xi}$ and $Z_{t,c,\xi}$. The latter is needed just to force the model to put the shortage equal to zero if the inventory is greater than zero. In particular, it is equal to 0 if there is shortage, and equal to 1 otherwise.

Model

The goal of the implemented model is that of finding the optimal quantity to order so that the inventory holding costs are minimized as long as the customers' satisfaction is ensured, meaning that we must also minimise the shortage costs. To do so, we derive the following model:

$$\min_{Q_{c,t-l_c}} \quad [\sum_{\xi}\sum_{c} P_{c,\xi}[\sum_{t}(h_c I_{t,c,\xi} + w_c F_{t,c,\xi})\alpha^{-t}]] \tag{5.13}$$

$$\text{s.t.} \quad I_{t,c,\xi} = (I_{t-1,c,\xi} + Q_{c,t-l_c} - D_{c,\xi})Z_{t,c,\xi} \qquad \forall t \in T,\ \forall c \in C,\ \forall \xi \in \Xi \tag{5.14}$$

$$F_{t,c,\xi} = (D_{c,\xi} - I_{t-1,c,\xi} - Q_{c,t-l_c})(1 - Z_{t,c,\xi}) \qquad \forall t \in T,\ \forall c \in C,\ \forall \xi \in \Xi \tag{5.15}$$

$$\text{where} \quad I_{t-1,c,\xi} = (I_{t-2,c,\xi} + Q_{c,t-l_c-1} - \mathbb{E}(D_{c,\xi}))Z_{t-1,c,\xi} \tag{5.16}$$

$$M \geq \sum_{c} I_{t,c,\xi} \qquad \forall t \in T,\ \forall \xi \in \Xi \tag{5.17}$$

$$Q_{c,t-l_c} \geq 0 \qquad \forall t \in T,\ \forall c \in C, \tag{5.18}$$

$$I_{t,c,\xi} \geq 0 \qquad \forall t \in T,\ \forall c \in C,\ \forall \xi \in \Xi \tag{5.19}$$

$$F_{t,c,\xi} \geq 0 \qquad \forall t \in T,\ \forall c \in C,\ \forall \xi \in \Xi \tag{5.20}$$

$$Z_{t,c,\xi} \in \{0,1\} \qquad \forall t \in T,\ \forall c \in C,\ \forall \xi \in \Xi \tag{5.21}$$

Expression (5.13) consists of the objective function to be minimised. With respect to Section 5.3.3 the formulation is slightly modified by adding a coefficient α^{-t}. This factor is needed since we are coping with predicted quantities that will have a growing uncertainty, directly proportional to the amount of time in the future we are considering for the forecast. This value is elevated to minus t because we want to give less importance to the farthest predictions, thus it performs as a weight for the different predictions in time. As a result the error will grow, but the importance of the forecast quantity will decrease. All the other constraints have the same meaning as for the stationary case.

Contrary to the previously analysed model, this one needs to be run multiple times to give the optimal quantities over a specific time period.

5.3.5 Results

After having processed all the input data provided by the company and implemented the two optimization models to cope with the stationary and non-stationary nature of the spare parts demand, some results can be shown. In particular, we will start by analysing the results of two optimization runs for both the models; then we will focus on the computational complexity of the two programs, highlighting the relevant variables that influence it; finally, we will underline how our model is able to provide improved results in terms of minimization of costs with respect to some naive baselines.

Stationary Demand Results

The input data values used for the model are summarised in Table 5.6. Notice that the demand and demand probability values are obtained from the discretization of the true distribution of data obtained in Section 5.1.5. In particular, we use an order periodicity of 2 weeks.

Input data	Value
# spare parts	2
# scenarios	3
Lead time ∀ items	[2, 3]
Initial inventory ∀ items	[0, 0]
Inventory Cost ∀ items	[1, 1]
Shortage Cost ∀ items	[10, 10]
Maximum Inventory Capacity	200
Demand item 1	[2, 4, 6]
Demand item 2	[1, 3, 5]
Demand Probabilities item 1	[0.55, 0.35, 0.10]
Demand Probabilities item 2	[0.60, 0.25, 0.15]

Table 5.6: Input data values for the stationary optimization model

We make the optimization model run and in a short amount of time the resulting ideal target inventory is found. In particular, Table 5.7 shows the inventory and shortage levels and the ordered quantities for every time instant at the last iteration, to have a proof of the goodness of the results.

As can be seen, the shortage is zero in almost all cases, and it is an evidence of the goodness of the model because the shortage costs are 10 times higher than the inventory holding ones, thus leading to the tendency of keeping more in stock. The quantities to buy for each component are all very similar, and it is the proof

$[t, c, \xi]$	$I_{t,c,\xi}$	$F_{t,c,\xi}$	$Q_{c,t-l_c}$
$[0, 1, 1]$	4	0	3.90
$[0, 1, 2]$	2	0	
$[0, 1, 3]$	0	0	
$[0, 2, 1]$	3.5	0	2.55
$[0, 2, 2]$	1.5	0	
$[0, 2, 3]$	0	0.5	
$[1, 1, 1]$	5	0	4.10
$[1, 1, 2]$	3	0	
$[1, 1, 3]$	0	0	
$[1, 2, 1]$	4.14	0	2.65
$[1, 2, 2]$	2.14	0	
$[1, 2, 3]$	0.14	0	
$[2, 1, 1]$	4	0	3.85
$[2, 1, 2]$	2	0	
$[2, 1, 3]$	0	0	
$[2, 2, 1]$	5	0	2.75
$[2, 2, 2]$	3	0	
$[2, 2, 3]$	1	0	
$[3, 2, 1]$	4	0	2.55
$[3, 2, 2]$	2	0	
$[3, 2, 3]$	0	0	

Table 5.7: Results for the stationary optimization model with holding-shortage cost ratio equal to 1:10

that the equilibrium situation has been reached and the ideal target value can be found. In the following we list the remaining results, that is the ideal value of the target inventory level S_c that has been found for both components, the objective function value, the number of optimization iterations that the model performed for obtaining the target and the computational time both for a single optimization and for completing the whole algorithm:

- $S_1 = 3$.
- $S_2 = 3$.
- Objective function value = 22.53 euros.
- Number of total optimization iterations = 19.
- Computational time for a single optimization in seconds = 0.88 s.

- Computational time for the whole algorithm in seconds = 41.42 s.

The overall results show that the model achieves the optimal target value result in a reasonable amount of time and ensures a quite good result in terms of monetary expense for the inventory management.

The same optimization can be performed by changing the ratio between the shortage and the inventory holding costs. In fact, keeping the exact same input values as in Table 5.6 and just changing the holding costs to $[10, 10]$ and the penalty to [1,1] we will obtain the results in Table 5.8. As a matter of fact, the model will tend to buy always the minimum possible quantity to ensure the satisfaction of the most likely demand. For this reason, the target inventory level for both components S_c is equal to 0.

- S_c for the first spare part = 0
- S_c for the second spare part = 0
- Objective function value = 7.80 euros
- Number of total optimization iterations = 5
- Computational time for a single optimization in seconds = 0.50 s
- Computational time for the whole algorithm in seconds = 31.71 s

The number of necessary iterations for getting the optimal result only corresponds to the ones needed to find the equal value of the quantity to buy.

$[t, c, \xi]$	$I_{t,c,\xi}$	$F_{t,c,\xi}$	$Q_{c,t-l_c}$
$[0, 1, 1]$	0	0	2
$[0, 1, 2]$	0	2	
$[0, 1, 3]$	0	4	
$[0, 2, 1]$	0	0	1
$[0, 2, 2]$	0	2	
$[0, 2, 3]$	0	4	
$[1, 1, 1]$	0	0	2
$[1, 1, 2]$	0	2	
$[1, 1, 3]$	0	4	
$[1, 2, 1]$	0	0	1
$[1, 2, 2]$	0	2	
$[1, 2, 3]$	0	4	
$[2, 1, 1]$	0	0	2
$[2, 1, 2]$	0	2	
$[2, 1, 3]$	0	4	
$[2, 2, 1]$	0	0	1
$[2, 2, 2]$	0	2	
$[2, 2, 3]$	0	4	
$[3, 2, 1]$	0	0	1
$[3, 2, 2]$	0	2	
$[3, 2, 3]$	0	4	

Table 5.8: Results for the stationary optimization model with holding-shortage cost ratio equal to 10:1

Non-Stationary Demand Results

The input data values used for running the non-stationary demand model are summarized in Table 5.9. Notice that the input values are quite similar to the ones of the stationary demand, however in this case we add the possibility to insert the previously ordered quantities that are meant to arrive after the corresponding lead time, since we are not imposing them anymore as in the former case. Moreover, the demand and demand probabiliy values are different for each time interval, as the demand changes over time. These values are meant to be obtained from the distribution of the predicted demand, where for each time instant we will have the expected value of the requested spare part with the corresponding standard deviation.

Results obtained from this optimization show that if no orders are placed before the moment in which we run the algorithm, thus if the previously ordered quantities

Input data	Value
# spare parts	2
# scenarios	2
α	0.7
Lead time ∀ items	[2, 2]
Initial inventory ∀ items	[0, 0]
Inventory Cost ∀ items	[1, 1]
Shortage Cost ∀ items	[10, 10]
Maximum Inventory Capacity	200
Previously ordered quantities for item 1	[0, 0]
Previously ordered quantities for item 2	[0, 0]
Demand item 1 at each time step	[4, 6], [6, 10], [1, 4], [3, 5]
Demand item 2 at each time step	[1, 3], [2, 5], [2, 7], [3, 6]
Demand Probabilities item 1	[0.70, 0.30],[0.25, 0.75], [0.10, 0.90], [0.40, 0.60]
Demand Probabilities item 2	[0.60, 0.40],[0.30, 0.70], [0.15, 0.85], [0.35, 0.65]

Table 5.9: Input data values for the non-stationary optimization model

are equal to 0, of course no quantity will be delivered at the first time instants, and the realized demand will result as shortage. However, in the following time instants, the quantities ordered at time 0 and delivered at time 2 are correctly optimized. This happens only at the start of the simulation, because contrary to the stationary model that needs to be run only once, this one must be run multiple times for covering the time horizon that we want to predict. Evidence is shown in Table 5.10, where the quantities to buy reflect the most likely demand that will occur. The objective function, also for the effect of the discount factor α which is in charge of weighting in a different way the closer prediction in time with respect to the farther ones, is equal to 258.68 euros. The computational time for running the optimization is quite reasonable and it is equal to 1.23 seconds.

Also in this case it is possible to perform the same optimization considering the shortage cost as lower with respect to the holding one. Results (Table 5.11) show that the model will tend to buy the strict minimum in order to keep the inventory as close to zero as possible. In this case, the objective function is equal to 48.48 euros, also as a result of the previously ordered quantities that in this particular case are equal to 0. The computational time does not change significantly with respect to the previous case.

$[t, c, \xi]$	$I_{t,c,\xi}$	$F_{t,c,\xi}$	$Q_{c,t-l_c}$
$[0, 1, 1]$	0	4	0
$[0, 1, 2]$	0	6	
$[0, 2, 1]$	0	1	0
$[0, 2, 2]$	0	3	
$[1, 1, 1]$	0	6	0
$[1, 1, 2]$	0	10	
$[1, 2, 1]$	0	2	0
$[1, 2, 2]$	0	5	
$[2, 1, 1]$	3	0	4
$[2, 1, 2]$	0	0	
$[2, 2, 1]$	5	0	7
$[2, 2, 2]$	0	0	
$[3, 1, 1]$	2	0	3.8
$[3, 1, 2]$	0	0	
$[3, 2, 1]$	3	0	4.3
$[3, 2, 2]$	0	0	

Table 5.10: Results for the stationary optimization model with holding-shortage cost ratio equal to 1:10

Computational Complexity

Computational complexity is one of the most important metrics when evaluating the goodness of an optimization model. In our work we sperimentally evaluate the computational time for both our models with respect to the variation of some meaningful parameters.

For the stationary demand case, we consider the lead time, the number of scenarios and the number of items, both for a single optimization run and for the whole cycle for optimizing the value of the target inventory. Results are shown in Figure 5.15 for the former case and in Figure 5.16 for the latter.

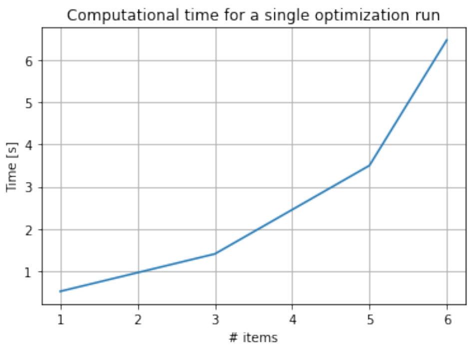
As expected, the computational time is directly influenced by the length of the lead time of a specific component. This is true both in the single optimization case and in the cyclic one, since the time instants to take into account for optimizing our objective function grow proportionally to the lead time. Also the number of scenarios affects the growth of the time to execute the algorithm, which is reasonable as the possible realizations of the demand to take into account increase. Notice that the behaviour of the increment is quite similar for both cases. For what concerns the number of items, also this value highly affects the computational time, and the increment is quite steep. However, it is possible to notice that the

$[t, c, \xi]$	$I_{t,c,\xi}$	$F_{t,c,\xi}$	$Q_{c,t-l_c}$
$[0, 1, 1]$	0	4	0
$[0, 1, 2]$	0	6	
$[0, 2, 1]$	0	1	0
$[0, 2, 2]$	0	3	
$[1, 1, 1]$	0	6	0
$[1, 1, 2]$	0	10	
$[1, 2, 1]$	0	2	0
$[1, 2, 2]$	0	5	
$[2, 1, 1]$	0	0	1
$[2, 1, 2]$	0	3	
$[2, 2, 1]$	0	0	2
$[2, 2, 2]$	0	5	
$[3, 1, 1]$	0	0	3
$[3, 1, 2]$	0	2	
$[3, 2, 1]$	0	0	3
$[3, 2, 2]$	0	3	

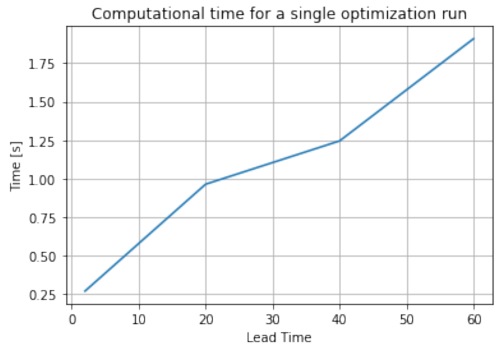
Table 5.11: Results for the non-stationary optimization model with holding-shortage cost ratio equal to 10:1

time to execute the algorithm, also when considering the iterative optimization, is reasonably low, regardless the non linearity of the model's constraint that of course contribute to the increase in time.

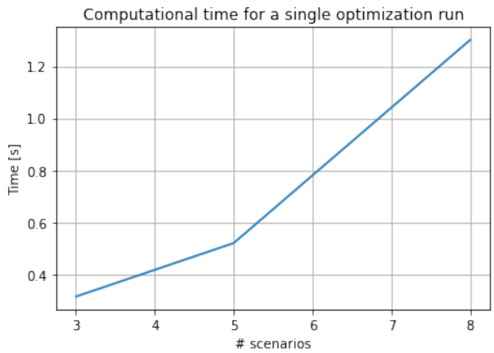
For the non-stationary demand case, instead, we consider the number of scenarios, the number of items, and the number of time instants up to which we perform the prediction of the demand value. In this specific case, we do not have any iterative optimization, thus we will only analyse the computational time for a single optimization. Results are shown in Figure 5.17. Also in this case, the same observations can be made, however we can notice that the time to execute the optimization is longer with respect to the stationary case, maybe due to the complexity of handling the optimization of the quantity to order at every time instant, and not just at the last one anymore.



(a) Number of items

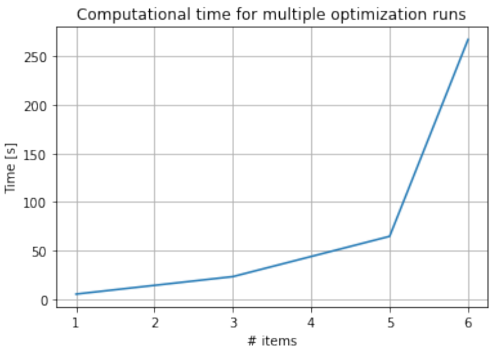


(b) Lead Time

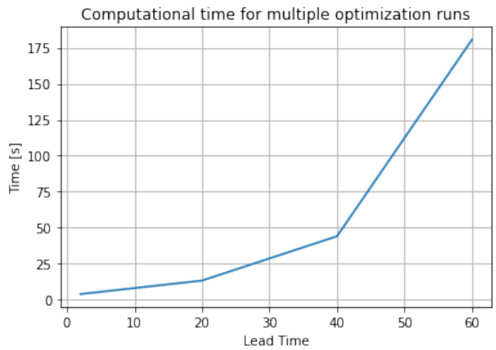


(c) Scenarios

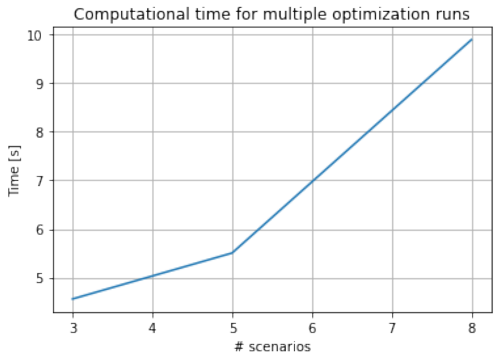
Figure 5.15: Computational time with respect to different parameters variation - One single optimization



(a) Number of items

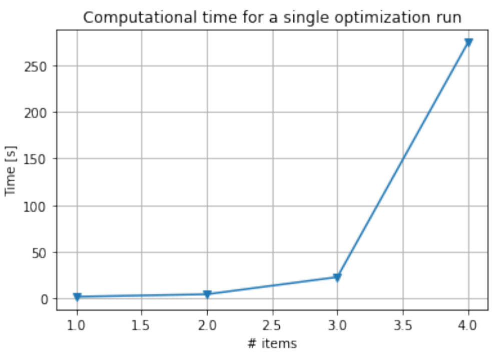


(b) Lead Time

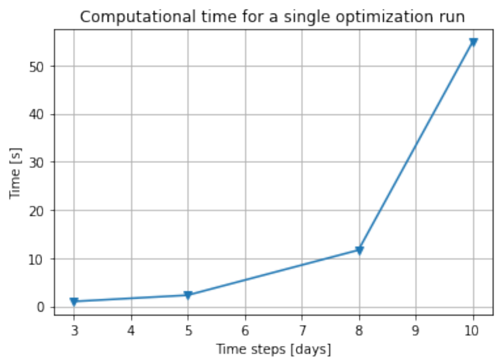


(c) Scenarios

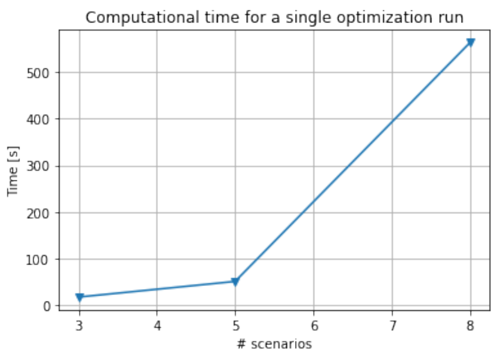
Figure 5.16: Computational time with respect to different parameters variation - Cyclic optimization



(a) Number of items



(b) Time instants



(c) Scenarios

Figure 5.17: Computational time with respect to different parameters variation - Non-Stationary case

Comparison with naive optimization strategies

To conclude our results analysis, we verify the goodness of the proposed solution by performing a comparison with some simple baseline policies. We evaluate the improvement in terms of objective function value, intended as how much money we manage to save by using the optimized strategy.
In particular, we choose three policies:

- Order always the **maximum** possible demand value.
- Order always the **minimum** possible demand value.
- Order always the **mean** value of the demand.

In order to asses how much we manage to save by adopting the optimal devised strategy with respect to each one of these baselines, we adopt a simulative approach: after having performed the optimization for the specified input values, we run 1000 simulations with the obtained best decision variables; at each iteration we randomly extract one of the values among the possible demand scenarios and we evaluate the corresponding cost in terms of euros; finally, we evaluate the average of all the simulation outcomes and we compare it with the naive strategies ones.
This is done both for the stationary and for the non-stationary demand models. The improvements are summarized in Table 5.12 and Table 5.13 both for the case in which the inventory cost is smaller than the shortage one and viceversa.

The optimization for the stationary case is performed considering as input two different items, with lead time equal to 2 and the following demand distribution: [10, 20, 30] with probabilities [0.55, 0.35, 0.10] for the first item and [20, 5, 10] with probabilities [0.60, 0.25, 0.15] for the second one. The resulting target value for both items respectively is $S_1 = 15$ and $S_2 = 5$ in case of holding costs 10 times smaller than penalty ones, while they are of course 0 in the opposite one.

The optimization for the non-stationary case is performed considering two items with lead time equal to 3, over 10 time periods with a demand that varies each time instant with 3 different scenario probabilities.

Model type	Δ wrt max	Δ wrt min	Δ wrt mean
Stationary demand	42,03	381,03	98,92
Non Stationary demand	724,99	941,6	52.7

Table 5.12: Improvement of cost when the holding cost is smaller than the shortage

Results show that in all cases the implemented optimization models bring an improvement in terms of amount of money that is spent to manage the inventory.

Model type	Δ wrt max	Δ wrt min	Δ wrt mean
Stationary demand	1139,25	∼0	352,84
Non Stationary demand	12.734,13	10,43	2623,43

Table 5.13: Improvement of cost when the holding cost is bigger than the shortage

Considering the relative variation in percentage with respect to the maximum, minimum and average policies, the stationary case features an improvement of 35%, 83% and 56%, while the non-stationary case shows an improvement of 55%, 60% and 9%.

Concerning Table 5.13, the relative gains for the stationary demand model are 96%, 0% and 88%, while for the non-stationary one 98%, 6% and 93%.

In particular we notice that, when the inventory holding cost is smaller than the shortage one, the models perform very well in all cases, while when the holding costs overcome the penalty ones the models' objective functions are extremely close to the "minimum-order" naive strategy, as the model tends to buy the least amount of components possible to ensure the demand satisfaction.

Chapter 6

Conclusions

In this thesis work we presented the problem of handling spare parts requests from industrial vehicles undergoing maintenance. The proposed approach consisted of identifying the different scenarios of components' demand and implementing an operational research stochastic model with the aim of minimizing the inventory holding costs while ensuring the components' demand satisfaction.

First, we analysed the importance of inventory management inside the broader context of the supply chain, highlighting the benefits and the challenges of inventory optimization, and we presented the main existing control policies and sources of costs for spare parts management.

Then, we explored the existing literature that analyses the uncertain nature of demand, identifying the main results in the proposed solutions both for stationary and non-stationary demand in relation to the most common inventory control policies. Moreover, we focused on the literature related to spare parts optimization, which requires a specific study, and the application of forecast strategies to handle maintenance prediction, identifying the similarities and differences with our work.

To carry out this study and answer the research questions on which it is based, we exploited the input data about industrial vehicle maintenance history and usage provided by Tierra S.p.A., the company with which we collaborated. These data were used to derive meaningful information for understanding the spare parts demand patterns. As mentioned at the beginning of the work, it is extremely important to correctly manage the uncertain nature of the demand, since it makes it difficult to order the right quantities of spare parts. After having pre-processed the dataset in order to analyse its characteristics, we distinguished two cases: the first one, in which the vehicles seemed to have short maintenance cycles with respect to the lead time, and the second one, in which maintenance cycles were longer, making it useful to perform predictions on the next maintenance. From the first case it was possible to obtain a stationary distribution of the demand, while the second was provided as input to the Random Forest regression algorithm that we used to predict the remaining days until the next maintenance, resulting in a non-stationary demand behaviour.

The presence of the uncertainty generated by the vehicles' maintenance patterns lead to the necessity of formalizing two stochastic optimization models to handle stationary and non-stationary spare parts requests. We implemented two mixed-integer non linear stochastic models: the one handling stationary demand aimed at finding an ideal target inventory level to maintain the optimal equilibrium in the inventory, minimizing its costs yet ensuring demand satisfaction. The other, which coped with non-stationary demand, aimed at optimizing the quantity to order at each time instant.

As a matter of fact, the devised operational research models allowed to provide a way to administrate the inventory in order to avoid long waiting times caused by shortages, exploiting inventory replenishment strategies that permitted to correctly place order quantities.

In addition to that, we gave evidence of the fact that our solution also avoided the overstocking of items in the inventory. In particular, the two implemented models showed an improvement in the minimization of costs with respect to some naive order policies, easy to adopt but negatively affecting holding costs for companies. Although the input data were not the ideal ones for performing predictions, the model implementation presented a high scalability and could be applied to different kinds of datasets, keeping reasonable computational times.

6.1 Future Work

While done with the utmost effort, this thesis work could be further extended and improved in some aspects, as discussed also with the company. For instance, the model can be applied to different categories of vehicles and adapted to the changing necessities of the specific fleet.

Furthermore, in this work we considered the spare parts demand as the only source of stochasticity, but the study can be further extended by taking into account the uncertainty in the shipment time, or considering the variation of the holding and shortage costs over time.

In addition to that, a further step can be performed considering also a shipment cost and the possibility that if it is quite high it could be convenient to aggregate the orders in specific time instants, deriving the information about when the order must be placed.

Bibliography

[1] *Supply chain: what is it? Definition and differences from logistics.* https://www.mecalux.com/blog/supply-chain-definition. 2022 (cit. on p. 4).

[2] *Supply chain: where do our products come from.* https://www.vacanciesinlogistics.com/blog/supply-chain-stages-where-do-our-products-come-from. 2019 (cit. on p. 4).

[3] R. K. Oliver and Melvin D Webber. «Supply-chain management: logistics catches up with strategy». In: 1982 (cit. on p. 4).

[4] Clive L Purchase. *Inventory Optimization: Show Me the Money.* https://www.scmr.com/article/inventory_optimization_show_me_the_money. 2011 (cit. on p. 7).

[5] Edward A. Silver. «Inventory Management: An Overview, Canadian Publications, Practical Applications and Suggestions for Future Research». In: *INFOR: Information Systems and Operational Research* 46.1 (2008), pp. 15–27. DOI: 10.3138/infor.46.1.15. eprint: https://doi.org/10.3138/infor.46.1.15. URL: https://doi.org/10.3138/infor.46.1.15 (cit. on p. 8).

[6] Tim Gallagher, Mark D. Mitchke, and M. C. Rogers. «Profiting from spare parts». In: 2007 (cit. on p. 9).

[7] John E. Boylan and Aris A. Syntetos. «Spare parts management: a review of forecasting research and extensions». In: *IMA Journal of Management Mathematics* 21.3 (Nov. 2009), pp. 227–237. ISSN: 1471-678X. DOI: 10.1093/imaman/dpp016. eprint: https://academic.oup.com/imaman/article-pdf/21/3/227/1828339/dpp016.pdf. URL: https://doi.org/10.1093/imaman/dpp016 (cit. on p. 9).

[8] Ashraf Labib and T Exton. «Spare parts decision analysis-the missing link in CMMS's (Part 1)». In: *Journal of Maintenance & Asset Management* 16.3 (2001), pp. 10–17 (cit. on p. 9).

[9] Tim Exton and Ashraf Labib. «Spare parts decision analysis–The missing link in CMMSs (Part II)». In: *Journal of Maintenance & Asset Management* 17.1 (2002), pp. 14–21 (cit. on p. 9).

[10] Koichi Nakade and Shiori Yokozawa. «Optimization of two-stage production/inventory systems under order base stock policy with advance demand information». In: *Journal of Industrial Engineering International* 12.4 (2016), pp. 437–458 (cit. on p. 11).

[11] SM Samak-Kulkarni and NR Rajhans. «Determination of optimum inventory model for minimizing total inventory cost». In: *Procedia Engineering* 51 (2013), pp. 803–809 (cit. on p. 11).

[12] Sven Axsäter. «A simple procedure for determining order quantities under a fill rate constraint and normally distributed lead-time demand». In: *European Journal of Operational Research* 174.1 (2006), pp. 480–491. ISSN: 0377-2217. DOI: `https://doi.org/10.1016/j.ejor.2005.01.037`. URL: `https://www.sciencedirect.com/science/article/pii/S0377221705001864` (cit. on p. 12).

[13] Hector D Perez, Christian D Hubbs, Can Li, and Ignacio E Grossmann. «Algorithmic approaches to inventory management optimization». In: *Processes* 9.1 (2021), p. 102 (cit. on pp. 12, 13).

[14] S Armagan Tarim and Brian G Kingsman. «Modelling and computing (Rn, Sn) policies for inventory systems with non-stationary stochastic demand». In: *European Journal of Operational Research* 174.1 (2006), pp. 581–599 (cit. on p. 13).

[15] Gitae Kim, Kan Wu, and Edward Huang. «Optimal inventory control in a multi-period newsvendor problem with non-stationary demand». In: *Advanced Engineering Informatics* 29.1 (2015), pp. 139–145. ISSN: 1474-0346. DOI: `https://doi.org/10.1016/j.aei.2014.12.002`. URL: `https://www.sciencedirect.com/science/article/pii/S1474034614001128` (cit. on pp. 14, 15).

[16] Mary Dillon, Fabricio Oliveira, and Babak Abbasi. «A two-stage stochastic programming model for inventory management in the blood supply chain». In: *International Journal of Production Economics* 187 (2017), pp. 27–41 (cit. on pp. 15, 16).

[17] Vidyadhar Kulkarni and Keqi Yan. «Production-inventory systems in stochastic environment and stochastic lead times». In: *Queueing Systems* 70.3 (2012), pp. 207–231 (cit. on p. 16).

[18] A Eaves and B Kingsman. «Forecasting for the ordering and stock-holding of spare parts». In: *Journal of The Operational Research Society - J OPER RES SOC* 55 (Apr. 2004), pp. 431–437. DOI: 10.1057/palgrave.jors.2601697 (cit. on pp. 17, 39).

[19] Qiwei Hu, John E. Boylan, Huijing Chen, and Ashraf Labib. «OR in spare parts management: A review». In: *European Journal of Operational Research* 266.2 (2018), pp. 395–414. ISSN: 0377-2217. DOI: https://doi.org/10.1016/j.ejor.2017.07.058. URL: https://www.sciencedirect.com/science/article/pii/S0377221717307014 (cit. on pp. 17, 18).

[20] Ling Li, Min Liu, Weiming Shen, and Guoqing Cheng. «An improved stochastic programming model for supply chain planning of MRO spare parts». In: *Applied Mathematical Modelling* 47 (2017), pp. 189–207 (cit. on pp. 17, 19).

[21] Rachel Zhang, Wallace Hopp, and Chonawee Supatgiat. «Spreadsheet Implementable Inventory Control for a Distribution Center». In: *J. Heuristics* 7 (Mar. 2001), pp. 185–203. DOI: 10.1023/A:1009613921001 (cit. on p. 18).

[22] Tongdan Jin and Haitao Liao. «Spare parts inventory control considering stochastic growth of an installed base». In: *Computers & Industrial Engineering* 56.1 (2009), pp. 452–460 (cit. on pp. 19, 20).

[23] Haitao Liao, Peng Wang, Tongdan Jin, and Sunny Repaka. «Spare parts management considering new sales». In: *2008 Annual Reliability and Maintainability Symposium*. 2008, pp. 502–507. DOI: 10.1109/RAMS.2008.4925847 (cit. on p. 20).

[24] Kostas-Platon Aronis, Ioulia Magou, Rommert Dekker, and George Tagaras. «Inventory Control Of Spare Parts Using A Bayesian Approach: A Case Study». In: *European Journal of Operational Research* 154 (May 2004), pp. 730–739. DOI: 10.1016/S0377-2217(02)00837-8 (cit. on p. 20).

[25] Federico Perrotta, Tony Parry, and Luis C Neves. «Application of machine learning for fuel consumption modelling of trucks». In: *2017 IEEE International Conference on Big Data (Big Data)*. IEEE. 2017, pp. 3810–3815 (cit. on p. 21).

[26] Seyed Mohammad Rezvanizaniani, Jacob Dempsey, and Jay Lee. «An Effective Predictive Maintenance Approach based on Historical Maintenance Data using a Probabilistic Risk Assessment: PHM14 Data Challenge». In: *International Journal of Prognostics and Health Management* 5 (Nov. 2020). DOI: 10.36001/ijphm.2014.v5i2.2237 (cit. on p. 21).

[27] Silvia Buccafusco, Andrea Megaro, Luca Cagliero, Francesco Vaccarino, Lucia Salvatori, and Riccardo Loti. «Profiling industrial vehicle duties using CAN bus signal segmentation and clustering». In: *EDBT/ICDT Workshops*. 2021 (cit. on pp. 22, 23).

[28] Dena Markudova, Elena Baralis, Luca Cagliero, Marco Mellia, Luca Vassio, Elvio Gilberto Amparore, Riccardo Loti, and Lucia Salvatori. «Heterogeneous Industrial Vehicle Usage Predictions: A Real Case». In: *EDBT/ICDT Workshops*. 2019 (cit. on p. 23).

[29] Sachit Mishra, Luca Vassio, Luca Cagliero, Marco Mellia, Elena Baralis, Riccardo Loti, and Lucia Salvatori. «Machine Learning Supported Next-Maintenance Prediction for Industrial Vehicles.» In: *EDBT/ICDT Workshops*. 2020 (cit. on p. 24).

[30] Dena Markudova, Sachit Mishra, Luca Cagliero, Luca Vassio, Marco Mellia, Elena Baralis, Lucia Salvatori, and Riccardo Loti. «Preventive maintenance for heterogeneous industrial vehicles with incomplete usage data». In: *Computers in Industry* 130 (2021), p. 103468. ISSN: 0166-3615. DOI: `https://doi.org/10.1016/j.compind.2021.103468`. URL: `https://www.sciencedirect.com/science/article/pii/S0166361521000750` (cit. on pp. 24, 25, 59, 66).

[31] *Nist optimization problem definition*. `https://xlinux.nist.gov/dads/HTML/optimization.html` (cit. on p. 36).

[32] *OR Society*. `https://www.theorsociety.com/` (cit. on p. 36).

[33] Andrzej Ruszczyński. *Nonlinear Optimization*. Princeton University Press, 2006. ISBN: 9780691119151. URL: `http://www.jstor.org/stable/j.ctvcm4hcj` (visited on 11/25/2022) (cit. on p. 40).

[34] Sabine Helwig, Falk Hüffner, Ivo Rössling, and Maik Weinard. «Chapter 3. Selected Design Issues». In: Oct. 2010. ISBN: 978-3-642-14865-1. DOI: `10.1007/978-3-642-14866-8_3` (cit. on p. 40).

[35] Selen Cremaschi and John D. Siirola. «Solution Approaches to Stochastic Programming Problems under Endogenous and/or Exogenous Uncertainties». In: *13th International Symposium on Process Systems Engineering (PSE 2018)*. Ed. by Mario R. Eden, Marianthi G. Ierapetritou, and Gavin P. Towler. Vol. 44. Computer Aided Chemical Engineering. Elsevier, 2018, pp. 101–102. DOI: `https://doi.org/10.1016/B978-0-444-64241-7.50011-2`. URL: `https://www.sciencedirect.com/science/article/pii/B9780444642417500112` (cit. on p. 41).

[36] Morris H DeGroot and Mark J Schervish. *Probability and statistics*. Pearson Education, 2012 (cit. on p. 41).

[37] Günter Last and Mathew Penrose. *Lectures on the Poisson process*. Vol. 7. Cambridge University Press, 2017 (cit. on p. 42).

[38] *Poisson distribution and Poisson process*. `https://towardsdatascience.com/the-poisson-distribution-and-poisson-process-explained-4e2cb17d459` (cit. on pp. 43, 44).

[39] Ronald E. Walpole, Raymond H. Myers, Sharon L. Myers, and Keying Ye. *Probability & statistics for engineers and scientists*. 8th. Upper Saddle River: Pearson Education, 2007 (cit. on p. 44).

[40] John R Birge and Francois Louveaux. *Introduction to stochastic programming*. Springer Science & Business Media, 2011 (cit. on p. 45).

[41] Julia L Higle. «Stochastic programming: optimization when uncertainty matters». In: *Emerging theory, methods, and applications*. Informs, 2005, pp. 30–53 (cit. on p. 46).

[42] Claudia Angelini. «Regression Analysis». In: *Encyclopedia of Bioinformatics and Computational Biology*. Ed. by Shoba Ranganathan, Michael Gribskov, Kenta Nakai, and Christian Schönbach. Oxford: Academic Press, 2019, pp. 722–730. ISBN: 978-0-12-811432-2. DOI: https://doi.org/10.1016/B978-0-12-809633-8.20360-9. URL: https://www.sciencedirect.com/science/article/pii/B9780128096338203609 (cit. on p. 48).

[43] Francis Galton. «Regression towards mediocrity in hereditary stature.» In: *The Journal of the Anthropological Institute of Great Britain and Ireland* 15 (1886), pp. 246–263 (cit. on p. 48).

[44] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An Introduction to Statistical Learning: with Applications in R*. Springer, 2013. URL: https://faculty.marshall.usc.edu/gareth-james/ISL/ (cit. on p. 48).

[45] Gilles Louppe. «Understanding random forests: From theory to practice». In: *arXiv preprint arXiv:1407.7502* (2014) (cit. on p. 48).

[46] *Decision Tree*. https://aiaspirant.com/introduction-to-decision-trees/ (cit. on p. 49).

[47] *Amazon Web Service*. https://aws.amazon.com/ (cit. on p. 52).

[48] *Amazon S3*. https://aws.amazon.com/s3/ (cit. on p. 53).

[49] *Amazon Athena*. https://aws.amazon.com/athena/?whats-new-cards.sort-by=item.additionalFields.postDateTime&whats-new-cards.sort-order=desc (cit. on p. 53).

[50] *Amazon SageMaker*. https://www.amazonaws.cn/en/sagemaker/ (cit. on p. 54).

[51] *Pandas library*. https://pandas.pydata.org/ (cit. on p. 54).

[52] *NumPy library*. https://numpy.org/ (cit. on p. 54).

[53] *Seaborn library*. https://seaborn.pydata.org/ (cit. on p. 54).

[54] *Matplotlib library*. https://matplotlib.org/ (cit. on p. 55).

[55] *SciPy library*. https://scipy.org/ (cit. on p. 55).

[56] *SciKit-Learn library*. https://scikit-learn.org/stable/ (cit. on p. 55).

[57] *AWSwrangler library*. https://pypi.org/project/awswrangler/ (cit. on p. 55).

[58] *Pyomo*. http://www.pyomo.org/ (cit. on p. 69).

[59] *Couenne solver*. https://www.coin-or.org/Couenne/ (cit. on p. 69).