FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Machine Learning Models for Predictive Quality

João Nuno Rodrigues Ferreira

Mestrado Integrado em Engenharia Informática e Computação

Supervisor: Prof. João Mendes Moreira

Second Supervisor: Ricardo Teixeira Sousa

July 31, 2021

Machine Learning Models for Predictive Quality

João Nuno Rodrigues Ferreira

Mestrado Integrado em Engenharia Informática e Computação

July 31, 2021

Abstract

Nowadays, companies that manufacture products, regardless of the product, have shown an increased interest in resolving one of their problems: manufacturing products with nonconformities. A nonconformity occurs when there is at least one product parameter that does not meet the intended specifications predetermined by the client. It can be classified as minor, major, and critical with the increase in negative effects on the product. The causes for the occurrence of nonconformities can range from the characteristics of the material to the conditions of the machines. However, they can also be related to the poor execution of the production process or changes in the product's characteristics.

The main purpose of this dissertation is the development of machine learning models capable of predicting whether a batch of manufactured products in a production line will be fabricated with or without nonconformities, considering the possible causes addressed before. The development of successful models can help companies reduce the production of defective products, leading to a decrease in material and energy waste and a reduction in the environmental impact.

A standard CRISP-DM methodology was followed. In an initial phase, the data was collected, preprocessed, and analyzed. Afterward, a phase of new features creation occurred and, henceforth, the dataset was prepared to be used in the training of the machine learning models by making use of several techniques and methods. Finally, parameter tuning was performed to improve the models' performances, resulting in a balanced accuracy average of was 84%. However, the mean precision and recall was of 37% and 70%, respectively. When comparing the AUC metric, the Artificial Neural Network classifier scored 0,91, Logistic Regression scored 0,97 and Random Forest scored 0,99.

Keywords: Industry, Nonconformity, Product Quality, SPC, Data Mining, Machine Learning, Supervised Learning, Prediction, Classification

Resumo

Hoje em dia, as empresas que fabricam produtos, quaisquer que sejam estes, têm demonstrado um crescente interesse na resolução de um dos seus problemas: manufaturar produtos com não-conformidades. Uma não-conformidade ocorre quando existe pelo menos um parâmetro do produto que não vai de encontro à especificação pretendida e pré-determinada pelo cliente. A não-conformidade pode ser classificada como secundária, principal ou crítica, de acordo com o aumento do efeito negativo no produto. As causas para a ocorrência de não-conformidades podem ir desde as caraterísticas da matéria-prima até às condições das máquinas. No entanto, podem também estar relacionadas com uma execução defeituosa do processo de produção ou com uma mudança nas caraterísticas do próprio produto.

O principal objetivo desta dissertação é o desenvolvimento de modelos de *machine learning* que sejam capazes de prever se um lote de produtos manufaturados numa linha de produção serão fabricados com ou sem não-conformidades, tendo em conta as causas referidas anteriormente. O desenvolvimento de modelos bem-sucedidos pode contribuir para a redução da produção de productos com não-conformidades, levando a uma diminuição no gasto de matérias-primas e de energia, bem como a um impacto ambiental mais reduzido.

Foi seguida uma metodologia *CRISP-DM*. Numa fase inicial, os dados foram colecionados, pré-processados e analisados. De seguida, ocorreu uma fase de criação de novos atributos e, daí em diante, o conjunto de dados foi preparado, através de um conjunto de várias técnicas e métodos, para ser usado no treino dos modelos preditivos. Finalmente, efetuou-se uma otimização dos parâmetros dos modelos de forma a melhorar o desempenho destes, resultando numa exatidão média de 84%. No entanto, a média da *precision* e do *recall* for de 37% e de 70%, respetivamente. Na comparação da métrica *AUC*, o classificador *Artificial Neural Network* obteve um valor de 0,91, *Logistic Regression* obteve 0,97 e, por fim, *Random Forest* obteve 0,99.

Keywords: Indústria, Não-Conformidade, Qualidade do Produto, SPC, Data Mining, Machine Learning, Aprendizagem Supervisionada, Previsão, Classificação

Acknowledgements

I would first like to thank my supervisor Prof. João Pedro Mendes Moreira, and co-supervisor Ricardo Teixeira Sousa for the support and guidance during these months in which I developed this dissertation and without whom the result would not be the same for sure.

I would also like to thank FEUP for the ridiculously awesome 5 years I was able to spend while finishing my Master's.

Last but not least, I would like to express my uttermost thanks to both my family (including my cat) and my friends who were always there in the good and bad moments. A special thanks to my mom for the unconditional love she always gave me.

To all of those. Thank you.

"We could change the world forever
We should turn it into something better
Than good."

Mac Miller

Contents

List of Figures

List of Tables

Abbreviations

DM	Data Mining
ML	Machine Learning
QC	Quality Control
SPC	Statistical Process Control
CL	Center Line
UCL	Upper Control Line
LCL	Lower Control Line
OCAP	Out of Control Action Plan
CRISP-DM	Cross-Industry Standard Process for Data Mining
SEMMA	Sample, Explore, Modify, Model, Assess
KDD	Knowledge Discovery in Databases
PCA	Principal Component Analysis
RFE	Recursive Feature Elimination
kNN	k-Nearest Neighbors
SVM	Support Vector Machine
DT	Decision Tree
NB	Naive Bayes
ANN	Artificial Neural Network
PIE	Inspection and Testing Plans
PO	Production Order
SMOTE	Synthetic Minority Oversampling Technique

Chapter 1

Introduction

In this initial chapter, firstly, the context of this dissertation is presented in Section 1.1. This section aims at contextualizing the reader on the topics covered by this work. Secondly, the motivation, Section 1.2, explains why this dissertation is relevant, and the established final objectives are then explained in Section 1.3. Finally, in the brief Section 1.4, the document structure is presented, summarizing the contents of each following chapter.

1.1 Context

We are living in a world which is getting more and more competitive. So nowadays, companies, regardless of their type of business (services, merchandising, manufacturing, and others), are forced to focus on small details that can help them thrive amongst all their competitors.

When it comes to a company related to the manufacturing sector, its product's quality is one of the main distinguishing factors of this company in relation to others in the same industry. Therefore, the consumer decision is highly affected by whether the person considers, or not, the product as having the characteristics desired.

There is no objective definition of what a product needs to have in order to be considered better than a similar product. However, Garvin (1987) has provided eight dimensions of a product that may be evaluated and taken into account when comparing it to another:

- **Performance** — Efficiency fulfilling the job;
- **Reliability** — Frequency of failures;
- **Durability** — Product's lifetime;
- **Serviceability** — Quickness and time/financial effort regarding repairs;
- **Aesthetics** — Visual aspect;

- **Features** — Differentiating characteristics;
- **Perceived Quality** — Reputation of both the company and the product itself;
- **Conformance of Standards** — Precision to meet the standards.

Looking at these dimensions, we understand that **quality is a problem that is related and has to be solved during the manufacturing process**.

Variability is present everywhere since no two objects are exactly the same, even though they may seem very similar. In the manufacturing business, the increase of variability in its products represents a decrease in quality. Variability is related to several slight differences that occur in, for example, raw materials, in the performance of operators, equipment, and machines. When in the presence of high variability, some products may be manufactured with **nonconformities** — a specific type of failure when a product fails to meet at least one specification. Once again, we understand that **the variability and its associated problems not only may but should be solved during the problem manufacture** [33].

1.2 Motivation

Manufacturing is a complex process in which the raw material goes through hundreds of different steps and is operated by several different machines and tools before becoming the final product. Although steel manufacturing in specific is a very advanced industry, mainly due to the ever-increasing global demand for steel over the past few years, nonconformities in this sector's products are still reasonably common, taking into account the precision needed to operate this material. As a result, this industry is susceptible to their products and slight deviations in the process' conditions like pressure, temperature, or cooling flow rate that can very easily lead to nonconformities [16, 46].

Besides the complexity addressed in the previous paragraph, the costs associated with the manufacture of steel products and, in particular, the costs related to the reduction of product quality are in the order of millions of dollars. This type of production is very energy-intensive. Thus, optimizing the process is the key to keep a company competitive by reducing the final output cost. Right now, companies are trying to reduce their production costs by attempting to predict which products are worth going through the process until the end or not. Taking into account all the existent variables, prediction is a difficult task to be carried out. However, the sooner flaws are detected, the more money can be saved from the material in production that does not need to undergo the rest of the production stages because it has already been detected as a probable final product with nonconformities [20, 46]. The advantages of early detection of nonconformities are:

- Improvement in **manufacturing efficiency**;
- **Yield** improvement;
- **Early detection** of weak overall outcomes.

Data Mining, a process used in the extraction of information from large datasets, when applied to a domain like industry, is usually associated with quality control and anomaly detection [46]. Nowadays, industry processes contain many measurement techniques using, for example, soft sensors [3], enabling the storage of vast amounts of data that benefit Data Mining [20]. The dimensions of the datasets not only rule out the possibility that experts perform the prevention and fault detection but also make it a perfect situation for DM and ML techniques to make use of its potential.

It must also be referred that, even though companies can reduce the cost of quality (costs related to the production, identification, avoidance, and even repair of products with nonconformances), it can never be zero. To reach such performance, the costs related to process-control techniques and to the system that collects and analyzes the data would be greater than the actual resulting reduction in the cost of quality. Nevertheless, decreasing the variability can lead to reductions in quality costs by 50% or 60% in organizations where this effort has not yet been made [33]. Last but not least, in addition to the obvious **financial benefit of reducing material and energy waste, companies can also reduce their environmental impact.**

1.3 Objectives

Data Mining and Machine Learning techniques may help QC and anomaly detection experts, as was mentioned in Section 1.2. Once the relevant variables in the production process have been identified, the relationships between those can be determined. These variables are related to the condition of the machines and the characteristics of the product material.

This dissertation proposes the combination of statistical methods and DM and ML techniques to create predictive models that are able to predict the outcome of a batch regarding its conformance in a production line environment. By doing this, there is the possibility of exploring the conditions or ordered sequence of events that led to a nonconforming product. The prediction is based on the variables mentioned in the previous paragraph and will be applied to a steel manufacturing plant. The final goal of this work is to minimize the material and energy waste and, by doing so, also minimizing the environmental impact of manufacturing companies.

1.4 Document Structure

This dissertation is constituted by six chapters, each with a self-explanatory name, as it follows:

- **Introduction** — Chapter 1 introduces the reader to the scope of the work. Includes the context, motivation, objectives, and this document structure;
- **Background** — Chapter 2 describes the contemporaneous state of the art on the subjects related to quality control and machining;
- **Data Mining and Machine Learning** — Chapter 3 contains the discussion about DM and ML methodologies, techniques, methods, and algorithms;

- **Experimental Setup** — Chapter 4 has the explanation about how the work was performed from data collection to model training;
- **Results** — Chapter 5 presents the obtained results of each predictive model that was trained;
- **Discussion and Conclusions** — Chapter 6 concludes the dissertation by briefly reviewing the developed work, interpreting the results, and suggesting the next steps in the development of this work.

Chapter 2

Background

Before developing the predictive models that were briefly explained in Section 1.3, it is of great importance to, first of all, carry out the study of some topics that this work involves since good work requires a solid knowledge base in the topics addressed by it. Taking that into account, an overview of those is performed in this chapter.

2.1 Quality Control

In the industry sector, the quality of companies' products is, if not the most, one of the most decisive factors for the success of that company. If it is able to incorporate in its products the characteristics referred in Section 1.1, there is a high likelihood that the company can prosper in the market segments where it is represented. However, it is understandable that achieving such success is not easy or, otherwise, every company would have manufacturing processes that allowed them to have top tier products in terms of quality.

Variability is the degree to which a distribution of, in this case, product measures is stretched or squeezed. Variability among products that should be equal is impossible to completely remove from a production system. Two products are never exactly the same in any aspect. **Nevertheless, companies attempt to minimize the variability in their products since the decrease in it leads to an increase in quality.** It can only be described in statistical terms [33].

There are two broad categories of methods related to Quality Control. These methods can be included in one of the following:

- Acceptance Sampling
- Statistical Process Control

2.1.1 Acceptance Sampling

Acceptance Sampling consists of retrieving a sample from a batch of objects in order to verify the amount of the nonconforming ones. It is commonly used in the conformity verification of two situations. It can be used to examine raw material or components before the production process starts or the final product after it finishes manufacture. The ratio between objects with nonconformities and objects that meet the specifications is then compared to pre-established standards. The approval of the objects lot depends on whether the ratio is below the standards or not. The quality levels should define the sample size and the maximum number of items with nonconformities so that the batch can be accepted. The larger the sample and the smaller the quantity of nonconforming items, the stricter the Acceptance Sampling is [19].

Several problems related to this method have been found:

- First of all, this method verifies the objects after they have already concluded their production process. This means that the resources have already been used and, in case a batch is rejected, not much is possible to be done afterward, leading to financial costs;
- Acceptance Sampling also creates an environment where a certain amount of products with nonconformities is considered normal. This is not a good production practice since it almost encourages the operatives to have a mindset in which products with nonconformities is not completely bad, but that problems like these happen and may be ignored;
- Finally, when the evaluation of a whole batch is performed using only a sample, there is always the risk that the sample does not contemplate the whole population's characteristics. This being said, there is a possibility that the whole batch of products is discarded due to the fact that the sample contained most of the few objects with nonconformities or a possibility that the lot is approved because the sample contained very few or none of the way too many nonconformant products.

2.1.2 Statistical Process Control

Statistical Process Control, better known as SPC, is a robust combination of statistical methods used to stabilize a process by minimizing the variability among manufactured products. By doing so, SPC can increase production capacity without additional financial investment in equipment, workforce, or overhead. It differs from the Acceptance Sampling method because **the verification of product conformity is partially done during the production process, rather than at the beginning or the end.** This allows preventive actions to be taken before the product goes through all the manufacturing processes. Therefore, SPC is a method with a preventive nature. When applied, patterns and significant variations are identified by comparing the current process results with the required standards. Consequently, deviations can be eliminated before the product is manufactured beyond the quality limits. However, **SPC does not solve the problems existent in the process, but it does detect them so that a solution can then be applied** [33, 19].

In a simple overview of this method, we can identify several benefits of SPC implementation in a manufacturing process. The main ones are summarized here [11]:

- Processes can be kept in control by the feedback provided;
- As soon as a problem occurs, SPC is triggered;
- Allows the differentiation between two types of problems — the ones that can be corrected and the ones that are due to chance;
- Companies can reduce the products inspection procedures;
- Gives insight knowledge during the production.

2.1.2.1 Variability Causes

There are countless reasons why variability is present in every manufacturing process. Those causes can range from defects in the operating machines to the mood of the employees. There have already been identified some of the most common and influential causes [19]:

- The **raw material** used in a process is also the final product from another one, which means that it also contains variability as one of its characteristics. Thus, we can not expect products to be concluded without variability from a process that already begins with it;
- **Environmental factors**, such as temperature and humidity, can affect the whole procedure;
- Differences between **machines' settings and calibrations** can very easily lead to variations. Besides this, it is known in the industry that, after **periodic adjustments**, variability tends to increase;
- Although it can happen in any process, when machines and tools are dealing with harder materials such as steel, they are more prone to have faster **natural wear**;
- Finally, the human intervention also plays its part in variability. The **natural ability, learning curves, and operatives experience** can increase or decrease the variation between products. Besides this, when **shifts change**, the quality usually also changes, since different people are assuming the tasks.

Despite covering a lot of the cases in which variability increases, there are a lot more possible causes for the increase than these previously mentioned. SPC is a method that can identify patterns in the manufacturing process and divide the causes of variability in two categories — **assignable causes** and **chance causes**. The main differences between these causes can be observed in Table 2.1, and the way SPC detects them will be explained in the next Subsection.

Table 2.1: Main differences between SPC's 2 types of variability causes.

Cause Category	Chance/Common	Assignable/Special
Variation	Random	Non-random
Pattern	None	May have one
Nature	Inherent to the process	Possible to explain and control
Process adjustment	Increases variation	Decreases variation

2.1.2.2 Control Charts

As it was said in the beginning of this section, SPC is a combination of techniques that are used to achieve stability in the manufacturing process. The tools used by SPC are called ***"The Magnificent 7"*** [33]:

- Histogram or Steam-and-Leaf plot
- Check sheet
- Pareto chart
- Cause-and-Effect diagram
- Defect concentration diagram
- Scatter diagram
- **Control chart**

From all these tools, **control charts are SPC's main technique and stand out for their complexity and for their overall better performance** at stabilizing a process, which improves the quality [9]. In this chapter, the only tool that will be explored is the control charts. I recommend the reader to see Montgomery's Introduction to Statistical Quality Control [33] for a good comprehension of the other tools. Control charts allow to graphically display the average of a measured quality characteristic in samples by plotting it versus process time or process sample number. Thus, companies can use them to **monitor and surveil the process**. An example of a control chart is shown in Figure 2.1 and it contains:

- **Center Line (CL)** — Where the product characteristic being analyzed should fall;
- **Upper Control Line (UCL)** — The upper boundary up to where the product characteristic may fall;
- **Lower Control Line (LCL)** — The lower boundary up to where the product characteristic may fall;

A process is considered to be in a state of in-control if the points are plotted within the control lines. Although processes can operate in this state for a long time, there will be assignable causes

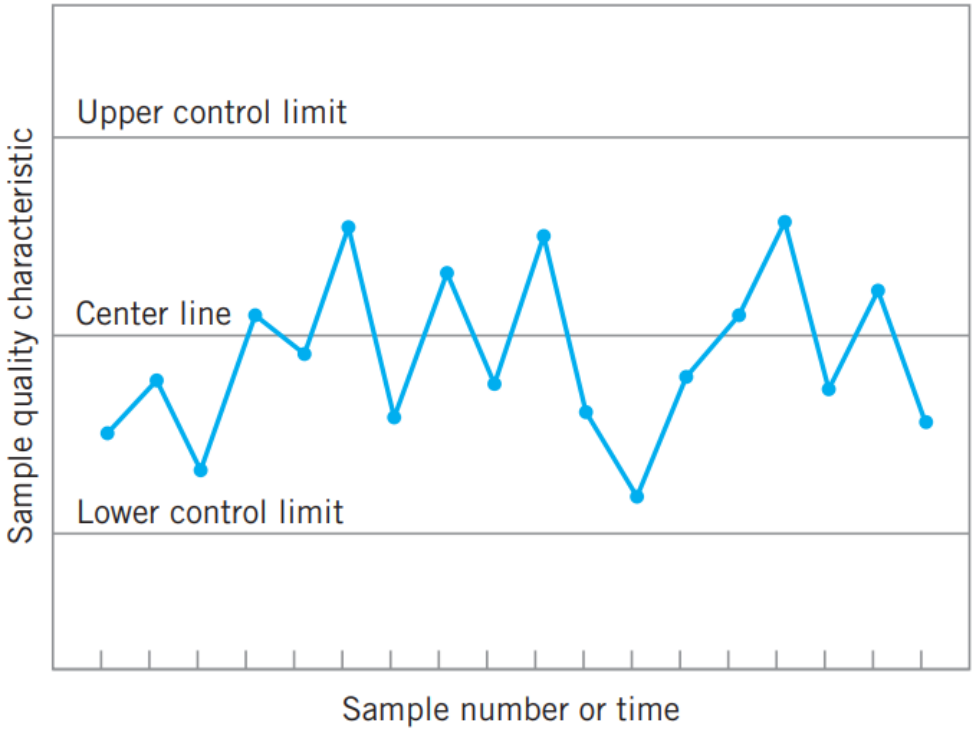

Figure 2.1: An example of a control chart presented by Montgomery [33].

that lead to losses of control sooner or later. The more out-of-control a process is, the more products will fall outside the limits. If the points start to plot outside the ranges, there is evidence that the process is in this state and should be corrected. Despite this, even if all the points are being plotted inside the limits, but a pattern can be identified, it does not necessarily mean that the process is controlled.

When analyzing control charts and the presence of patterns in those, it is important to know what **runs** are. A run occurs when some consecutive points are always ascending, *run up*, or descending, *run down*, in magnitude. Besides runs up and down, we can also identify a run when several points are all either above or below the CL. Any quite long run is an indicator that the process might be out-of-control. Therefore, we can say that control charts deal with the problem of pattern identification, regardless of the type of these [33].

Western Electric, an American company with connections to the industry sector, has proposed in 1956 some of the rules used in control charts to indicate nonrandom patterns. Later in 1984, Lloyd S. Nelson, based on Western Electic's rules, proposed some more. The rules are listed here:

1. **At least 1 point is plotted outside the control lines;**

2. At least 2 out of 3 consecutive points are plotted inside the last third of the control limits;

3. At least 4 out of 5 consecutive points are plotted inside the middle and last third of the control limits;

4. A run of at least 8 points on one of the sides of the CL;
5. A run up or run down of at least 6 points;
6. At least 15 consecutive points are plotted in the first third of the control limits;
7. At least 14 points in a row alternating between increasing and decreasing;
8. At least 8 consecutive points plotted outside both first thirds of the control limits;
9. Any other pattern that looks unusual;
10. Points plotted very close to the control limits.

After looking at these rules, it is understandable that there are quite a few that can be used to decide whether a process is in an out-of-control state. It is also important to point out that control charts can be classified into two different groups, depending on the quality characteristic's nature.

A **Variable Control Chart** is used when dealing with quality characteristics that are expressed by **numeric values**. In these cases, it is important to monitor the **mean value**, using an $\bar{x}$ **control chart**, and the **variability** of the quality characteristic, using either an s control chart (standard deviation) or an ***R* control chart** (range). The interpretation of one of the charts must always be accompanied by the other, and, additionally, the $\bar{x}$ chart should not be analyzed if the R chart reveals that the process is not in-control. When interpreting these, some common patterns appear:

- **Cyclic** — As the pattern name indicates, the points appear in a cycle. This is usually due to other cyclic changes that can range from environmental changes to periodic changes machine or workforce related;
- **Mixture** — Occur when the majority of plotted points fall close to the control limits;
- **Shift** — A shift in the mean value followed by a stabilization in it. It may be caused by new workers, processes, or tools;
- **Trend** — A steady progress of the mean value in one direction explained by, for example, degradation on the condition of the tools;
- **Stratification** — When points tend to gather around the CL. Many times this is due to errors in the control limits values.

When interpreting variable control charts, it must always be taken into account if there is any correlation between $\bar{x}$ and R. If this happens, according to Montgomery, *"the underlying distribution is skewed"*, meaning that there may be an error in the data analysis.

An **Attribute Control Chart** is used when dealing with quality characteristics that can not be expressed numerically and are expressed as **attributes**, such as *product with nonconformities* or *product without nonconformities*. This type of chart does not provide as much information as the previous one since numeric values are able to represent things that attributes can not. There are three different types of charts used to represent attributes:

- **Control chart for Fraction Nonconforming** — Represents the proportion between nonconforming products and the total amount of products in a population. It is also called ***p* chart**;
- **Control chart for Fraction Nonconforming** — Can be named as ***c* chart** and shows the number of nonconforming products instead of their ratio;
- **Control chart for Nonconforming per Unit** — Employed when it is more convenient to study the average number of nonconforming products per unit manufactured. It is known as ***u* chart**.

2.1.2.3 OCAP

We have already seen in this section that SPC is a method with a preventive nature. This implies that, by using SPC's capabilities, companies are able to detect the existing problems in their production processes. However, the problems are not solved by this method. **To eradicate the root causes of the problems, OCAP can be used**. Out of Control Action Plan or OCAP explains the chain of actions that should be performed when one problem and its cause are detected. This plan can either be a detailed text description or a schema for an easier interpretation and to take corrective actions faster [33]. An OCAP consists in:

- **Checkpoints** — The different possible problem causes;
- **Terminators** — The actions that, when taken, will solve the associated problem.

This type of plan can and should be adjusted and improved as the knowledge about the process grows. **Control charts should be followed by an OCAP so that preventive and corrective actions can be conducted.** An example of an OCAP is shown in Figure 2.2.

2.2 Machining Operations

Machining is the controlled process in which cutting tools are used to mechanically cut material in order to obtain the chosen geometrical parts. This procedure is usually related to metal shaping. Due to the required precision, accuracy, and efficiency in the production of a vast diversity of geometries, it is considered by many as one of the most adaptable and skilled manufacturing processes [15].

The **machine tool** or **cutting tool** is a powerful instrument used to manufacture components by removing material from the workpart. Machine tools are usually composed of four elements that are essential in the manufacturing process [34]:

- An energy source;
- A part that ensures the process is carried out with safety;

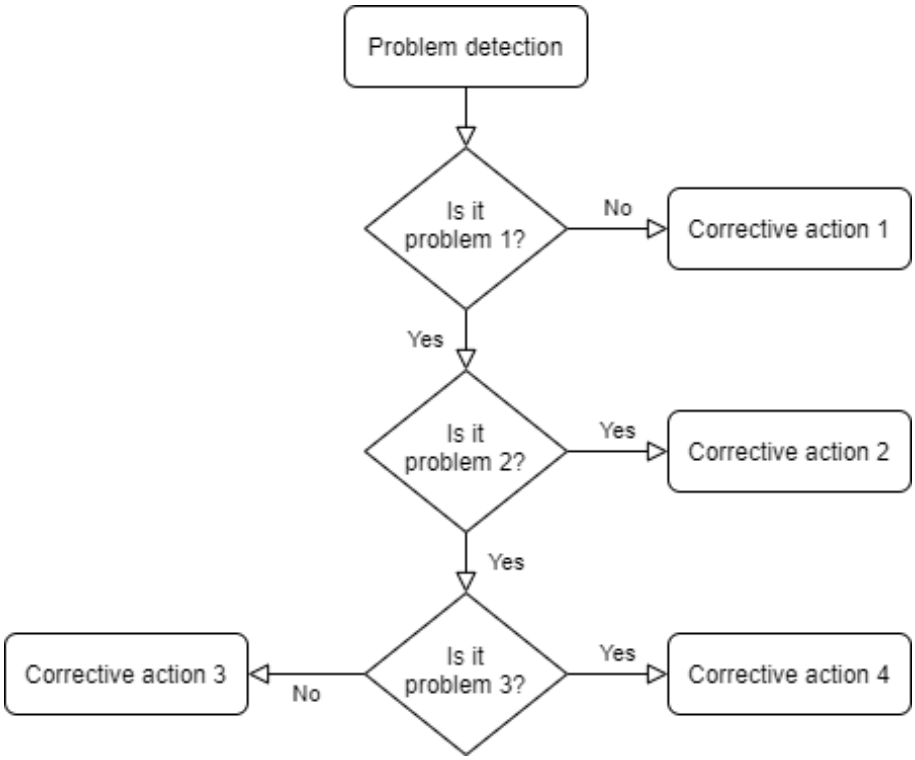

Figure 2.2: A generic OCAP.

- A segment that orients the machine and makes sure it is secured;
- A component that supervises the previous elements.

Machining operations usually involve **generating** and **forming** the necessary geometries. Whereas in the first one the new geometry of the workpart — the material to be machined — is shaped by the trajectory of the cutting tool, in the second one it is controlled by the shape of the instrument. Several operations can be performed using these two factors:

- **Turning**
- Drilling
- **Milling**
- Shaping
- Planing
- Broaching
- Sawing

In this dissertation, there is a particular interest in the turning and milling operations, since those are the ones that are present in the later provided database. Based on this, only those two will be described in more detail.

2.2.1 Turning

Turning is one of the most common operations. In this process, the workpart is cut while rotating, creating axially symmetrical shapes with respect to the rotation axis. The produced parts may present characteristics such as different diameter steps, holes, and grooves.

2.2.1.1 Turning Operations

There are various variations in the turning process itself. The most usual variations are [27, 28]:

- **Face Turning** — Also known as **Facing**, it is a turning operation in which a cut perpendicular to its rotation axis reduces the length of the workpart. It produces a smooth and flat surface;
- **Outer Diameter Turning** — It is an external operation that occurs on the outer surface of the workpart, machining the outside diameter of it. Due to the fact that it is the most commonly used type of turning, it is also the one that produces its output with the highest quality;
- **Inner Diameter Turning** — Also known as **Boring**, it is an internal turning operation that machines the profile on the inside diameter of the workpart. Although it can only be used to enlarge an already existing hole, it produces precise holes;
- **Grooving** — In this process, the diameter of the workpart is reduced over a narrow surface, which means that the machining occurs between two edges of the piece. This process can be used both in the inside and outside of the workpart;
- **Threading** — This is the operation applied when the objective is creating threads in a workpart. These are generated by feeding a pointed tool across the surface of a rotating workpart. If the threads are made on the outside surface, it is called an external threading, otherwise it is called an internal threading.

2.2.1.2 Turning Tool

In the modern industry, there are various cutting tools, each designed specifically for fulfilling a particular job. In order to obtain knowledge from the process, there is the need to get familiar with different tools and the way these are applied to everyday turning operations.

In the case of turning processes, the tool used is a *lathe* (Figure 2.3). It shapes the workpart by rotating it while a tool (sharp single cutting point) is pressed against the workpiece. Several parts form lathes [42]:

- **Headstock** — A fixed support that holds the machine so that the vibrations caused by the rotation do not affect the manufacture;
- **Tailstock** — Piece that supports the rotational axis of the workpiece. An operator can move the tailstock so that the workpart can fit in the machine;

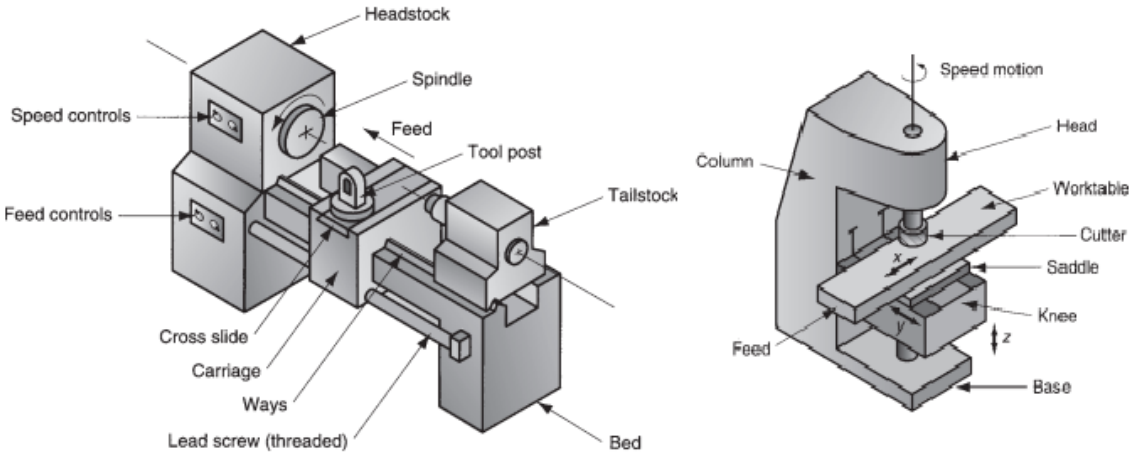

Figure 2.3: A lathe (left) and a milling machine (right) diagrams retrieved from Mikell Groover's Fundamentals of Modern Manufacturing book [15].

- **Bed** — As the name suggests, this element is a robust base in which headstock, tailstock, and other components are held. It absorbs all the vibrations caused by the cutting and is particularly important when the workpiece is not balanced;
- **Carriage** — Its function is to provide cross and longitudinal movement so that the sharp cutting tool can be placed in the right position along the bed;
- **Feed Mechanism** — This part is the one that allows the operator to control and adjust the feed rate through the spindle speed. The feed is the distance that the tool proceeds into the workpiece in each rotation.

2.2.2 Milling

Milling is an operation in which the workpart goes through a rotating tool that removes the unnecessary material. It is an interrupted operation since the cutting points are not always touching the workpart. Generally, milling is used to obtain planar surfaces and not axially symmetric, but it can be used to obtain other types of surfaces.

2.2.2.1 Milling Operations

There are various variations in the milling process itself. The most usual variations are [26, 25]:

- **Side Milling** — Also known as **Contouring**, it is an operation applied to the outside edges of the workpart. Side milling is used when the objective is the production of a flat surface. The machine can control the depth of the cut;
- **Face Milling** — Face milling is similar to side milling since both aim at obtaining a flat surface. However, the first is applied to the edges of the workpart, whereas the second is

performed so that it cuts the top face of the part. The tool is placed perpendicular to the workpart allowing it to cut the material;

- **Slot Milling** — This operation is used when the operator intends to make a slot or a groove in the workpiece. Besides this, the machining is performed between two edge surfaces. Since slots can be deep or shallow, wide or narrow, closed or open, the tool machining the piece must be conforming to the slot. Using slot milling with enough depth in the workpiece allows to cut it in two;

- **Plunge Milling** — While usually the tool cuts with its periphery, the cutting occurs at the end of it in this operation. This modification changes the direction of the cutting forces from radial to axial, which is beneficial to the process. It is a robust process when dealing with complex shapes or deep and closed slots;

- **Ramping** — This operation is the most complex one since it requires the cutting tool to move along the X, Y, and Z-axis simultaneously. This technique can be used to make angular paths such as pockets, cavities, and engravings. N

2.2.2.2 Milling Tool

Just like in turning operations, there is also a specific tool that can be used to deal with milling processes. The tool used is a *milling machine* (Figure 2.3). Milling machines are versatile tools used to remove metal from a workpiece. They do so by using a milling cutter, a rotating cutting tool, that moves along the fed piece. This movement allows the tool to obtain the desired workpiece shape on flat, rough, and irregular surfaces.

There are some variations on this machine so that it can successfully perform the various existing milling operations:

- **Horizontal Milling Machine**
- **Vertical Milling Machine**
- Knee-Type Milling Machine
- Ram-Type Milling Machine
- Bed-Type Milling Machine
- Planer-Type Milling Machine

Regardless of the variation, all the milling machines have in common the parts that compose them [30, 31]:

- **Base and Column** — These two parts are, together, the milling machine's foundation since they are able to absorb the vibrations caused by the milling and keep it stiff. The rest of the parts are attached to them. It is also in these parts that the oil and coolant are stored;

- **Knee** — The knee is a moving part that moves vertically, either mechanically or hydraulically. By sliding it up and down, the operator can bring the tool and the workpart closer or farther. Its function is to, besides holding all the gearing mechanism, support both the saddle and table;
- **Saddle** — This part is also a moving one. Located between the knee and table, it supports the last one and can move transversely to fulfill its primary function of granting motion to the workpiece. Just like the knee, by sliding it in and out, the operator is able to adjust the position of the workpart in relation to the tool;
- **Table** — The table is a rectangular casting part that is located above the knee and saddle. Its main purpose is to hold the piece tightly while it is machined and, in order to do this, it contains several T-slots. By moving both the knee and the saddle, the operator can place the table and, with it, the workpiece, wherever he needs;
- **Spindle** — A rotating part that drives the tool. It contains a slot at its front end where the cutting tool can be placed. Depending on the type of milling machine, it can be in a horizontal or vertical position.

2.2.3 Machining Centers

Machining centers are a computerized machine tool that is able to perform multiple operations almost without human interaction. As opposed to the general procedure in which a worker needs to swap tools, machining centers are capable of doing it themselves, reducing effort and time in the production line. These machines can work with at least 3 and up to 5 different axes [15].

2.3 Ishikawa Diagram

Created in the 1940s by a Japanese Professor and improved until its popularization in the 1960s, the Ishikawa Diagram, also known as **fishbone diagram**, is a simple and yet effective cause-and-effect diagram used in quality control. An example of a fishbone diagram is shown in Figure 2.4. In the manufacturing sector, this diagram helps to understand the causes that lead to reductions in product quality by relating all the possible causes to the problem. In it, the causes are divided and grouped according to the field where they fall [48, 24].

When this type of diagram is applied to situations regarding the manufacturing of products, the causes identified are usually the ones called *Ishikawa's 6 Ms*. These causes can be described as follows [18]:

- **Machine** — Failures and wrong calibration in the machines that are being operated can very easily result in failures in production. A frequent maintenance is able to solve many of machine related problems;
- **Method** — The method applied in the process of manufacture;

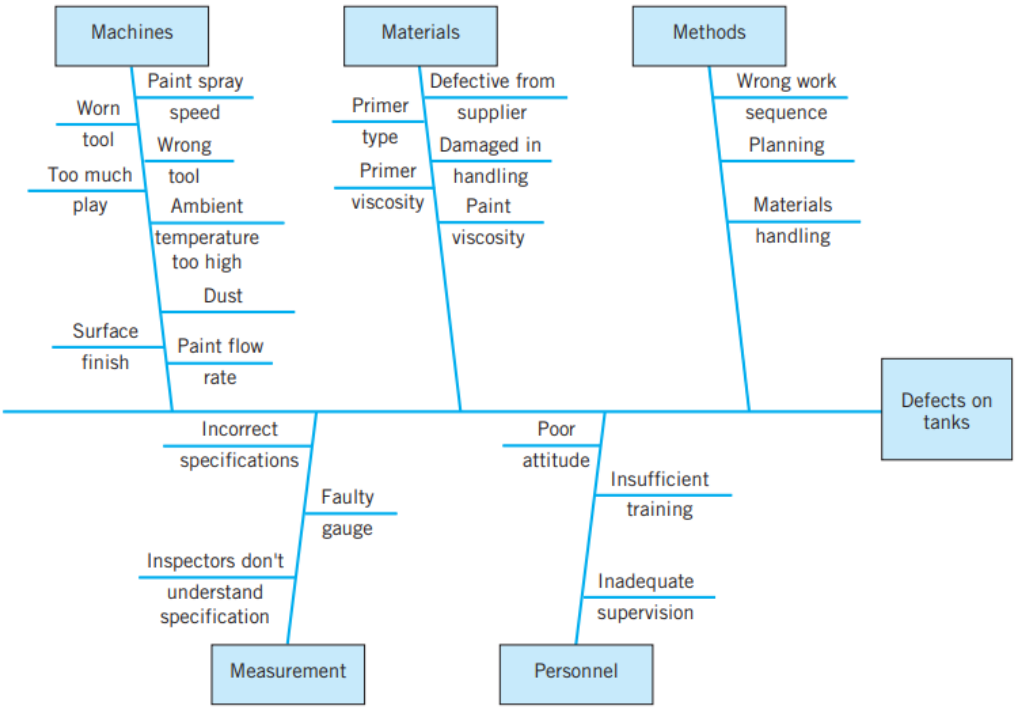

Figure 2.4: An example of an Ishikawa diagram retrieved from Montgomery's Statistical Quality Control [33].

- **Material** — Raw material issues can lead to issues with the final product;
- **Workforce** — Labor with insufficient skill to work in a certain process can cause nonconformities;
- **Measurement** — This cause is related to the inspection part of the process. The flaws on the process associated with this cause are usually either incorrect measurements or incorrect product specifications;
- **Mother Nature** — The environment is able to influence the whole production process through changes in some measures such as temperature, pressure, and humidity.

This diagram was found to be relevant in order to conduct data collection and to filter features with the purpose of obtaining relevant data.

2.4 Summary

This chapter is divided into three sections. Section 2.1 introduces the two quality control methods used nowadays in the industry. Initially, Acceptance Sampling is explained, and its issues are explored. The SPC subsection defines SPC, resumes the two different variability causes, and explains in detail control charts and their variations. The Out of Control Action Plan is also discussed.

Section 2.2 contains a brief analysis of machining operations, with a particular attention to the operations encompassed by this work, turning (Subsection 2.2.1) and milling (Subsection 2.2.2).

Section 2.3 describes the Ishikawa diagram that assists in the identification of possible causes for poor manufacturing performances.

Chapter 3

Data Mining and Machine Learning

Data Mining and Machine Learning are both subfields of Data Science. The two concepts are similar since they are applied to solve complex problems making use of data. Using both, a developer is able to extract relevant information from large datasets. This information is usually retrieved in the form of a pattern that allows humans to acquire knowledge that, otherwise, would not be possible to obtain due to the data's dimension. By making use of this information, the developer can apply an algorithm that generates a model capable of making either predictions or decisions based on the provided data.

3.1 Data Analytics

Nowadays, although technology has improved, data analysts deal with increasingly large amounts of data and more complex databases. This means that despite the current computers are able to retrieve data from those databases reasonably easily, the task of analyzing it has become more complicated [45]. This increase in complexity makes it necessary that are experts handling the data. Besides this, due to the modern manufacturing market competitiveness, companies need to have implemented a system that can systematically help it take decisions. The decisions can be related to several aspects of the company, such as its manufacturing process [22]. The system referred is therefore a system that implements data analytics.

Data analytics is the process in which a company makes use of its extensive data resources and analyzes them. It is an interdisciplinary field that puts together powerful tools such as statistics, data mining, machine learning, artificial intelligence and others. The correct application of data analytics allows a company to support its decisions. According to Thomas A. Runkler, there are four phases in a data analysis project that are shown in Figure 3.1.

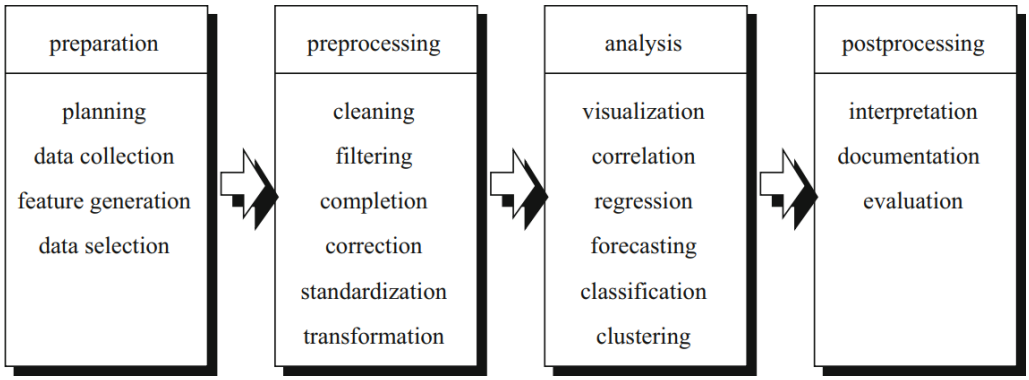

Figure 3.1: Data analysis phases retrieved from Thomas A. Runkle [40].

3.2 CRISP-DM Methodology

Cross-Industry Standard Process for Data Mining or CRISP-DM is a methodology that aims at improving the overall process of retrieving information from datasets. CRISP-DM is a properly structured and yet flexible process that determines that, in order to successfully datamine, one should follow **a cycle of six stages**. These stages are shown in Figure 3.2 and are also described next [35, 47]:

1. **Business Understanding** — In this stage, the project's objectives and requirements are defined so that the problem can be formulated as a DM problem. The work plan is also determined;

2. **Data Understanding** — Initially, the data is collected and analyzed. As the familiarization with data increases, the existing issues related to it are identified. Data Understanding and Business Understanding phases are interlinked since a good problem formulation must take into account the characteristics of the data;

3. **Data Preparation** — This phase encompasses several actions that, after being performed, result in the final dataset. Here, the relevant variables are selected, missing and erroneous data is determined, new derived features can be added, the formats are defined, and data from various sources can be integrated;

4. **Modeling** — This is the phase in which the data mining tools are run. After a careful selection of the modeling techniques, these are selected and can be applied. Using their results, the parameters can be recalculated until the best values are discovered. Once again, there is an interlink between this phase and the previous one;

5. **Evaluation** — As the name implies, the evaluation of the model's results is performed in this stage. By reviewing them, it is possible to understand whether the business objectives have been met or not. This stage looks both at the past and the future of the process. The previous actions are assessed, and the next ones are determined;

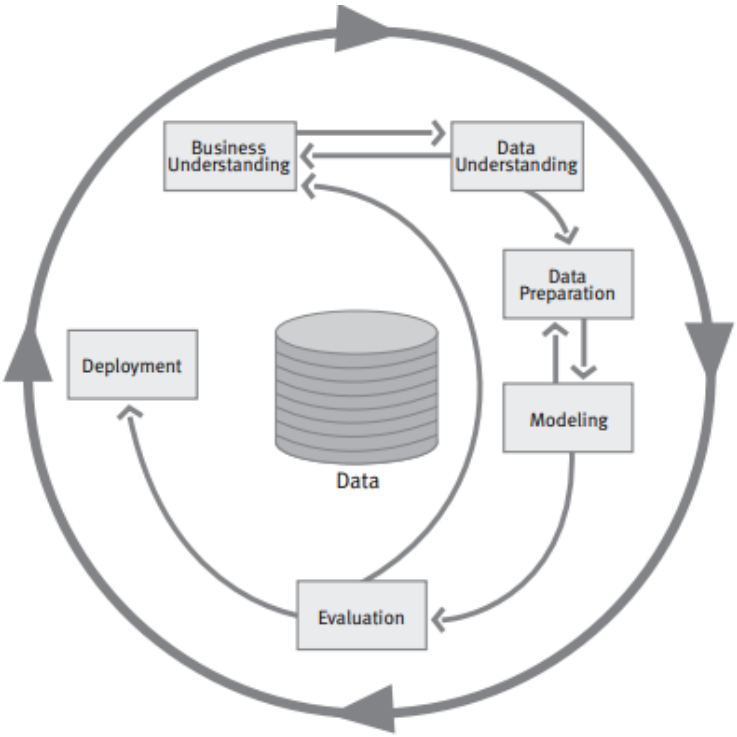

Figure 3.2: The CRISP-DM phases retrieved from Chapman et al [8].

6. **Deployment** — In this final phase of the DM process, the final model or models are put into practice. The knowledge obtained by these is organized so that it can be presented to the client.

Other methodologies can be used in DM projects, such as SEMMA and KDD, but CRISP-DM is usually the preferred method in organizations worldwide. The reason for this choice is related to its completeness [2].

3.3 Feature Selection

Feature selection, also known as variable selection, is the process in which the number of variables handled by a predictive model is reduced. This selection of the **subset of features** that will be used in the ML problem is essential since removing certain variables allows to reduce the computational cost and training time of a model. Besides these, it also decreases the number of irrelevant, possibly misleading, attributes.

During the feature selection process, the correlation between each input and output variables is computed. A subset of variables is then defined, and only those with strong correlations are included in it. **The higher the correlation, the more an input variable affects the final value of the output one** [23].

There are several methods to perform the feature selection but, taking into account the scope of this work, only the **supervised methods** are eligible. In supervised learning, the model is initially trained knowing the value of both input and output variables [29]. The supervised methods can be divided into three types of methods:

- **Wrapper** — These methods base the subset's choice with the most relevant attributes in the creation of a large number of models in which different subsets of input variables are tested, and the ones with higher accuracy are then selected. The main disadvantage of these techniques is their computational cost due to the way the search is performed;
- **Filter** — The filter methods are statistical methods in which the feature selection is independent of the model. The correlation between the input and the target variables is computed and, taking those scores into account, the attributes are then selected. Although these processes do not have high computational costs and are powerful tools against overfitting, they are prone to select redundant variables since the correlation between input attributes is not considered;
- **Intrinsic** — Also known as Embedded, these methods are characterized by being the ones that are built-in in the model. Thereby, the feature selection process is executed during the training phase.

Despite its high computational cost, the wrapper methods are usually the preferred ones when that is not a problem because they are not model-independent [5]. The primary method among the ones included in the wrapper ones is **Recursive Feature Elimination**, more commonly known as RFE.

3.3.1 Recursive Feature Elimination

As it was mentioned earlier, RFE is a wrapper method. Initially, this technique creates a model that uses the complete set of variables. These variables are then ranked by importance using a relevance score. The attribute with the lowest score is removed, the model fully rebuilt, and the scores computed once again. As this iterative process proceeds, the subset of variables being used by the model becomes more and more restricted, and only high-scoring features are taken into consideration. This method also requires that a hyperparameter representing the final number of variables included in the subset is defined. When the process ends, the resulting subset is considered as the optimal one, and the one that should be used to train the predictive model [50].

Several studies on RFE have been performed, and, in these, it is used alongside different algorithms. Darst et al., 2017, have shown that, when dealing with high-dimensional data and numerous highly correlated features, RFE in association with Random Forests tend to not be appropriate [7]. In the work of Granitto et al., 2006, a comparison between the use of RFE with Support Vector Machine (SVM-RFE) and RFE with Random Forests (RF-RFE) was made. This

comparison took as evaluation metrics the algorithm's capacity to find subsets with high capability of discrimination and its capacity in finding the minimum possible selection error. RF-RFE outperformed SVM-RFE [14].

3.4 Supervised Binary Classification Algorithms

ML problems have many possible approaches but these can be classified and divided by its available feedback into three categories [32]:

- **Supervised Learning** — The model receives from a teacher a training dataset with examples of inputs and the expected result. Using those examples the model attempts to learn the function that maps the inputs to the outputs;
- **Reinforcement Learning** — In this type of learning the model receives less feedback than Supervised Learning. This one is related to models that need to perform certain actions and their objective is the maximization of the rewards;
- **Unsupervised Learning** — There is no teacher and, therefore, there is no possible evaluation that the model can receive.

Taking into account the characteristics of this work, it will be included in the **Supervised Learning approach**. The methods included in this category can, once again, be classified and divided into two types:

- **Regression** — This type of algorithms focuses on problems in which the dependent variable is a quantitative value;
- **Classification** — Algorithms of this type focus on problems in a discrete dependent variable is supposed to be grouped into a class.

Once again, if the characteristics of the work that is to be developed in this dissertation are considered, we understand that it is included in the Classification type. More specifically, this is a **Binary Classification** problem. Binary Classification is the subfield that refers to the prediction of one between two classes. When dealing with this type of problems, there are several algorithms that can be used [6]:

- **Logistic Regression**
- **k-Nearest Neighbors**
- **Decision Trees**
- **Support Vector Machine**
- **Naive Bayes**

- **Artificial Neural Networks**

Each algorithm will now be described in its own subsection. Besides its description, there is also an overview of some works on Machine Learning where these algorithms were applied so that a better comprehension of their use can be obtained.

3.4.1 Logistic Regression

Logistic regression algorithms are statistical algorithms that fit a logistic function into the dataset when applied to classification problem. This algorithm in specific is commonly used in binary classification because, in those cases, the output variable is dichotomous, meaning that the two possible output values are mutually exclusive [43]. Using logistic regression, the model is able to output the probability of the dependent variable belonging to a particular class. The fact that the model returns a probability also means that the limits between which each class is contained can easily be modified if needed. Besides this, these models can, without difficulties, update themselves when new data is inserted since they infer neither features' distributions nor relationships between dependent and independent variables [36]. However, for this method to be successful, the features with strong correlations should be removed during the feature selection process, and large samples should be supplied to the model [41].

Stephan Dreiseitl and Lucila Ohno-Machado, 2002, performed a review on some models applied to classification tasks in the medical field. They claim that this algorithm is relatively popular due to the ease of developing the model in comparison to other algorithms. Besides this, the hyperparameters required can easily be interpreted and defined [12].

In another work, Abdulhamit Subasi and Ergun Erçelebi, 2005, examined and used classification techniques to analyze the brain's electrical activity. According to them, the major issue when developing a logistic regression model is that, although the hyperparameters can be clearly understood, there is no previously acquired knowledge regarding the combination of those that produce the optimal model. Finally, as stated before, the authors also mention that this algorithm struggles when dealing with high-dimensional data and outliers, just as the majority of methods that rely on statistics [44].

3.4.2 k-Nearest Neighbors

k-Nearest Neighbor, more known as kNN, is a non-parametric algorithm. Its common use is in cases in which the model searches for the most related class [43]. It is also considered a lazy learning algorithm and, despite its ease of implementation, it is not very efficient and largely depends on the value assigned to k. The principle behind this method is the assignment of a class to an yet unlabeled point taking into consideration the k nearest points and their own class. Bearing this in mind, it is understandable that this algorithm is particularly susceptible to noise and outliers [4]. Finally, kNN's performance is very affected by data size [17].

When looking at previous works where kNN is applied to data with high dimensions, one can understand that they focus on reducing the search time without affecting the accuracy of the model.

For example, Zhenyun Deng et al., 2016, proposes a kNN model in which the data is clustered into several m parts. Afterward, the classification is performed normally with the difference of each yet unlabeled point being clustered into an already existing cluster (part) and not to other already classified points. According to the authors, their model was more efficient without having a significant reduction in accuracy [10].

Adeniyi et al., 2014, apply the kNN classification method to a recommendation system on a website. This system considers the sequence of links that the website's visitor follows, *click path*, and suggests other links. The authors chose the kNN algorithm due to the fact that this is a real-time situation in which the quickness of the classification is relevant. They note that the quality of the recommendation is around 70%, which is acceptable in this case. However, they also mention that other classification techniques could easily outperform kNN if the computational time were not such a relevant factor [1].

3.4.3 Decision Trees

A decision tree is an algorithm that is used to explicitly represent decisions and the way these are forked taking into account some input variables in a tree structure. By walking through the created decision tree the model can go through decision nodes, the ones with input variables and that are further expanded, and leaf nodes, the final ones where the value of the output variable is obtained [43].

Decision trees are frequently used due to the plenty of advantages they contain. Models applying this algorithm are almost not affected by outliers, strong correlated data, missing values, and redundant features. They can also deal with several types of data. Nonetheless, DT has some difficulties when facing datasets with many dimensions. Since small errors propagate through the tree, as the number of dimensions increases, the propagation of errors becomes a bigger problem [49]. Tree prunning is usually necessary in those cases or overfitting will highly likely occur [4].

There are quite a few variations of this algorithm such as ID3, C4.5, C5.0, and CART.

Joanna Ronowicz et al., 2015, studied the cause-effect relationships in composition differences in pellet production. Decision trees were used to discover these hidden correlations in the large datasets. According to the authors, this technique's use was successful since it increased their knowledge of the process. Due to the fact that decision trees are not black box models, it is possible to obtain the rules that guided the algorithm while it was solving the classification problem. These rules can now be used to support rational decision-making in the pellet manufacturing process [39].

In another work focusing on decision trees, Wenjing Zhao et al., 2019, proposed a classification method for power quality disturbance. In order to be able to compare, besides applying decision trees, the authors also use Support Vector Machines (SVM) and kNN. As stated in their results, decision trees outperformed the other two methods in accuracy and time to perform the classification. However, the authors also refer that, if the feature selection had been specifically performed to attend SVM's or kNN's needs, the accuracy results could have differed [51].

3.4.4 Support Vector Machine

Support Vector Machine or SVM is a complex, but highly accurate, algorithm that is applied in both regression and classification problems. When dealing with this method it is important to understand the concept of hyperplanes. An hyperplane is how the input variables are split. SVM attempts to maximize the minimum distance between a hyperplane and its nearest sample point [43]. The larger the distance, the more optimal the hyperplane and the less the expected generalization error [4].

Viviana Fernandez, 2006, performs a comparison between Autoregressive Integrated Moving Average (ARIMA), Unobserved Components, Wavelet-based and SVM. This work focuses in the accuracy of the prediction in U.S. metal shipments data and in the manufacturing industry. Fernandez notes that SVM has a higher forecast accuracy than the other prediction methods. Moreover, the comparison showed that SVM is also the most robust method, since it was the one that better performed in more circumstances. Finally, it was verified that SVM is a method that can easily be used to predict time series [13].

3.4.5 Naive Bayes

Naive Bayes or NB is a simple to implement algorithm that is based in the application of the Bayes' theorem. This method simplifies the learning process by making the assumption that attributes are independent from each other, which invokes the Class Conditional Independence theory [17, 37]. By making this assumption, NB requires less time in training. Just like logistic regression methods, NB models deal with probabilities. It is also of interest to refer that this algorithm does not require any hyperparameters, but that it does not deal in an effective way with high-dimensional data [49].

In Mehdi Salehi's dissertation, 2018, the Bayes inference was applied as the predictive modeling approach in several different methods. These models were created to predict machining performance metrics. NB algorithm is considered to perform well in this environment where there are relatively high uncertainties in the manufacturing process. Furthermore, Salehi considers NB the right algorithm to be used due to its ability to incorporate prior knowledge. This allows the measures performed during the machining process to be updated [37].

In another work, I. Rish, 2001, proposes a study in which NB's performance is analyzed in several different scenarios. Rish shows that NB is usually successful in the prediction task, even when the probabilities are not accurately calculated. The author also demonstrates NB's good performance when dealing with independent features and functionally dependent ones [38].

3.4.6 Artificial Neural Networks

The final algorithm that is reviewed in this section is the Artificial Neural Network or ANN. This network is based on the neural network that constitutes the brain and is composed by simple, but adaptable, input and output units, called neurons, connected to each other. Each connection has a certain weight associated that changes throughout the learning process [4]. An ANN can be

defined by its network topology, the characteristics of its neurons, and, finally, by its training and learning rules.

Artificial Neural Networks are applicable to many real-life situations where both input and output values are continuous, and the relationships between them can be not only non-linear, but also fairly dynamic. Although the learning process in ANN is accelerated by its parallel strategy, it still involves a quite long learning time. Lastly, this algorithm has the disadvantage of representing the knowledge acquired in the form of a network, which makes it harder to interpret, and the fact that it requires tuning of certain parameters such as the number of layers, number of neurons in each layer and also the network structure [43, 32].

Javed Khan et al., 2002, developed a classification model using ANN. This model is used to diagnose human cancers and label them according to their gene expression signatures. PCA, Principal Component Analysis, was used to reduce the total number of evaluated features. The results show that the generated model was able to correctly classify all of the 63 different types of cancer and did not reveal signs of overfitting. However, the authors pointed out that this ANN-based model is not appropriate to discover the cause-effect relationships that lead to cancer [21].

3.5 Summary

This chapter is divided into four sections. Section 3.1 introduces the topic of Data Analytics, explaining what it is, why this field was created, and which stages its process comprises.

Section 3.2 contains the explanation of the CRISP-DM methodology and a detailed description of each of the phases that incorporate it.

Section 3.3 describes the procedure of choosing the subset of features that are more relevant to the project domain. It also describes the different types of methods with a particular emphasis on RFE.

Section 3.4 narrows the initial problem to a supervised binary classification problem and contains a comprehensive explanation of the logistic regression, kNN, decision trees, SVM, Naive Bayes, and ANN classifiers.

Chapter 4

Experimental Setup

The experimental setup is the stage of the project in which the development occurs. In this dissertation, the experimental setup consists of a sequence of steps in which each step deals with a particular problem by making use of some of the methods and techniques described in Chapter 3. The initial phases are related to data collection, preprocessing, and analysis. Afterward, some procedures of new feature creation are described. Finally, the training of models and the operations that precede it to prepare the dataset are explained.

As the previously mentioned phases can indicate, a CRISP-DM methodology was applied in this project. However, the processes and methods described in this chapter are already the final ones, despite not all of them being present since the beginning.

4.1 Dataset

Regardless of the machine learning project and its subject, it is always of great importance that the data is analyzed and studied. The interpretation of the dataset allows the Data Mining and Machine Learning experts to drive effective decision-making in what concerns the resolution of the data problems and how to explore it in a meaningful way. If, on the one hand, data itself is just numbers and words, on the other hand, well presented and structured data becomes relevant information.

4.1.1 Database Description

The dataset used in the prediction task results from several joins between five of the seven different tables retrieved from the provided database. The data used in this work had already been directly fetched from the database and was already delivered inside *csv* files. The diagram that represents it is shown in Figure 4.1, and the tables it comprises are explained more in detail in the following subsections.

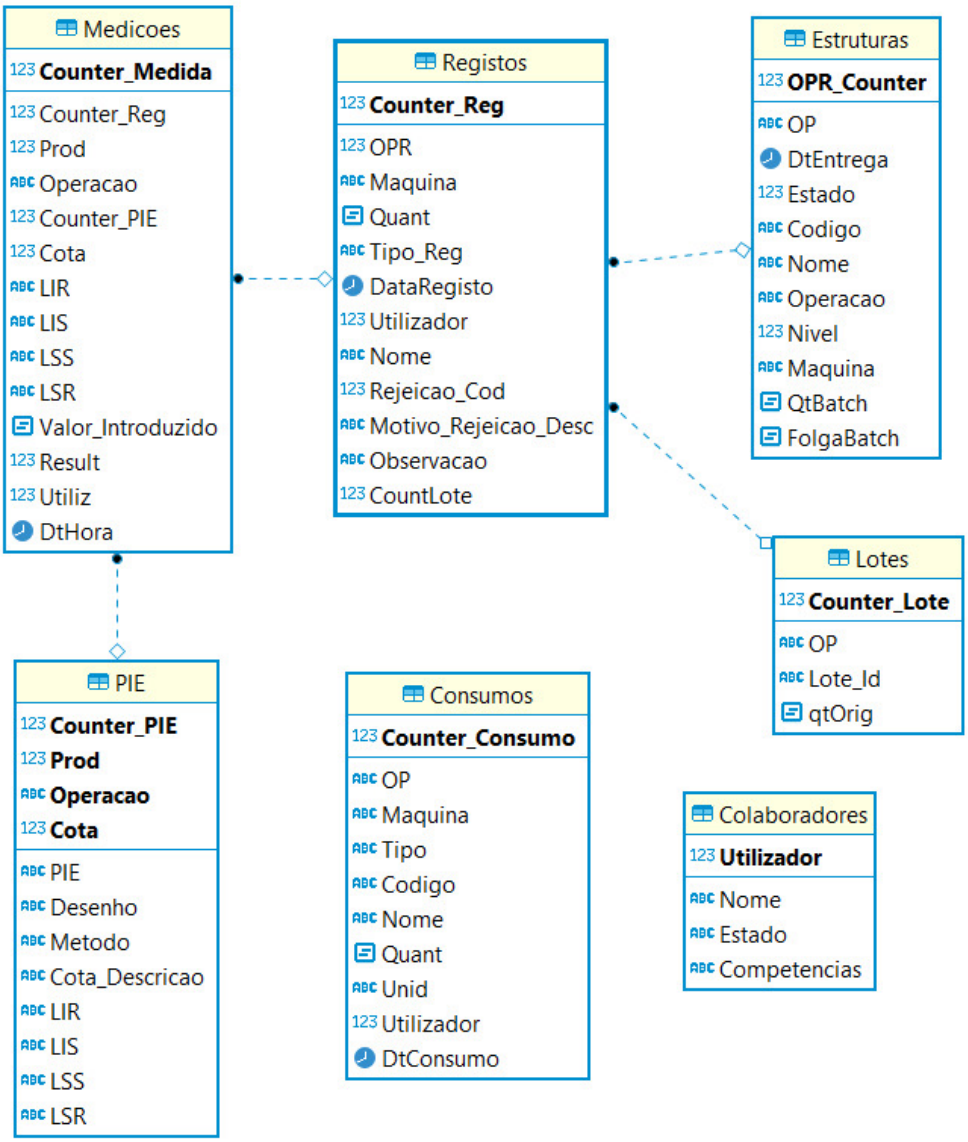

Figure 4.1: UML for the provided database.

4.1.1.1 *Medicoes* or Measurements

This table contains the information related to the measurements performed in the manufactured products, and each item represents a single measurement. The attributes of this table can be described as follows:

- **Counter_Medida** — Primary key for *Medicoes* table;
- **Counter_Reg** — Foreign key for *Registos* table;
- **Prod** — Foreign key for *PIE* table;

- **Operacao** — Foreign key for *PIE* table;
- **Counter_PIE** — Foreign key for *PIE* table;
- **Cota** — Foreign key for *PIE* table;
- **LIR** — Lower Rejection Limit;
- **LIS** — Lower Safety Limit;
- **LSS** — Upper Safety Limit;
- **LSR** — Upper Rejection Limit;
- **Valor_Introduzido** — Measurement value;
- **Result** — An attribute that can be either *0*, *1*, or *2*. These values stand respectively for an accepted product, an accepted one but outside the safety limits, or a rejected one. This is the target variable in the binary classification problem. In a later stage, values equal to *2* will be replaced by *1* values as both represent a nonconformity;
- **Utiliz** — Identifier of the worker in charge of the measurement;
- **DtHora** — Date and time of the measurement.

4.1.1.2 *Registos* or Records

This table contains information related to the production orders (PO). Each item represents a PO, which is the directive that describes the products that must be manufactured and their respective quantities. The attributes of this table can be described as follows:

- **Counter_Reg** — Primary key for *Registos* table;
- **OPR** — Foreign key for *Estruturas* table;
- **Maquina** — Identifier of the machine where the product is manufactured;
- **Quant** — Amount of products manufactured;
- **Tipo_Reg** — Value that indicates whether the products were accepted or not during the measurements;
- **DataRegisto** — Date and time of the products' manufacture;
- **Utilizador** — Identifier of the worker in charge of the manufacture;
- **Nome** — Name of the worker in charge of the manufacture;
- **Rejeicao_Cod** — Identifier of the rejection reason;

- **Motivo_Rejeicao_Desc** — Description of the rejection reason;
- **Observacao** — Any particular note on the products;
- **CountLote** — Foreign key for *Lotes* table.

4.1.1.3 *Estruturas* or Structures

This table contains information related to the structure of the production orders, and each item represents the structure of a single PO. The attributes of this table can be described as follows:

- **OP** — PO code;
- **DtEntrega** — Foreseen delivery date of the products;
- **Estado** — State of the PO;
- **Codigo** — Reference code;
- **Nome** — Reference name;
- **OPR_Counter** — Primary key for *Estruturas* table;
- **Operacao** — Performed operation;
- **Nivel** — Sequence number of the PO;
- **Maquina** — Machine in which the production is performed;
- **QtBatch** — Maximum quantity of control sampling;
- **FolgaBatch** — Tolerance for the *QtBatch* attribute.

4.1.1.4 *PIE*

This table contains information related to the inspection and testing plans (PIE), and each item represents a single PIE. The attributes of this table can be described as follows:

- **Counter_PIE** — One of the columns that constitute the primary key for *Estruturas* table;
- **PIE** — Name of the PIE;
- **Desenho** — Drawing of the product for control purposes;
- **Prod** — One of the columns that constitute the primary key for *Estruturas* table;
- **Operacao** — One of the columns that constitute the primary key for *Estruturas* table;
- **Metodo** — Measurement instrument or method;

- **Cota** — One of the columns that constitute the primary key for *Estruturas* table;
- **Cota_Descricao** — Description of the part of the piece that should be measured;
- **LIR** — Lower Rejection Limit;
- **LIS** — Lower Safety Limit;
- **LSS** — Upper Safety Limit;
- **LSR** — Upper Rejection Limit.

4.1.1.5 *Colaboradores* or Workers

This table contains information related to the company's workers, and each item represents a single worker. The attributes of this table can be described as follows:

- **Utilizador** — Primary key for *Colaboradores* table and the identifier of a worker;
- **Nome** — Name of the worker;
- **Estado** — State of worker, regarding if he/she either active or not;
- **Competencias** — Skills of the worker.

4.1.1.6 *Lotes* or Batches and *Consumos* or Tool use

These two tables contain information related to both the batches of manufactured products and the tools to manufacture those. There is no need to detail these tables since they are not part of the dataset used in the prediction task. The reason for this is later explained in Subsection 4.1.3.

4.1.2 Data Preprocessing

Data preprocessing is the process in which the data is modified so that its quality issues are found and fixed. These issues can be of several types, including wrong, duplicate, and missing values. The inconsistencies in data are relatively frequent and may often result either from the fact that data is extracted from more than one source or problems concerning the procedure of data gathering. In this project, data was corrected table by table.

The programming language used to handle the data the way it is going to be described was Python. Python is a simple yet efficient language that contains a vast amount of frameworks and libraries. One of these, *pandas*, a frequently used data manipulation library, was employed throughout the whole project.

4.1.2.1 *Medicoes*

In the *Medicoes* table, the focus was on correcting values related to the limits (*LIR*, *LIS*, *LSS*, *LSR*), measurement value (*Valor_Introduzido*), and measurement result value (*Result*).

Initially, rows with negative values in the first five mentioned attributes were removed. Afterward, rows with invalid limits were dropped, and the *Result* attribute was corrected when wrong. Limits were considered to be invalid when they were not according to the following list of rules:

- If at least three of *LIR*, *LIS*, *LSS*, and *LSR* are null;
- If *LIR* or *LSR* is null:
  - If *LIS* is greater than *LSS*.
- If *LIS* or *LSS* is null:
  - If *LIR* is greater than *LSR*.
- If *LIR*, *LIS*, *LSS* and *LSR* are all present:
  - If *LIR* is greater than *LIS*;
  - If *LIS* is greater than *LSS*;
  - If *LSS* is greater than *LSR*.

The next step was to replace the remaining missing values in the limits columns. Rows with only one or two missing values out of the four limit variables were imputed. On the confirmation that there are no inadequate limits and that all rows are complete, the *Result* value was confirmed in line with the following list of rules:

- If *Valor_Introduzido* is between *LIS* and *LSS*, then *Result* must be *0*;
- If *Valor_Introduzido* is between *LIR* and *LSR* but not between *LIS* and *LSS*, then *Result* must be *1*;
- If *Valor_Introduzido* is outside the range of *LIR* and *LSR*, then *Result* must be *2*.

Finally, outliers were identified and removed. Outliers are atypical and, usually, extreme values. It is essential to remove these anomalies since they increase variability significantly, which may result in more unsatisfactory predictive algorithm performance. The outlier rows were identified and dropped using standard deviation. If a value from the *Valor_Introduzido* or any limits columns fell outside a certain number of standard deviations then, the row was dropped.

The obtained table was *MedicoesCorr*.

4.1.2.2 *Registos*

In the *Registos* table, the actions taken were as follows:

- Replace in *CountLote* 'None' for *0*;
- Drops rows containing values in *Maquina* that had no correspondence to values in *Maquina* of the *Estruturas* table as it makes no sense to have references to machines that have no record in the database;
- Remove impossible values such as negative values in the *Quant* column.

The obtained table was *RegistosCorr*.

4.1.2.3 *Estruturas*

Regarding the *Estruturas* table, the following steps were taken:

- Drop rows with values in *Estado* that are not valid;
- Drop rows with values that do not follow the date and time format in *DtEntrega*;
- Clean the values in the *QtBatch* column and fill the missing values with the most frequent class.

The obtained table was *EstruturasCorr*.

4.1.2.4 *PIE*

The course of action for the *PIE* table was:

- In the four limits columns, blank spaces inside the numbers were removed, numbers with '..' instead of '.' as a notation to represent decimals were corrected, and 'None' strings were converted to actual Nan values;
- Clean *Cota_Descricao* values as each class was represented in several different ways;
- Rows with invalid limits were dropped. The rules for this removal are the same as the ones described in Subsection 4.1.2.1.

The obtained table was *PIECorr*.

4.1.2.5 *Colaboradores*

In the *Colaboradores* table, there was no need to carry out data processing operations.

4.1.3 Join Tables

After understanding how the data is structured and divided into several tables and data being cleaned, the following procedure is to fuse them into a single dataset containing all the data. Even though all the features distributed across the different tables might be useful in the prediction task, some are seemingly more relevant than others. For example, based on *Ishikawa's 6 Ms*, features related to the machine, method, workforce, and measurement are considered more important. These features can be found in *Medicoes*, *Registos*, *Estruturas*, and *Colaboradores* tables. However, data from other tables can also increase the knowledge in the production process. The procedures that are going to be described were performed using Python's library, *pandas*.

Initially, an inner join between *Medicoes* and *Registos* was performed, resulting in the *MedReg* table. Afterward, another inner join between *MedReg* and *Estruturas* was used to produce the *MedRegEst* table.

The next step was supposed to be joining the so far obtained table, the *PIE*, and *Lotes* table. Nevertheless, when attempting to execute the inner joins between these tables, the number of rows in the dataset decreased excessively. After a more careful inspection of the database, it was found that there probably was missing data, which meant that an inner join would result in loss of information. This being said, it was decided to use left joins to include data from *PIE* and *Lotes*. Once again, a problem was found since the *Lotes* table had almost no correspondence to *MedRegEst*. It was then determined that the *Lotes* table would drop and not be included in the final dataset. The table attained at the end of this step was called *MedRegEstPIE*.

Finally, only *Colaboradores* and *Consumos* tables were missing in the dataset. The first one was successfully added through a left join, and it was found that the latter had the same problem as the *Lotes* table. The approach to solving this issue was the exact same as to the one in the similar complication. The resulting table was *MedRegEstPIECol*, and some final cleaning operations were performed on this table. The most relevant ones were:

- Replacement of the diverse classes of *Medicoes Operacao* that referred to turning and milling operations into the simple *strings* "Tornear" and "Fresar";
- Replacement of the value '2' in the *Medicoes Result* variable by '1' since both values already represent a nonconformity. By doing this, the variable that is to be predicted becomes a binary one;
- Drop certain columns that were either redundant, represented primary or foreign keys in the database, their values were already a consequence of the nonconformity, or that were not part of the production process itself.

After concluding all these operations, the *MedRegEstPIECol* table contained twenty-two features and over one million and four hundred thousand rows.

It should be noted that when adding data from the *Colaboradores* table, both the name and state of the worker were removed due to privacy and data protection reasons.

4.1.4 Variable Analysis

At this point, the table named *MedRegEstPIECol* had the complete dataset, which included over one million and four hundred rows of data and twenty-two features. Considering its dimension, it is understandable that it is rather hard to retrieve any information on the data by just observing it without analysis tools. With this in mind, the data was graphically visualized using different types of charts, according to the adequate representation for each variable. Pie, scatter, and box plots, histograms, and bar charts were used. Besides these, reports with some data properties were also created.

Both plots and reports are shown in Appendix A.

4.1.5 Dataset Selection

In a manufacturing process, it is highly likely that the operation and the machine on which the operation is being performed are the two most important factors when obtaining nonconformities. Also, according to the Pareto principle, around eighty percent of the consequences come from twenty percent of the causes. Therefore, two Pareto diagrams were created in order to understand the distribution of operations and machines in rows that corresponded to nonconformities. They can be seen in Figure 4.2 and in Figure 4.3.

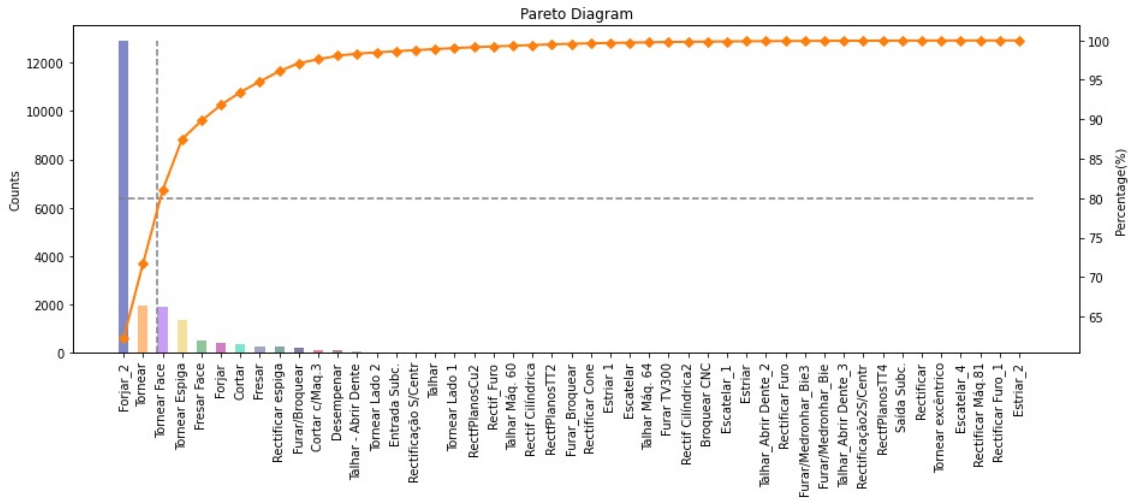

Figure 4.2: Pareto diagram for *Operacao* variable.

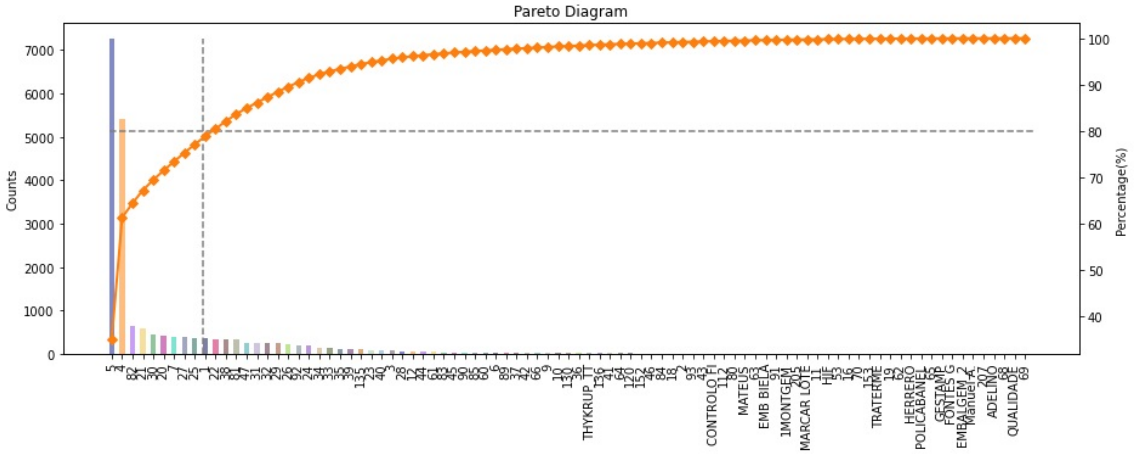

Figure 4.3: Pareto diagram for *Maquina* variable.

At this time, it was thought that attempting to create predictive models that could work for all the machines would be counterproductive. This is because models would have to be extremely generic to predict for different machines and operations effectively. Considering this, it was decided to restrict the dataset to a subset of itself that contained only the data about one machine. In order to decide which machine to use, a table of tuples was created. Each tuple was formed by an operation, a machine, and the number of occurrences of that operation in that machine. Once again, a Pareto diagram for the tuples was created (Figure 4.4).

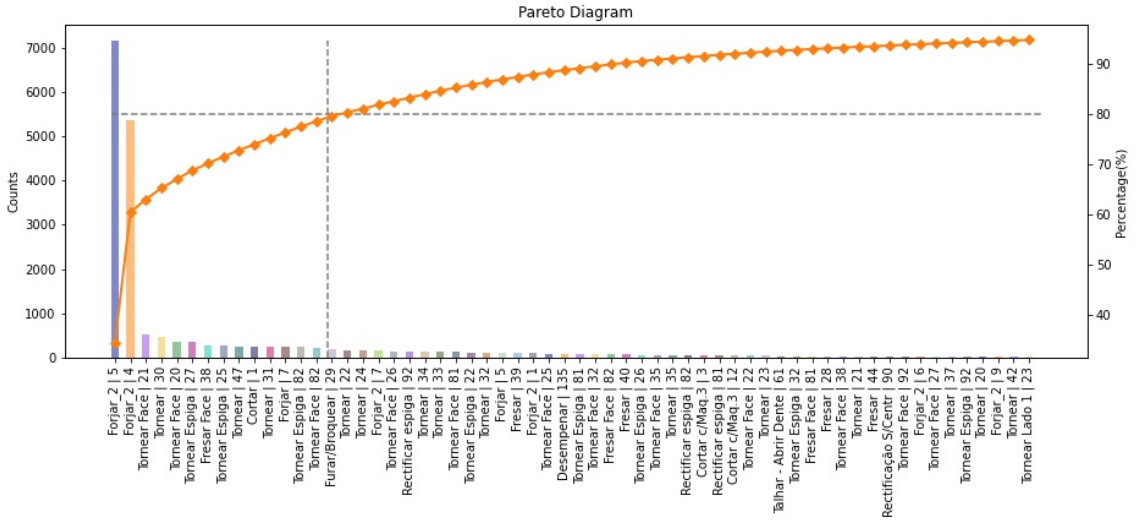

Figure 4.4: Pareto diagram for the tuples that consist in the combination of *Maquina* and *Operacao* variables.

From this diagram, we understood that the tuple with the highest number of occurrences consisted of machine *5* with operation *Forjar_2*. However, this work is focused on predicting nonconformities in two specific operations:

- **Turning** — Operations that are named *Tornear*
- **Milling** — Operations that are named *Fresar*

Considering this, the relevant tuple with the highest number of occurrences contained operation *Tornear Face* in machine *21*. With this information, we were able to restrict the whole dataset to one that contained only information about a single machine, *Maquina 21*. After obtaining this new subdataset, the rows were ordered by the *DtHora* variable. By doing so, we were able to obtain the sequence of events that may or may not have led to nonconformities. This sequence is called a *series*.

4.2 Feature Engineering

Feature engineering is a common task in machine learning projects, which consists of creating new features that may or may not assist the algorithms in their prediction duty. Although it is not a necessary task, doing feature engineer may allow for more accurate predictions. Moreover, even if they are not relevant or are even harmful to the predictive models, feature selection can ensure that features stay or are discarded.

4.2.1 Created Variables

As mentioned in Section 2.3, when the process that is being dealt with is a manufacturing process, some characteristics are usually related to nonconformities in products. Bearing this in mind, ten features were added to the dataset. Once again, the library used in this process was Python's *pandas*. Each one of the new features is explained in the following subsections.

4.2.1.1 *SeriesDuration*

SeriesDuration is one of the created variables. Its value was calculated using *DtHora* variable, date and time, and represents the duration of the series. As an assumption, nonconformities might be caused by series either too long or too short in terms of duration.

4.2.1.2 *SeriesOpChange*

SeriesOpChange is a new attribute that was computed using operations, *Operacao* values, within the series. It contains the number of different operations performed in that machine during the series. According to Ishikawa, the method applied in manufacturing is essential. Therefore, a higher number of different operations in a series may translate into a higher chance of obtaining nonconformities.

4.2.1.3 *ProductCotas*

ProductCotas represents the number of different *Cotas*, dimensions, a product has. Each different value in *Cotas* for the same product is one different dimension of the product that has to be

measured. The more dimensions a product has, the more complex it is and the more prone to nonconformities it might be.

4.2.1.4 *ProductOps*

ProductOps holds the number of different *Operacao*, operations, that can be performed on a manufactured item. It is similar to *ProductCotas* in the sense that the more operations conducted in an item, the more complex it is.

4.2.1.5 *UtilizadorOps*

UtilizadorOps is the number of different *Operacao*, operations, one *Utilizador*, worker, can perform. According to Ishikawa, the skill of the workforce when working in certain processes is relevant. This new variable attempted to represent the specialization of the workers in the several possible operations.

4.2.1.6 *UtilizadorExperience*

UtilizadorExperience contains the time, in days, since that *Utilizador*'s first record. With this information, it is possible to know how long the worker has been working in this production line. More inexperienced workers may lead to higher chances of getting nonconformities.

4.2.1.7 *SeriesUtiliz*

SeriesUtiliz is another variable created based on the series. It has the number of different *Utilizador*, workers, operating in the series. The more changes occur in a short duration of time, the more vulnerable the process might be to manufacture products with nonconformities.

4.2.1.8 *UtilizadorDailyOps*

UtilizadorDailyOps contains the number of different *Operation*, operations, an *Utilizador*, worker, performed earlier during the day. A tired or a still too fresh worker can mean higher chances of getting nonconformities.

4.2.1.9 *SeriesMeanVI*

SeriesMeanVI is the mean of the *Valor_Introduzido* in the series. Obtaining the mean of numeric values is a common practice in feature engineering as it is able to represent its series slightly.

4.2.1.10 *SeriesNonConformities*

Finally, *SeriesNonConformities* contains the number of rows containing nonconformities in the series. Using this value, the model can understand whether the presence of nonconformities in the recent operations can affect or not the appearance of new ones.

4.2.1.11 Final Operations

After creating the variables described in the previous subsections, the following columns were dropped since they no longer had use:

- *Medicoes Valor_Introduzido* was convenient during the verification of the *Result* variable and, besides that, its value is already a consequence of the production or not of nonconformities. Nonetheless, its value was used in the generation of the *SeriesMeanVI* variable;
- *Medicoes DtHora* was used to sort the dataset by the date of product manufacture and was also used in the creation of the new features *Series Duration*, *UtilizadorExperience*, and *UtilizadorDailyOps*;
- *Registos Maquina* no longer had use since the whole dataset refers to the same machine;
- *PIE Cota* was employed in the creation of the *ProductCotas* feature.

4.2.2 Created Variable Analysis

As explained in Subsection 4.1.4, graphically displaying data is indispensable to understand how it is distributed. Identical to that previous analysis, the created variables were also studied at this point. The plots and reports on this data are shown in Appendix B.

4.3 Model Training

In machine learning, model training is a process in which data is fed as input to an ML algorithm so that this one can discover patterns on the dataset. Then, when receiving data it has never encountered, a trained model is able to predict, with some degree of accuracy, the outcome of the new data.

In this section, both the model training itself and some preparations methods and techniques towards it are described. The programming language was Python, and the two most relevant libraries were *pandas* for data manipulation and *sklearn* for ML support.

4.3.1 Encoding

At this point, the dataset consisted of over sixty thousand rows, each with twenty-seven columns, each representing a variable. Before advancing to model training, some almost mandatory operations need to be performed in the dataset so that it becomes suitable to be used by the different classifiers.

One of the preparations needed is the encoding of data. Encoding is a technique used to transform categorical data or other types of characters sequence into a useful format in its context. In this case, the encoding was performed by transforming some data into numbers and was achieved by making use of two different methods. The pros and cons of each of them will be evaluated in the following two subsections.

4.3.1.1 Label Encoding

Label encoding consists of replacing each categorical value in a column with a numeric one so that the same unique numeric value replaces all the same categorical ones. This is a simple method that benefits from the fact that it does not create new variables, maintaining the dataset's dimensionality. Nevertheless, this technique suffers from the fact that numbers have an order and the ordinal relationship may not make sense when looking at what each number represents. This is a major issue when dealing with classifiers such as Logistic Regression, SVM, or Artificial Neural Networks that can not deal with this type of encoding. However, since Random Forest classifiers can deal with it, a dataset with this encoding was produced for this classifier training. The number of columns contained by this dataset did not suffer changes, staying at twenty-seven.

4.3.1.2 One Hot Encoding

As an alternative for the label encoding technique, one hot encoding was also used. In this method, for each unique categorical value, a binary, usually called *dummy*, variable is created containing only *0* and *1*. This practice does solve the problem related to the ordinal relationship addressed in Subsection 4.3.1.1 at the cost of drastically increasing the dataset's dimensionality. A dataset created with the help of this type of encoding was created for the use of Logistic Regression and ANN classifiers. It contains a total of 2726 features.

4.3.2 Scaling Data

One of the big issues with data is the fact that the range of values can differ hugely between variables. An often-used example is that one can think of the height and weight of a person as variables. A single unit of measure difference in both affects more the first feature since its range is smaller. This means that, for the classifiers that use distance measures, the variables with wider ranges are dominant over other variables regarding the objective function. As a result, data scaling or data normalization is required. This method normalizes data so that all features are around the same range of values.

Scaled data is quite simple to obtain when using the *sklearn* library. It contains a module developed specifically for scaling, and, in this case, the programming method *StandardScaler* was used. It normalizes data by reducing the mean to a value of zero and scaling to unit variance.

4.3.3 Feature Selection

As mentioned in Section 3.3, feature selection is a process that can reduce the set of features that serve as input for the predictive model. The eliminated features are the ones that are considered to be harmful to the model since they decreased its predictive capability. In this project, both RFE and PCA were used to decrease the dataset dimensionality. Both methods were applied with the help of the *sklearn* library.

4.3.3.1 RFE

Initially, Recursive Feature Elimination was used as the primary feature selection technique. As explained in Section 3.3.1, RFE is an iterative method that wraps the machine learning algorithms. In order to find the subset of variables that get the best ML algorithms' performance out of all features in the training dataset, RFE has to build and rebuild several different models. The more existent features in the entire dataset, the more computationally expensive this technique becomes. Although it was possible to use this method for the dataset obtained through the label encoding and explained in Subsection 4.3.1.1, when dealing with higher dimensionalities, its capabilities become more and more reduced.

4.3.3.2 PCA

The second feature selection technique used was Principal Component Analysis. PCA is a statistical and classifier-independent technique that reduces the dimensionality of the dataset without the cost of decreasing the accuracy of the model significantly. It determines the principal components, which are created as linear combinations of the features initially present in the training dataset. PCA is exceptionally efficient when dealing with datasets that contain correlated variables. After using PCA, the features or principal components obtained are the ones that are able to explain most of the variability in the data.

As has been said before in Subsection 4.3.1.2, the training dataset obtained through one hot encoding has high dimensionality. Due to this, PCA was preferred over RFE since the time consumed by the latter to perform a single run was excessive.

4.3.3.3 PCA with RFE

As mentioned in Subsection 4.3.1.2, the dataset obtained through one hot encoding contained 2726 features. Although it was also mentioned in Subsection 4.3.3.2 that PCA was preferred over RFE, the large number of features forced the use of the combination of the two previous methods to reduce the number of dataset's attributes to lower values.

4.3.4 Splitting Data

The split of data into training and test is a crucial task to be performed in any machine learning project. Since test data is not used in the training phase, when testing the model with it, it is possible to understand whether the algorithm has actually been able to learn the patterns hidden in the dataset or not.

In this project, the value chosen to be the split percentage was 25%, meaning that 75% of the whole dataset was used for training and 25% for testing purposes. Once again, Python's library *sklearn* was used to complete this operation.

4.3.5 Dealing with Imbalance Problem

An imbalanced dataset is an issue in a classification problem. This occurs when the class distribution of the feature that is supposed to be predicted is skewed. Bearing in mind that most classifiers assume that the data is balanced and that, in most cases, the minority class is the focus of the machine learning problem, there is the need to balance the data. The imbalance in the dataset that is being manipulated in this project is shown in Figure A.5.

In order to balance the data, one can use:

- **Oversampling** — Duplicate random examples of the minority class
- **Undersampling** — Delete random examples of the majority class

On the one hand, oversampling can help to balance the distribution, but on the other hand, the duplication of examples does not add information to the process. Regarding undersampling, the elimination of random examples leads to the loss of information on the process.

With the intention of neither losing information nor adding redundancy to the dataset, **SMOTE** or Synthetic Minority Oversampling Technique was used. This is an oversampling technique that, instead of simply duplicating random examples, synthesizes new ones. SMOTE generates examples that are close to already existing ones of the minority class.

For this stage of the work, a library called *imblearn* was used to balance the data. It is a complement library of *sklearn* that provides methods for the use of SMOTE. Towards an acceptable balance of the data, while still attempting not to change the class distribution drastically, it was decided to oversample the data using SMOTE so that the ratio of samples of the minority class over the majority one would be one to four after the oversampling. Finally, it should also be noted that this operation was only performed in the training dataset and not in the test one so that the original values suffer no changes.

4.3.6 Classifiers

A classifier is a machine learning algorithm that is able to learn the hidden relationships between the several features that constitute the dataset and the label attribute, the one that is predicted. As it was mentioned in Section 3.4, taking into account the specifications of this project, several different classifiers can be used to deal with this supervised binary classification problem, such as:

- **Logistic Regression**
- **k-Nearest Neighbors**
- **Decision Trees**
- **Support Vector Machine**
- **Naive Bayes**
- **Artificial Neural Networks**

From these, kNN and Naive Bayes were excluded right from the beginning of the model training due to their lack of efficiency when dealing with high-dimensional data. SVM also has the same issue and does not deal well with overlapped classes. Besides this, it was also decided to use Random Forests instead of Decision Trees.

The four remaining classifiers were applied using the *sklearn* library.

4.3.6.1 Metrics

It is fundamental to evaluate any model's performance to understand if it can be used in the prediction task or not. Therefore, with the intention of assessing the performance of the model, several metrics can be used, including:

- **Precision** — Relates the number of samples that were correctly labeled as one of the target classes and the ones that were also labeled as that class but belong to the other. For example, the formula used to compute the precision for the positive class is:

$$Precision = \frac{TruePositives}{TruePositives + FalsePositives}$$

- **Recall** or **Sensitivity** — Relates the number of samples that were correctly labeled as one of the target classes and the ones that were labeled as the other despite belonging to the first one. The formula for, for example, the positive class would be:

$$Recall = \frac{TruePositives}{TruePositives + FalseNegatives}$$

- **F1 score** — It is the weighted average of the precision and recall, in which the contribution of both is the same.

$$F1 = 2\frac{Precision * Recall}{Precision + Recall} = \frac{2TruePositives}{2TruePositives + FalsePositives + FalseNegatives}$$

- **ROC AUC** — Area Under the Receiver Operating Characteristic Curve is probably the most complex metric. It represents the probability of the classifier in use to predict a higher value for a positive random sample than for a negative random one;

- **Accuracy** — The most easily understandable metric, and it represents the ratio of correct predictions. The formula used to calculate the model's accuracy is as follows:

$$Accuracy = \frac{TruePositives + TrueNegatives}{ActualPositives + ActualNegatives}$$

- **Balanced Accuracy** — Similar to the above metric, but it is used when dealing with imbalanced datasets. The accuracy is computed taking into account the number of samples of

each class. The formula is:

$$BalancedAccuracy = \frac{1}{2}(\frac{TruePositives}{ActualPositives} + \frac{TrueNegatives}{ActualNegatives})$$

These metrics were used to evaluate the models' performances as they were being developed and improved.

4.3.6.2 Parameter Tuning

Each classifier contains several parameters that differ from classifier to classifier. The classifiers use these parameters to change and influence the learning procedure and, since they can not be learned, they need to be passed as arguments. Hence, to discover the best values for at least some of the parameters, there is the need to make parameter tuning. This is the process in which the set of optimal parameters is attempted to be found.

Regarding the programming approach, *GridSearchCV* and *Optuna* were used. While *GridSearchCV* is a function that is included in a *sklearn* module, *Optuna* is a parameter optimization framework for Python that was chosen due to its lightweight, efficiency, and easy integration with this library. Both methods receive as an input certain ranges of values and test combinations of these, measuring the accuracy of the models when using those parameters. The obtained optimal values are available in Chapter 5.

4.4 Summary

This chapter is divided into three sections. Section 4.1 introduces the process behind obtaining the dataset. It starts with the description of the database, addresses the preprocessing of data, the join of the multiple database tables, the analysis of the features, and, finally, the selection of the data that will be part of the training dataset.

Section 4.2 contains the explanation of the variables that were built from the existing attributes. It also contains the analysis of these new features.

Section 4.3 describes the processes that lead to the training of the models. More specifically, it addresses data encoding, scaling, splitting, and oversampling. It is also considered the classifiers used in training, the metrics that can evaluate them, and how the parameter tuning was performed.

Chapter 5

Results

In this chapter, the obtained experimental setup results are first grouped by the used classifier and then provided and evaluated. Considering that this project was related to a predictive classification problem, it makes sense to evaluate the different models using metrics such as overall accuracy, balanced accuracy, precision, recall, f1-score, and ROC AUC.

The presented results are the final ones. During the development of this work, the pipeline of the processes responsible for both the obtainment of the dataset and the training using that dataset was rearranged and improved to upgrade the final results.

5.1 Logistic Regression

In order to acquire the results for the Logistic Regression classifier, the following tasks were performed:

1. The dataset obtained through one hot encoding was used;
2. Data was scaled;
3. PCA was used to reduce some of the dataset dimensionality and remove correlated variables;
4. Data was split into training and test dataset;
5. Oversampling the training dataset by using SMOTE;
6. RFE was applied to lower the number of features to fifty;
7. *Optuna* framework helped to find the best values for the parameters:
    - *C* — Inverse of regularization strength;
    - *max_iter* — Maximum number of iterations until the solver converges;
8. Training the model with the optimal parameters and testing it with the test dataset.

Using 18.683 as the optimal value for parameter *C* and 158 for *max_iter*, the obtained confusion matrix of the model is shown in Table 5.1 and the different metrics are shown in Table 5.2.

		Predicted Values	
		0	1
Actual Values	0	14661	462
	1	29	114

Table 5.1: Confusion Matrix for the Logistic Regression classifier.

Target Class	Precision	Recall	F1-Score	Accuracy	Bal. Accuracy
0	1.00	0.97	0.98	0.968	0.883
1	0.20	0.80	0.32		

Table 5.2: Results using the Logistic Regression classifier.

5.2 Random Forest

For the use of the Random Forest classifier, the following operations were executed:

1. The dataset obtained through label encoding was used;
2. Data was split into training and test dataset;
3. Oversampling the training dataset by using SMOTE;
4. *Optuna* framework helped to find the best values for the parameters:
   - *n_estimators* — Forest's number of trees;
   - *max_depth* — Maximum depth of each tree;
5. Training the model with the optimal parameters and testing it with the test dataset.

Since the Random Forest algorithm has only one single dataset for the many decision trees it comprises, each tree only receives a random sample of the whole data. Due to this randomness, each time the model is trained, the trees are created differently, resulting in non-identical outcomes. Considering this and with the intention of securing the best possible results, each of the five best results was trained five times, and the average of the metrics was then computed. The best average obtained confusion matrix of the model is shown in Table 5.3, and its different metrics are shown in Table 5.4. These tables were achieved by using, as parameters, 82 for *n_estimators* and 17 for *max_depth*.

		Predicted Values	
		0	1
Actual Values	0	15022	108
	1	42	94

Table 5.3: Confusion Matrix for the Random Forest classifier.

Target Class	Precision	Recall	F1-Score	Accuracy	Bal. Accuracy
0	1.00	0.99	1.00	0.990	0.842
1	0.47	0.69	0.56		

Table 5.4: Results using the Random Forest classifier.

5.3 Artificial Neural Network

So that the ANN classifier could be applied, the following actions were taken:

1. The dataset obtained through one hot encoding was used;
2. Data was scaled;
3. PCA was used to reduce some of the dataset dimensionality and remove correlated variables;
4. Data was split into training and test dataset;
5. Oversampling the training dataset by using SMOTE;
6. RFE was applied to lower the number of features to fifty;
7. *GridSearchCV* helped to find the best values for the parameters:
    - *hidden_layer_sizes* — Number of hidden layers and number of neurons in each;
    - *activation* — Activation function for the hidden layer;
    - *solver* — Solver for weight optimization;
    - *alpha* — Regularization term;
    - *learning_rate* — Learning rate schedule for weight updates;
8. Training the model with the optimal parameters and testing it with the test dataset.

Like the Random Forest classifier, ANN can also produce different values even when trained with the exact same parameters and dataset. In this case, the randomness is mainly due to the random assignment of initial weights. Once again, much like the process described in Section 5.1, each of the five best results was trained five times, and the average of the metrics was then computed. The best average obtained confusion matrix of the model is shown in Table 5.5, and its different metrics are shown in Table 5.6. The values used for these results were the rectified linear unit function for the *activation* parameter, 0.0001 for *alpha*, a constant *learning_rate*, the Adam method for stochastic optimization in the *solver*, and, finally, one hidden layer with a total of forty neurons for the topology of the ANN, *hidden_layer_sizes*.

		Predicted Values	
		0	1
Actual Values	0	15036	99
	1	53	78

Table 5.5: Confusion Matrix for the Artificial Neural Network classifier.

Target Class	Precision	Recall	F1-Score	Accuracy	Bal. Accuracy
0	1.00	0.99	0.99	0.990	0.794
1	0.44	0.60	0.51		

Table 5.6: Results using the Artificial Neural Network classifier.

5.4 ROC Curves Comparison

In Sections 5.1, 5.2, and 5.3, the three models in which the experiments were performed were analyzed, considering the precision, recall, f1-score, accuracy, and balanced accuracy. Therefore, the only metric from Subsection 4.3.6.1 that was still not being assessed was ROC AUC. ROC is a probability curve representing the trade-off between recall, sensitivity, or TPR (True Positive Rate) and specificity or TNR (True Negative Rate).

$$TPR = \frac{TruePositives}{ActualPositives + FalseNegatives}$$

$$TNR = \frac{TrueNegatives}{TrueNegatives + FalsePositives}$$

and AUC is calculated from the area under that curve.

AUC or Area Under the Curve is computed from the ROC curve and represents the area under it. The better the model, the further the curve is from a forty-five-degree diagonal, and the closer the AUC is from 1,0. The ROC curves and their respective AUCs are shown in Figure 5.1.

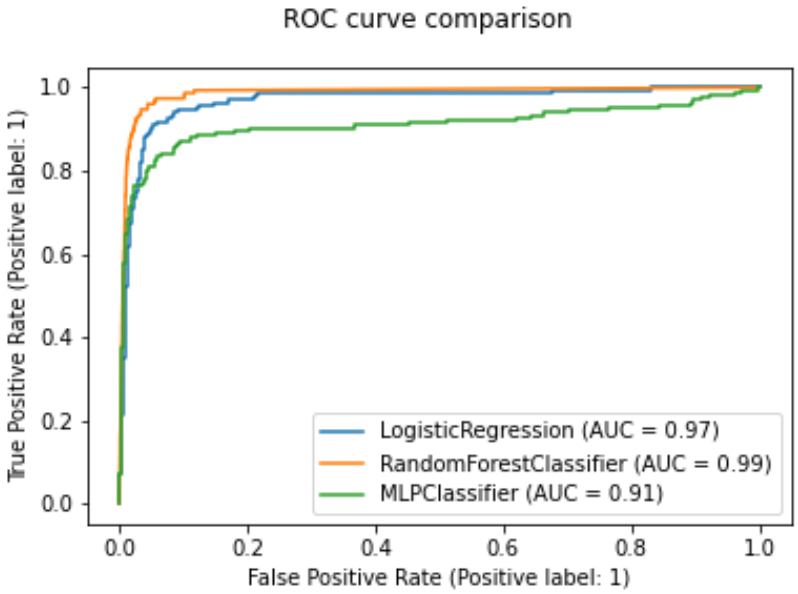

Figure 5.1: Comparison of ROC curves and respective AUCs for Logistic Regression, Random Forest, and Artificial Neural Network classifiers.

5.5 Summary

This chapter is divided into three sections. Section 5.1 describes all the actions performed to train the Logistic Regression classifier and presents the obtained confusion matrix and metrics for the optimal parameters of its model.

Section 5.2 describes all the actions performed to train the Random Forest classifier and presents the obtained confusion matrix and metrics for the optimal parameters of its model.

Section 5.3 describes all the actions performed to train the Artificial Neural Network classifier and presents the obtained confusion matrix and metrics for the optimal parameters of its model.

Section 5.4 contains the comparison of the ROC curves and their respective AUCs between each of the classifiers mentioned in the previous sections.

Chapter 6

Discussion and Conclusions

This final chapter comprises the summary of this dissertation, the discussion of the obtained results, and, finally, the possible future improvements to this work that were considered to be relevant in this context.

6.1 Overview of Developed Work

The purpose of this dissertation consisted of investigating the possibility of making use of machine learning models to predict the creation of nonconformities in a production line of precision parts.

In order to perform the previous task, the initial focus was on exploring the state-of-the-art, which was addressed in Chapters 2 and 3. As a result, these chapters included the already existing information on machining and manufacturing and the current methodologies, techniques, and algorithms that are being applied in the fields of Data Mining and Machine Learning.

After studying and understanding the concepts behind the modern methods related to the scope of this dissertation, it was time to start developing the machine learning project described in Chapter 4. In an initial stage, data was transformed with the aim of correcting and cleaning it, the data's different sources were fused, the obtained dataset was analyzed, and some features were created based on the current ones, as was reported in Sections 4.1 and 4.2. The next step was to prepare the dataset so that the different machine learning classifiers that were to be used according to the initial plan could handle it, Section 4.3. This process encompassed various techniques such as data encoding, scaling, balancing, splitting, and variable selection. Finally, as a result of this project following a CRISP-DM methodology which has an iterative approach, the whole process was repeated a few times so that the results could be improved incrementally.

6.2 Results Discussion

When evaluating the results of a prediction problem, it is essential to understand which metrics should be used to assess how accurate each model is and how it performs. Looking at only one or two metrics may be misleading due to some of the dataset's characteristics. For example, in the case of this dissertation, the provided dataset was very imbalanced. As a result, all the classifiers got an overall accuracy of over 95%. As a matter of fact, if a model only classified the data presented to it as a conforming product, due to the skew in the distribution, the model would also get an accuracy of over 95%. Taking this into account, several metrics were used. For each of the classifiers, their **confusion matrix**, **precision**, **recall**, and **f1-score** of each target class, as well as the overall **accuracy** and **balanced accuracy** were measured.

Looking at the tables of Chapter 5 that contain the information about each metric, it is possible to see that, for all the classifiers, the precision, recall, and f1-score for the target class *0* was always almost 1,00. However, this is quite normal taking into account the distribution of the target classes. In prediction problems involving imbalanced datasets, it is natural to give more relevance to the minority class, *1*, as, usually, the error in its classification is more significant.

One thing that was common to all of the classifiers was that, during the improvement of each one, an increase in balanced accuracy, which contemplates the total number of samples of each target class, resulted in the decrease of overall accuracy. In general, when attempting to increase the number of true positives (positive samples labeled correctly as positive), the number of false positives (negative samples mislabeled as positive) was also growing. This effect was especially evident in the Logistic Regression classifier, which ended up resulting in a remarkable decrease in precision (Table 5.2). Nonetheless, this was also the classifier with the lowest number of incorrectly predicted nonconformities (Table 5.1), leading to the highest recall value out of all the predictive models. In addition, Logistic Regression also exhibited the highest balanced accuracy score.

Regarding the Random Forest results (Section 5.2), one can observe that it was the classifier with the most balanced performance across most of the presented metrics (Table 5.4). It detected more than two-thirds of the nonconformities, which is slightly below the Logistic Regression model results. However, it was able to significantly increase the precision of the model since not so many conforming products were classified as nonconforming. It was also observed during the experimentation stage that the use of PCA or data scaling in RF did not increase the model's performance. This is understandable as RF can handle both correlated and unscaled data. Although both the feature selection and the data preprocessing techniques were initially used before training the classifier, they were eventually dropped considering performance benefits did not outweigh the computational and time cost of performing those operations.

Artificial Neural Networks performed slightly worse than Random Forests. The reason for this is believed to be the larger number of parameters to tune and the complexity of these. While other classifiers were being optimized on two parameters, ANNs had five parameters for tuning, some of which have drastic effects on the classifier's performance. Furthermore, such a number of variables

creates a model containing too many degrees of freedom. For example, *hidden_layer_sizes* is the variable related to the topology of the neural network, and discovering the optimal value is a complex task as there are no rules to it.

Regarding the comparison of the ROC curves in Section 5.4, all the three classifiers obtained values of AUC above 0,90. Suppose one were to look only at these numbers by themselves. That person would probably think that the classifiers possess a near-perfect measure of separability, meaning that the models were able to almost always discern between the positive and the negative class. Nevertheless, the precision and recall metrics show that this is not precisely the case. The problem with AUC is the fact that it is not imbalance-insensitive.

Lastly, in order to better understand the general final results in a prediction problem, it is also necessary to evaluate the whole process that led to the model training and to the outcomes of it:

- A machine learning algorithm can only predict based on the data that it received during the training phase. In cases where the input data does not reflect, at least, most of the cases in the domain, the models may struggle when presented with new instances, especially when dealing with such particular situations as the manufacture of nonconformities in a production line;

- Nowadays, there is an overwhelming number of machine learning algorithms. Choosing and implementing them is a puzzling task that involves several variables. An inappropriate choice may lead to mediocre results since the chosen model might not be optimal for a particular dataset or domain;

- Every method or technique has its advantages and downsides. Some of the procedures performed during this work, although having increased the overall performance of the classifiers, may have also contributed to their limitations. For example, SMOTE, which is a commonly used technique to increase the total number of samples in the minority class, does not contemplate the majority one. This may lead to an intense overlap of the classes, which in turn, may generate ambiguous samples;

- Data preprocessing is one of the key and most time-consuming tasks during the development of a machine learning project considering that data quality has a massive impact on the performance of every classification model. Looking back at how this step in specific was conducted during this dissertation, it is now easy to understand that it should have been a more cautious and lengthy one. As the development went forward, it was required to take a step back several times to correct missing values, incorrect data, or inconsistent data formats. This led to delays in the project, and the time consumed could easily have been used in, for example, increasing models' performances.

6.3 Future Work

In a machine learning project, there is always room to improve. In this particular dissertation, some experiments and details were not carried out mainly due to the lack of time. Although the time management may not have been the optimal one, the fact that this was the first time performing a Data Science project also helped to some extent to the delay when accomplishing certain tasks. Besides this, the feature selection algorithm RFE or the hyperparameter optimization searches, for instance, required days to finish a single run when dealing with the complete dataset. Nevertheless, it feels like part of the future work can be identified. The list of experiments that may come after this work is as follows:

- New features may be created based on the current ones. As an example, workers' skills are one of the original attributes of the dataset. Although that was included in the final dataset, the relationship between the skills and the operations being performed by that worker was not analyzed. Definitively, a new binary variable could be generated so that the models could understand whether the worker officially has the skills to perform that operation or not. Another similar example could be a feature containing, for example, the number of times a certain worker has already performed a certain operation representing its experience doing it;

- One of the most obvious experiments that are still to be done is using other classifiers besides the ones already being used. Classifiers such as Naive Bayes, despite its relatively poor performance handling large datasets, and SVM could turn out to be quite successful when predicting in this domain. Likewise, AdaBoost and Gradient boosting are also two techniques that could be implemented;

- Hyperparameter optimization is an almost never-ending process. Continuing to adjust these values can help to improve the classification model's performances;

- Taking into account the limited data samples, cross-validation is an imperative procedure. Although it is being performed in the model training, a simple 5-fold cross-validation is being used. Initially, a repeated stratified 5-fold cross validator was applied, but this was too expensive in terms of time and computational resources;

- The last possibility and the one that presents the greatest challenge would be the exploration of online machine learning in this context. The work developed so far already accounts for the sequence of events ordered by time, *series*, but the models are only being trained with the entire dataset. With online methods, the algorithms would be able to adapt to new patterns in case these change from time to time.

6.4 Summary

This chapter is divided into three sections. Section 6.1 reviews the primary goal of this dissertation, explains what was done both in terms of domain study and in the experimental part.

Section 6.2 contains the discussion of the obtained results. It analyzes the models' scores, evaluating which one performed better regarding each metric, and examines the process that led to the results, describing some of the obstacles found along the way.

Section 6.3 lists some of the possible future improvements to the work developed during the dissertation.

Classifier	Target Class	Precision	Recall	F1-Score	Accuracy	Bal. Accuracy
LR	**0**	1.00	0.97	0.98	0.968	0.883
	1	0.20	0.80	0.32		
RF	**0**	1.00	0.99	1.00	0.990	0.842
	1	0.47	0.69	0.56		
ANN	**0**	1.00	0.99	0.99	0.990	0.794
	1	0.44	0.60	0.51		

Table 6.1: Results using the Logistic Regression classifier.

Target Class	Precision	Recall	F1-Score	Accuracy	Bal. Accuracy
0	1.00	0.99	1.00	0.990	0.842
1	0.47	0.69	0.56		

Table 6.2: Results using the Random Forest classifier.

Target Class	Precision	Recall	F1-Score	Accuracy	Bal. Accuracy
0	1.00	0.99	0.99	0.990	0.794
1	0.44	0.60	0.51		

Table 6.3: Results using the Artificial Neural Network classifier.

Appendix A

Variable Analysis

This appendix provides the charts and reports mentioned in Subsection 4.1.4.

A.1 Plots

A.1.1 Medicoes Operacao

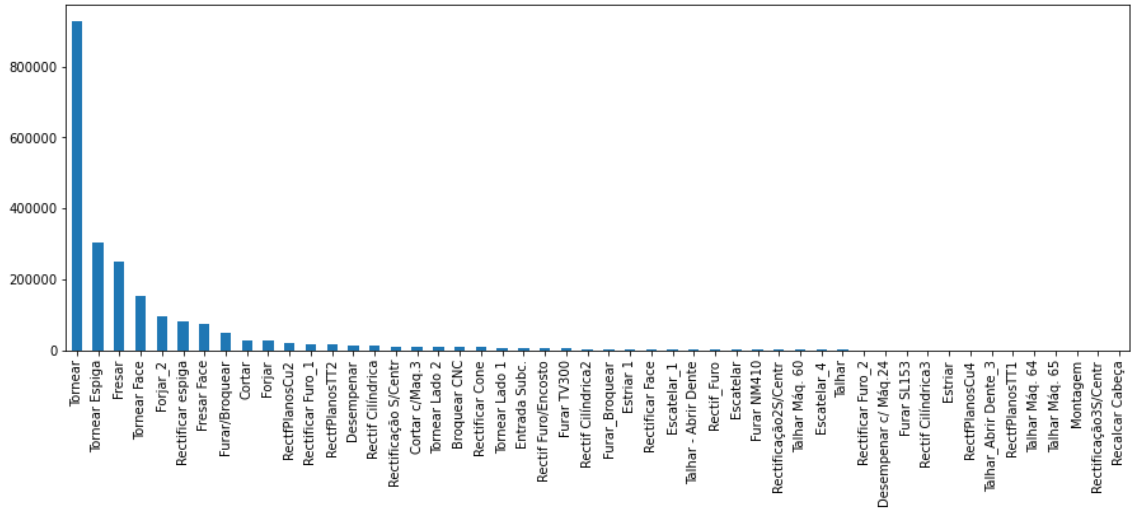

Figure A.1: A bar chart representing the distribuition of the 50 most frequent *Operacao* variable classes of the *Medicoes* table.

A.1.2 Medicoes Operacao

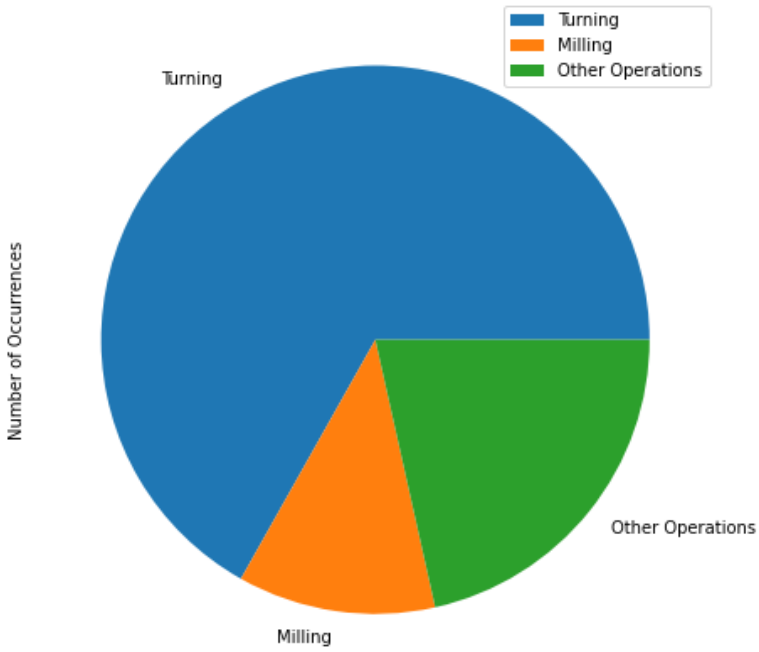

Figure A.2: A pie chart referring to the *Operacao* variable of the *Medicoes* table showing the distribution of the two most important operations as well as the rest of them grouped.

A.1.3 Medicoes Cota

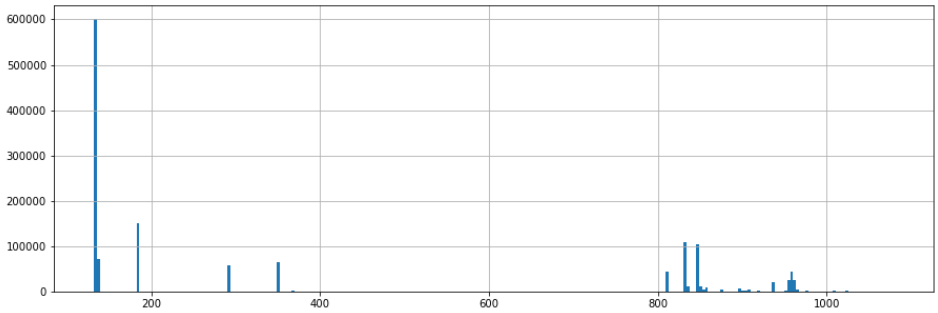

Figure A.3: An histogram representing the distribuition of *Cota* variable of the *Medicoes* table.

A.1.4 Medicoes LIR, LIS, LSS, LSR and Valor_Introduzido

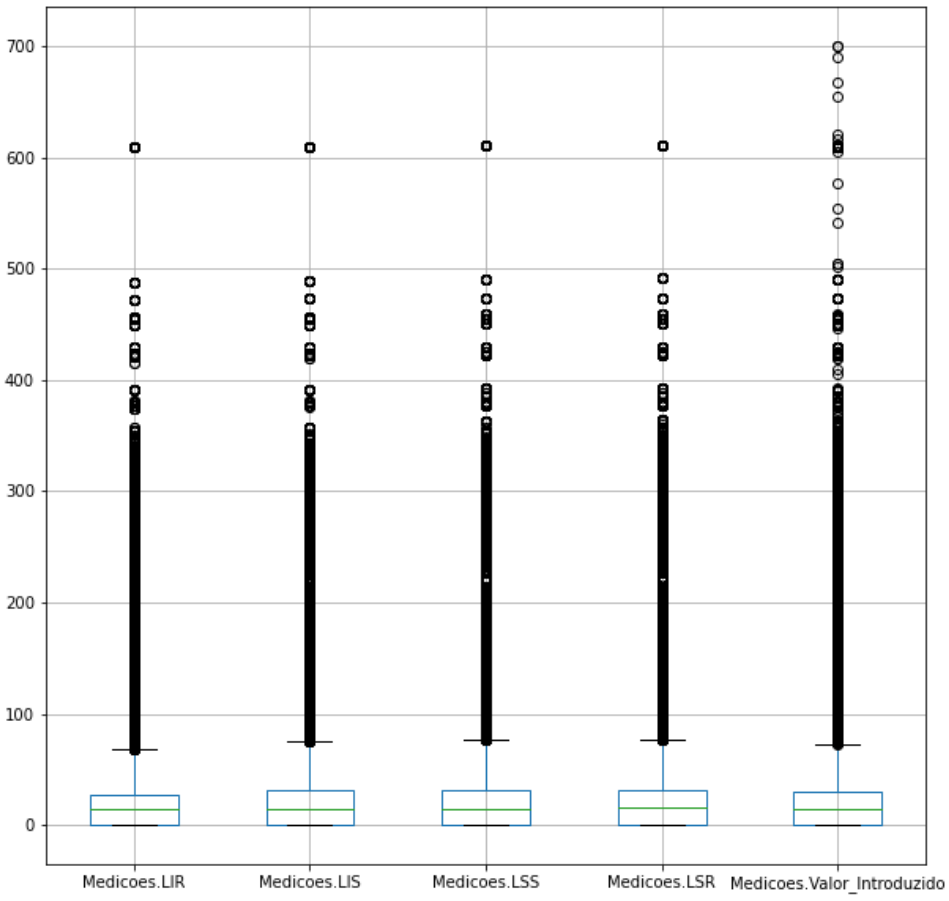

Figure A.4: A box chart graphically depicting *LIR*, *LIS*, *LSS*, *LSR*, *Valor_Introduzido* of the *Medicoes* table.

A.1.5 Medicoes Result

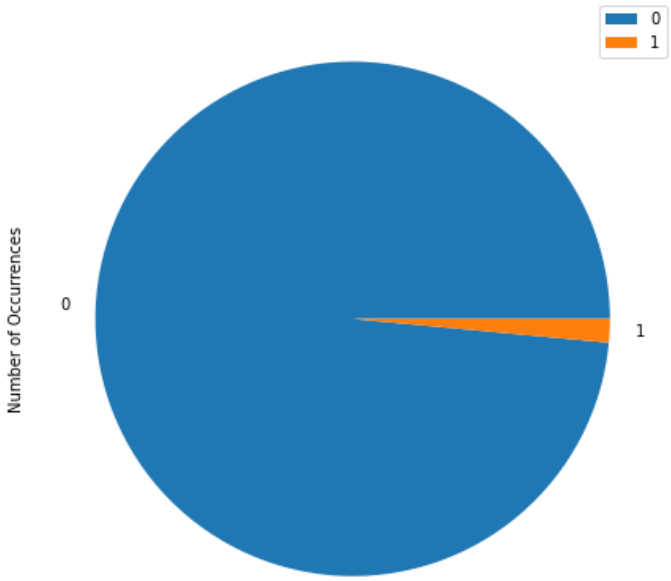

Figure A.5: A pie chart showing the distribution of the *Result* variable in the *Medicoes* table.

A.1.6 Medicoes Utilizador

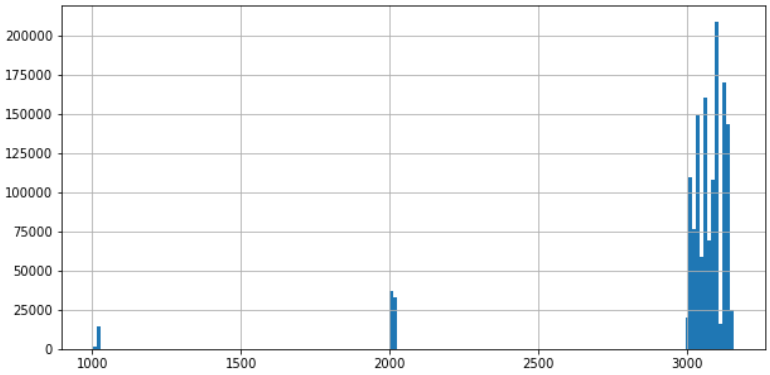

Figure A.6: An histogram representing the distribuition of *Utilizador* variable of the *Medicoes* table.

A.1.7 Registos Quant

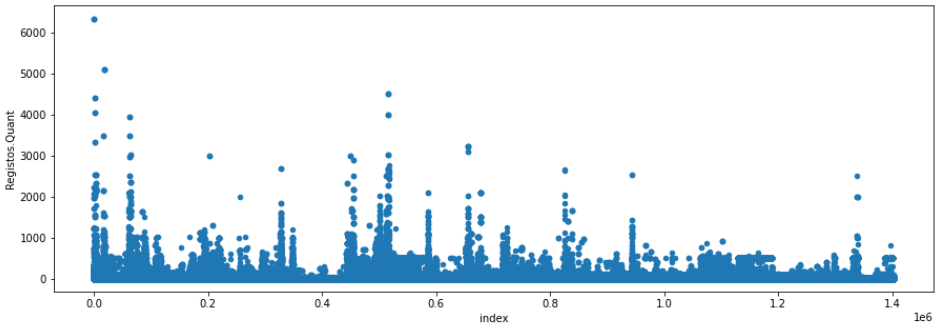

Figure A.7: A scatter plot representing the distribution of the *Quant* variable of *Registos* table across the dataset.

A.1.8 Registos Maquina

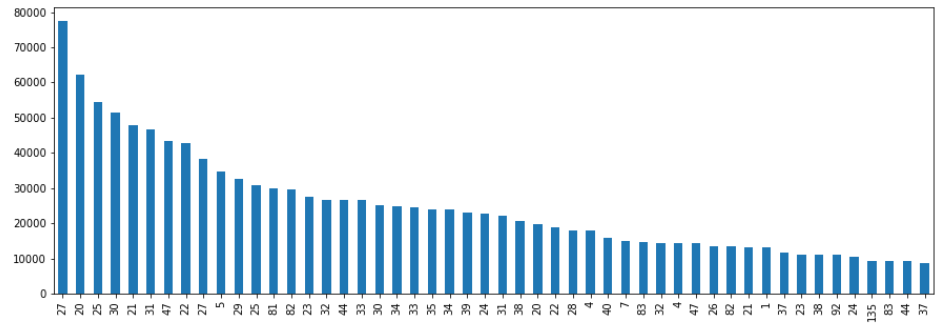

Figure A.8: A bar chart representing the distribuition of the 50 most frequent *Utilizador* variable classes of the *Registos* table.

A.1.9 Estruturas QtBatch

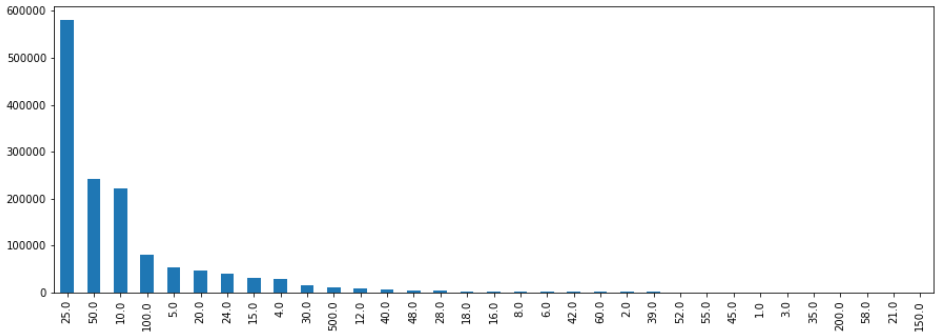

Figure A.9: A bar chart representing the distribuition of the *QtBatch* variable of the *Estruturas* table.

A.1.10 Colaboradores Competencias

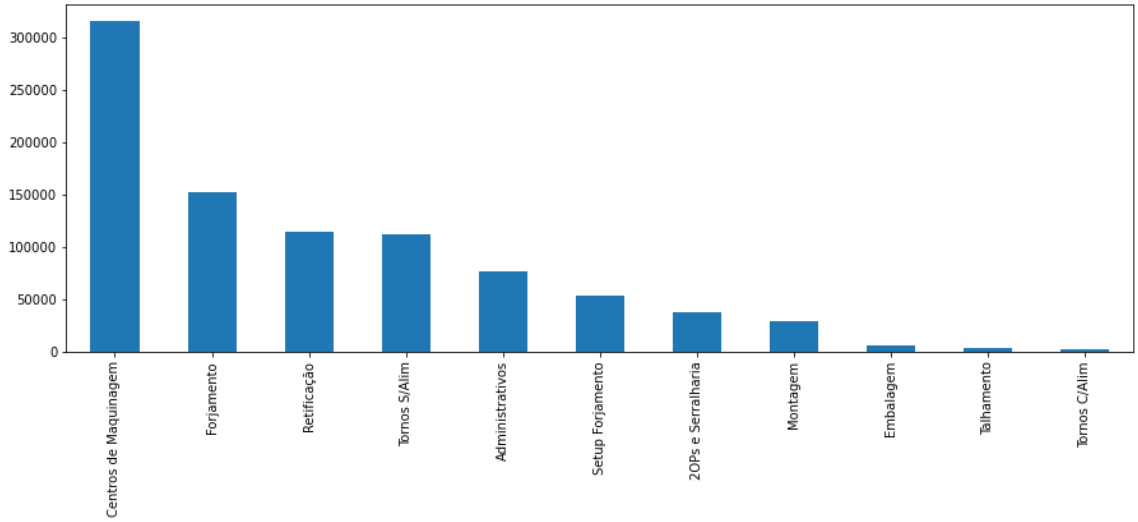

Figure A.10: A bar chart representing the distribuition of the *Competencias* variable of the *Colaboradores* table.

A.2 Reports

A.2.1 Numeric Variables

	Variable Name	Nan#	Type	Values#	Mean	Std. Dev.	Median	Max	Min
0	Medicoes.LIR	0	float64	1402088	26.568	41.741	14.7	609.5	0.0
1	Medicoes.LIS	0	float64	1402088	27.309	41.825	15.15	609.5	0.0
2	Medicoes.LSS	0	float64	1402088	28.249	42.854	15.26	610.5	0.01
3	Medicoes.LSR	0	float64	1402088	29.004	44.246	15.3	610.5	0.01
4	Medicoes.Valor_Introduzido	0	float64	1402088	27.427	42.241	15.01	700.0	0.0
5	Registos.Quant	0	float64	1402088	32.299	70.485	24.0	6337.0	0.3
6	Estruturas.QtBatch	0	float64	1402088	33.573	47.152	25.0	500.0	1.0

Figure A.11: A report on the numerical variables present in the complete dataset.

A.2.2 Categorical Variables

	Variable Name	Nan#	Type	Values#	Classes#	Mode Class	Mode Class#	Minimum Class	Minimum Class#
0	Medicoes.Utilizador	0	int64	1402088	140	3042	93856	1025	1
1	Medicoes.Operacao	0	object	1402088	78	Tornear	566887	MontCamb	5
2	Medicoes.Cota	0	int64	1402088	138	132	291858	1032	1
3	Medicoes.Result	0	int64	1402088	2	0	1381379	1	20709
4	Registos.Maquina	0	object	1402088	196	27	77416	87	1
5	Estruturas.OP	0	float64	1402088	34597	1402912.0	15658	1603388.0	1
6	Estruturas.Codigo	0	object	1402088	4153	6363003	54524	6030089999	1
7	Estruturas.Nivel	0	float64	1402088	227	2010.0	282075	510010.0	1
8	Estruturas.Nome	0	object	1402088	4144	Sprocket W 16T	54524	F3 Cambota Face Curt	1
9	PIE.PIE	435567	object	966521	5181	F3CF-RESPØ20/00	39158	F1BIE178FORJ/00	1
10	PIE.Metodo	435567	object	966521	170	PAQUIMETRO	411280	TAMPÃO LISO P/NP ø15	1
11	PIE.Cota	435567	float64	966521	117	132.0	235394	1032.0	1
12	PIE.Cota_Descricao	435567	object	966521	22	others	251146	21	137
13	Colaboradores.Competencias	876433	object	525655	10	Centros de Maquinage	146627	Talhamento	960

Figure A.12: A report on the categorical variables present in the complete dataset.

A.2.3 Date Variables

	Variable Name	Nan#	Type	NonConvToDate#	Youngest	Oldest	MaxDateDiff	MinDateDiff
0	Medicoes.DtHora	0	object	0	2017-04-20 16:16:24.670	2004-08-31 20:30:33.797	208 days 15:09:51.	0 days 00

Figure A.13: A report on the date variables present in the complete dataset.

Appendix B

Created Variable Analysis

This appendix provides the charts and reports mentioned in Subsection 4.2.2.

B.1 Plots

B.1.1 SeriesDuration

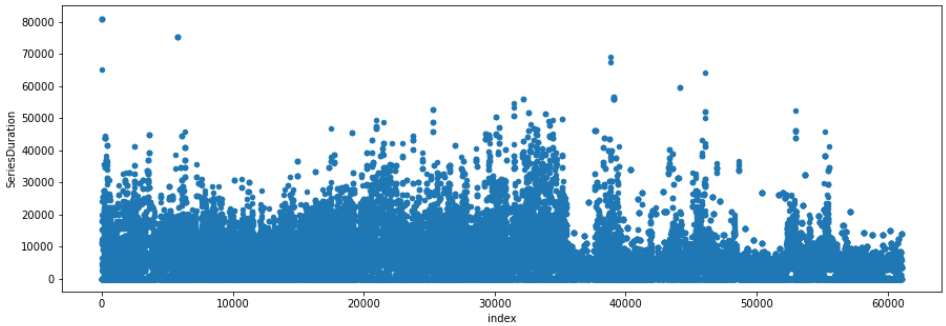

Figure B.1: A scatter plot representing the distribution of *SeriesDuration* across the dataset.

B.1.2 SeriesOpChange

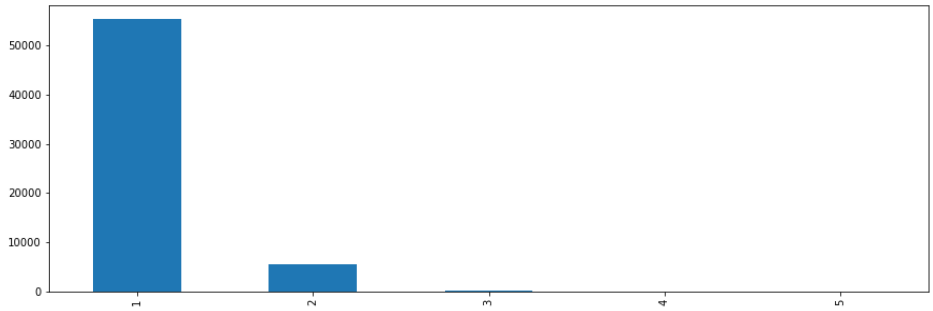

Figure B.2: An histogram showing the amount of times each value in *SeriesOpChange* feature is present in the dataset.

B.1.3 ProductCotas

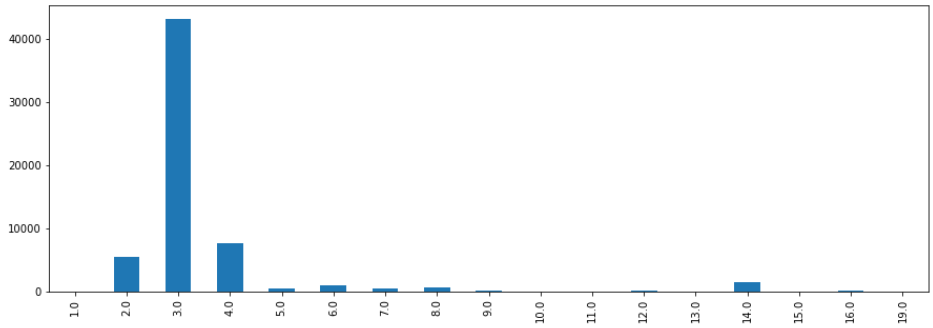

Figure B.3: An histogram showing the amount of times each value in *ProductCotas* feature is present in the dataset.

B.1.4 ProductOps

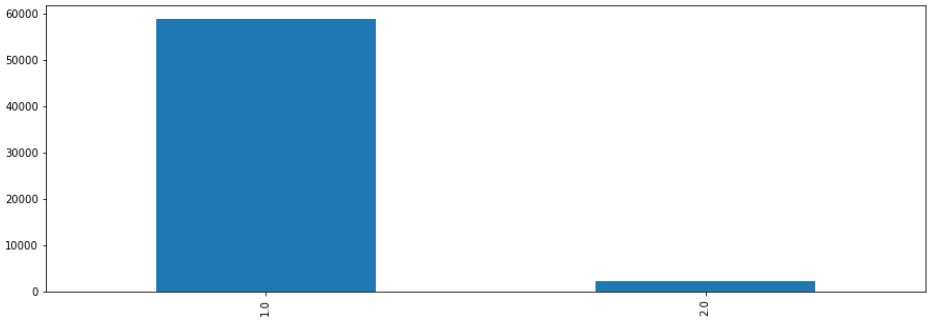

Figure B.4: An histogram showing the amount of times each value in *ProductOps* feature is present in the dataset.

B.1.5 UtilizadorOps

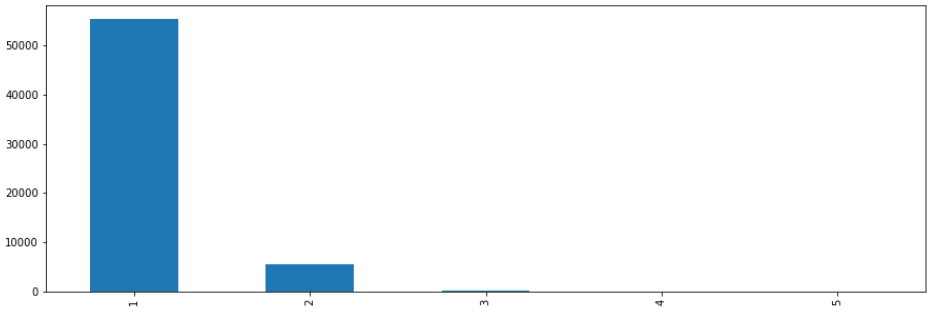

Figure B.5: An histogram showing the amount of times each value in *UtilizadorOps* feature is present in the dataset.

B.1.6 SeriesUtiliz

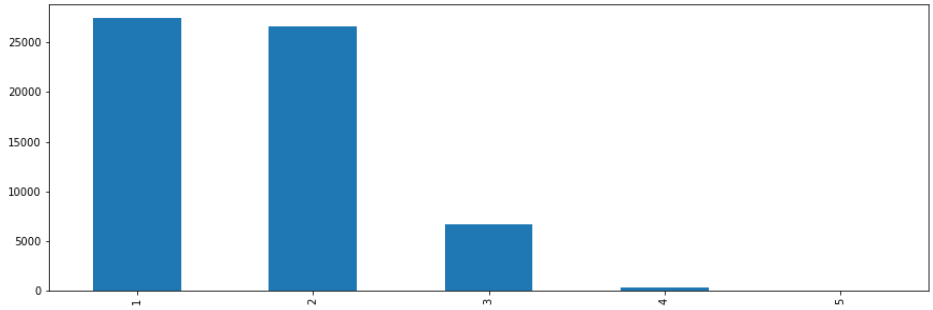

Figure B.6: An histogram showing the amount of times each value in *SeriesUtiliz* feature is present in the dataset.

B.1.7 UtilizadorDailyOps

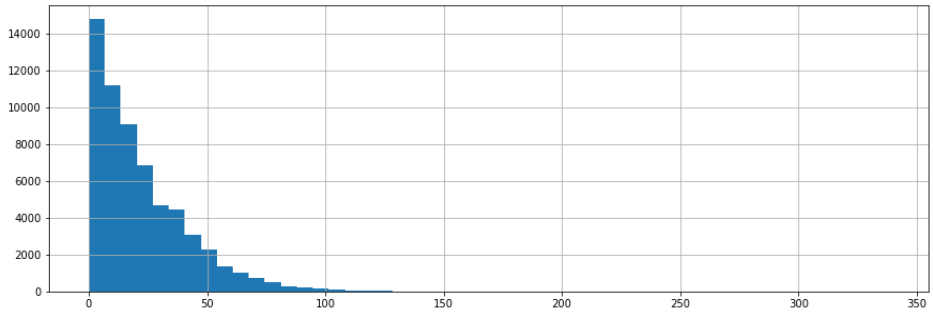

Figure B.7: A bar chart representing the distribution of *UtilizadorDailyOps* variable the dataset.

B.1.8 SeriesMeanVI

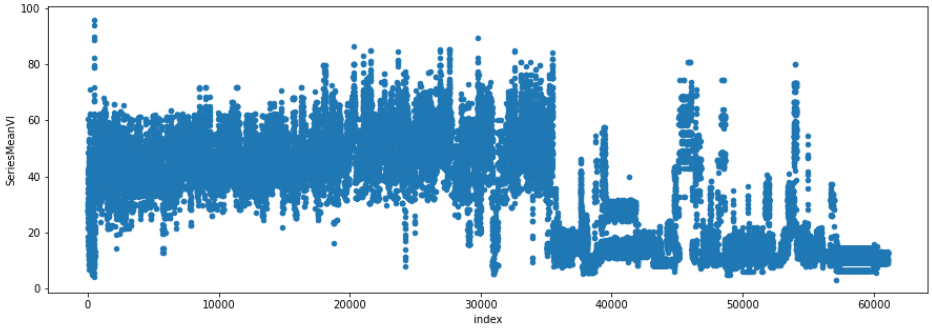

Figure B.8: A scatter plot representing the distribution of the *SeriesMeanVI* variable in the dataset.

B.1.9 SeriesNonConformities

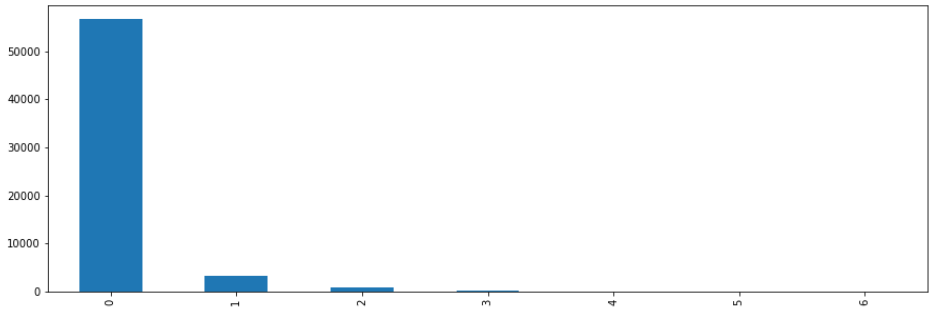

Figure B.9: A scatter plot representing the distribution of the *SeriesNonConformities* variable in the dataset.

B.2 Reports

B.2.1 Created Variables

	Variable Name	Nan#	Type	Values#	Mean	Std. Dev.	Median	Max	Min
0	SeriesDuration	0	float64	61064	7124.929	7276.228	5399.402	80997.97	0.0
1	SeriesOpChange	0	int64	61064	1.097	0.31	1.0	5.0	1.0
2	ProductCotas	0	float64	61064	3.515	2.023	3.0	19.0	1.0
3	ProductOps	0	float64	61064	1.036	0.187	1.0	2.0	1.0
4	UtilizadorOps	0	int64	61064	59.379	28.28	62.0	111.0	2.0
5	UtilizadorExperience	0	int64	61064	1211.844	1098.455	722.0	4159.0	0.0
6	SeriesUtiliz	0	int64	61064	1.671	0.688	2.0	5.0	1.0
7	UtilizadorDailyOps	0	int64	61064	22.272	20.442	17.0	338.0	0.0
8	SeriesMeanVl	0	float64	61064	34.875	17.848	38.493	96.173	3.01
9	SeriesNonConformities	0	int64	61064	0.095	0.382	0.0	6.0	0.0

Figure B.10: A report on the variables created during feature engineering.

References

[1] D.A. Adeniyi, Zune Wai, and Y. Yongquan. Automated web usage data mining and recommendation system using k-nearest neighbor (knn) classification method. *Applied Computing and Informatics*, 36, 10 2014.

[2] Ana Azevedo and Manuel Santos. Kdd, semma and crisp-dm: A parallel overview. In *IADIS European Conference on Data Mining*, pages 182–185, 01 2008.

[3] Paula Sofia Lourenço Barbosa. *Soft-sensor Development for Hydrocracker Product Quality Prediction*. PhD thesis, Instituto Superior Técnico da Universidade de Lisboa, Av. Rovisco Pais 1, 1049-001 Lisboa, 2014.

[4] Hetal Bhavsar and Amit Ganatra. A comparative study of training algorithms for supervised machine learning. *International Journal of Soft Computing and Engineering (IJSCE)*, 2, 01 2012.

[5] Avrim L. Blum and Pat Langley. Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 97(1):245–271, 1997. Relevance.

[6] Jason Brownlee. 4 types of classification tasks in machine learning. Available at https://machinelearningmastery.com/types-of-classification-in-machine-learning/, Accessed last time in February 2021, 2020.

[7] Kristen C. Malecki Burcu F. Darst and Corinne D. Engelman. Using recursive feature elimination in random forest to account for correlated variables in high dimensional data. *BMC Genetics*, 19, 2018.

[8] P. Chapman, J. Clinton, R. Kerber, T. Khabaza, T. Reinartz, C. R. Shearer, and R. Wirth. Step-by-step data mining guide. In *CRISP-DM 1.0*, 2000.

[9] Erdi Dasdemir. *Phase I Analysis of Variables Type Correlated Data*. PhD thesis, Hacettepe University, Ankara, Turkey, 07 2015.

[10] Zhenyun Deng, Xiaoshu Zhu, Debo Cheng, Ming Zong, and Shichao Zhang. Efficient knn classification algorithm for big data. *Neurocomputing*, 195:143–148, 2016. Learning for Medical Imaging.

[11] Stephen Doyle. Statistical process control key. Available at https://slideplayer.com/slide/7575703/, Accessed last time in January 2021, 2016.

[12] Stephan Dreiseitl and Lucila Ohno-Machado. Logistic regression and artificial neural network classification models: a methodology review. *Journal of Biomedical Informatics*, 35(5):352–359, 2002.

[13] Viviana Fernandez. Wavelet- and svm-based forecasts: An analysis of the u.s. metal and materials manufacturing industry. *Resources Policy*, 32(1):80–89, 2007.

[14] Pablo M. Granitto, Cesare Furlanello, Franco Biasioli, and Flavia Gasperi. Recursive feature elimination with random forest for ptr-ms analysis of agroindustrial products. *Chemometrics and Intelligent Laboratory Systems*, 83(2):83–90, 2006.

[15] Mikell P. Groover. Machining operations and machine tools. In *Fundamentals Of Modern Manufacturing*, page 507–551. John Wiley & Sons, 2010.

[16] RapidMiner Inc. Product quality prediction and optimization in steel manufacturing. Available at https://rapidminer.com/resource/quality-prediction-optimization-manufacturing/, Accessed last time in January 2021, 2018.

[17] M. J. Islam, Q. M. J. Wu, M. Ahmadi, and M. A. Sid-Ahmed. Investigating the performance of naive- bayes classifiers and k- nearest neighbor classifiers. In *2007 International Conference on Convergence Information Technology (ICCIT 2007)*, pages 1541–1546, 2007.

[18] Slobodan Stefanovic; Imre Kiss; Damjan Stanojevic; Nenad Janjic. Analysis of technological process of cutting logs using ishikawa diagram. *Acta Technica Corviniensis - Bulletin of Engineering*, 7:93–98, 2014.

[19] Alexandre Reis Graeml Jurandir Peinado. *Administração da produção*. Biblioteca do UnicenP, 2011.

[20] Hung-An Kao, Yan-Shou Hsieh, Cheng-Hui Chen, and Jay Lee. Quality prediction modeling for multistage manufacturing based on classification and association rule mining. *MATEC Web of Conferences*, 123:00029, 01 2017.

[21] Javed Khan, Jun S. Wei, Markus Ringnér, Lao H. Saal, Marc Ladanyi, Frank Westermann, Frank Berthold, Manfred Schwab, Cristina R. Antonescu, Carsten Peterson, and Paul S. Meltzer. Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. *GBM Annual Fall meeting Halle 2002*, 2002, 09 2002.

[22] Ron Kohavi, Neal J. Rothleder, and Evangelos Simoudis. Emerging trends in business analytics. *Commun. ACM*, 45(8):45–48, August 2002.

[23] Jundong Li, Kewei Cheng, Suhang Wang, Fred Morstatter, Robert P. Trevino, Jiliang Tang, and Huan Liu. Feature selection. *ACM Computing Surveys*, 50(6):1–45, Jan 2018.

[24] L. Luca, M. Pasăre, and A. Stăncioiu. Study to determine a new model of the ishikawa diagram for quality improvement. *Fiabilitate şi Durabilitate*, 1:249–254, 2017.

[25] Inc MachiningCloud. Introduction to milling tools and their application. Available at https://www.machiningcloud.com/wp-content/uploads/2016/05/MachiningCloud_SelectingTurningTools.pdf, Accessed last time in March 2021, 2016.

[26] Inc MachiningCloud. Introduction to selecting milling tools. Available at https://www.machiningcloud.com/wp-content/uploads/2016/05/MachiningCloud_SelectingTurningTools.pdf, Accessed last time in March 2021, 2016.

[27] Inc MachiningCloud. Introduction to selecting turning tools. Available at https://www.machiningcloud.com/wp-content/uploads/2016/05/MachiningCloud_SelectingTurningTools.pdf, Accessed last time in March 2021, 2016.

[28] Inc MachiningCloud. Introduction to turning tools and their application. Available at https://www.machiningcloud.com/wp-content/uploads/2016/05/MachiningCloud_SelectingTurningTools.pdf, Accessed last time in March 2021, 2016.

[29] Kjell Johnson Max Kuhn. *Applied Predictive Modeling*. Springer, 2013.

[30] mech4study admin. Milling machine: Parts and working. Available at https://www.mech4study.com/2016/05/milling-machine-parts-and-working.html, Accessed last time in March 2021, 2016.

[31] Pankaj Mishra. What is milling machine – operation, parts and types. Available at https://www.mechanicalbooster.com/2016/12/what-is-milling-machine-operation-parts-types.html#5_Manufacturing_or_Bed_Type_Milling_Machine, Accessed last time in March 2021, 2016.

[32] L. Monostori, A. Markus, H. Van Brussel, and E. Westkämpfer. Machine learning approaches to manufacturing. *CIRP Annals*, 45(2):675–712, 1996.

[33] Douglas C. Montgomery. *Introduction to Statistical Quality Control*. John Wiley & Sons, seventh edition, 2012.

[34] Masahiko Mori, Adam Hansel, and Makoto Fujishima. Machine tool. In Luc Laperrière and Gunther Reinhart, editors, *CIRP Encyclopedia of Production Engineering*, pages 792–801, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg.

[35] Sérgio Moro, Paulo Cortez, and Raul Laureano. Using data mining for bank direct marketing: An application of the crisp-dm methodology. In *Proceedings of the European Simulation and Modelling Conference*, 10 2011.

[36] Daryl Pregibon. Logistic regression analysis. In *The Annals of Statistic*, volume 9, pages 705–724, 1981.

[37] David G. Stork Richard O. Duda, Peter E. Hart. *Pattern Classification, 2nd Edition*. Wiley-Interscience, 2000.

[38] Irina Rish et al. An empirical study of the naive bayes classifier. In *IJCAI 2001 workshop on empirical methods in artificial intelligence*, volume 3, pages 41–46, 2001.

[39] Joanna Ronowicz, Markus Thommes, Peter Kleinebudde, and Jerzy Krysiński. A data mining approach to optimize pellets manufacturing process based on a decision tree algorithm. *European Journal of Pharmaceutical Sciences*, 73:44–48, 2015.

[40] Thomas A. Runkler. *Data Analytics: Models and Algorithms for Intelligent Data Analysis*. Vieweg+Teubner Verlag, third edition, 2012.

[41] Cosma Rohilla Shaliz. *Advanced Data Analysis from an Elementary Point of View (Chapter 12)*. Cambridge University Press, 2019.

[42] Zia Sialvi. Lathe machine and its mechanism. *Mechanics and Mechanical Engineering*, 06 2020.

[43] A. Singh, N. Thakur, and A. Sharma. A review of supervised machine learning algorithms. In *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*, pages 1310–1315, 2016.

[44] Abdulhamit Subasi and Ergun Erçelebi. Classification of eeg signals using neural network and logistic regression. *Computer Methods and Programs in Biomedicine*, 78(2):87–99, 2005.

[45] S. Tyagi. Using data analytics for greater profits. *Journal of Business Strateg*, 24(3):12–14, 2003.

[46] Sholom M. Weiss, Amit Dhurandhar, and Robert J. Baseman. Improving quality control by early prediction of manufacturing outcomes. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, page 1258–1266. Association for Computing Machinery, 2013.

[47] R. Wirth and Jochen Hipp. Crisp-dm: Towards a standard process model for data mining. *Proceedings of the 4th International Conference on the Practical Applications of Knowledge Discovery and Data Mining*, 01 2000.

[48] Kam Cheong Wong. Using an ishikawa diagram as a tool to assist memory and retrieval of relevant medical cases from the medical literature. *Journal of Medical Case Reports*, 5, 03 2011.

[49] Daniela XHEMALI, Christopher J. HINDE, and Roger G. STONE. Naive bayes vs. decision trees vs. neural networks in the classification of training web pages, September 2009.

[50] Ke Yan and David Zhang. Feature selection and analysis on correlated gas sensor data with recursive feature elimination. *Sensors and Actuators B: Chemical*, 212:353–363, 2015.

[51] Wenjing Zhao, Liqun Shang, and Jinfan Sun. Power quality disturbance classification based on time-frequency domain multi-feature and decision tree. *Protection and Control of Modern Power Systems*, 4:27, 12 2019.