Faculty of Sciences and Technology

Department of Informatics Engineering

Diagnosis and Prognosis of Aircraft Systems State

Maria Inês Ferreira Fonseca

Dissertation in the context of the Master in Informatics Engineering, Specialization in Intelligent Systems advised by Prof. Alberto Cardoso and co-advised by Prof. Bernardete Ribeiro and presented to the
Faculty of Sciences and Technology / Department of Informatics Engineering

September 2020

UNIVERSIDADE D
COIMBRA

This page is intentionally left blank.

Acknowledgements

I would like to thank my family and my friends that supported me over the last year that I was involved in this investigation.

Also, I am grateful to my advisor Alberto Cardoso and co-advisor Bernardete Ribeiro for all the support and feedback that allowed me to improve my thesis as well as, personal knowledge. Furthermore, I am thankful for being part of the ReMAP team, more specifically, for all the opinions, observations, and commentaries, in the weekly meetings, that help me to develop my work.

This research is supported by European Union's Horizon 2020 programme under the ReMAP projet, grant No 769288.

This page is intentionally left blank.

Abstract

With the evolution of technology there is a need to improve the world, taking advantage of it. And the aviation industry is no exception. As time went by, the need arose to extend the useful life of a system or component, through an analysis based on its health state (CBM). With the implementation of this type of methodologies it is also possible to predict when it will fail, calculating the life left until this event. This is one of the approaches that needs to be improved, once the timely detection of this type of anomalies allows airlines to save money, as well as an adequate maintenance management by the teams responsible for them.

In order to respond to these needs, the literature explains some approaches made in this regard, highlighting the PHM methodologies. Through these, an analysis is made of the behavior of the sensors, thus reflecting the health status of the respective systems, as well as the corresponding degradation, making it possible to predict the occurrence of a failure. In this way, most of the approaches found analyze the data of the sensors present in the system, through the application of Machine Learning algorithms, Kalman Filters, among others, which allow observing their behavior showing a possible degradation of the system.

The work done throughout this investigation resulted in the adoption of two different approaches, benefiting from the data for the two phases of this work. For the diagnostic phase, a methodology based on supervised Machine Learning algorithms was developed and, for the prognosis phase, a method was developed based on the interpretation of the behavior of the sensor signals.

Both approaches were applied, as proof of concept, for the systems Air Bleed and CACTCS that belong to Boeing 747 and Boeing 787, respectively. Despite they belong to different aircraft, the purpose of them is equivalent - extract the air from the outside of the airplane and take it to the places where it is needed.

The obtained results, for these systems, are promising. Regardless exist some differences between the systems, they, in general, predict with some security the health of the systems and the occurrence of a failure. This way, these results could have a huge impact because they permit to extend the life of a system, helping the schedule in the maintenance team. However, these results need to be validated by the airline companies involved in the ReMAP project.

Keywords

Aircraft Maintenance, Artificial Intelligence, Condition-Based Maintenance, Diagnosis, Machine Learning, Predictive and Health Management, Prognosis

This page is intentionally left blank.

Resumo

Com a evolução da tecnologia existe uma necessidade de melhorar o mundo, tirando partido dela. E a indústria aeronáutica não é exceção. Com o evoluir dos tempos, surgiu a necessidade de estender o tempo de vida útil de um sistema ou componente, através de uma análise baseada no estado de saúde da mesma (CBM). Com a implementação deste tipo de metodologias é possível também prever quando é que o mesmo irá falhar, calculando o tempo de vida restante até este acontecimento. Esta é uma das abordagens que é necessário melhorar, uma vez que a deteção atempada deste tipo de anomalias permite poupar dinheiro às companhias aéreas, bem como uma gestão adequada de manutenções por parte das equipas responsáveis pelas mesmas.

Com o intuito de dar resposta a estas necessidades, a literatura explica algumas abordagens feitas nesse sentido, destacando-se as metodologias PHM. Através destas é feita uma análise ao comportamento dos sensores refletindo, assim, o estado de saúde dos respetivos sistemas, bem como a correspondente degradação, possibilitando a previsão da ocorrência de uma falha. Deste modo, a maioria das abordagens encontradas analisam os dados dos sensores presentes no sistema, através da aplicação de algoritmos de Machine Learning, Filtros de Kalman, entre outros, que permitem observar o seu comportamento evidenciando uma possível degradação do sistema.

O trabalho realizado ao longo desta investigação resultou na adoção de duas abordagens distintas, beneficiando dos dados para as duas fases deste trabalho. Para a fase de diagnóstico, foi desenvolvida uma metodologia baseada em algoritmos de Machine Learning supervisionados e, para a fase de prognóstico, foi desenvolvido um método baseado na interpretação do comportamento dos sinais dos sensores

Ambas as abordagens foram aplicadas, como prova de conceito, nos sistemas Air Bleed e CACTCS, que pertencem ao Boeing 747 e ao Boeing 787, respetivamente. Apesar de os sistemas fazerem parte de aviões diferentes, o objetivo de ambos é o mesmo - extrair o ar do exterior e conduzi-lo adequadamente até aos locais onde ele é necessário.

Os resultados obtidos para estes sistemas são promissores. Apesar de existirem algumas diferenças entre os sistemas, em geral, é prevista com alguma segurança o estado de saúde dos mesmos e a respetiva ocorrências de falhas. Deste modo, estes resultados poderão ter um enorme impacto, pois através da antecipação de possíveis falhas, poderão ser evitados acidentes ou despesas avultadas para resolver os problemas, permitindo prever o estado dos componentes e dos sistemas de forma a promover processos eficazes de manutenção. No entanto, estes resultados necessitam de ser validados pelas companhias aéreas envolvidas no projeto ReMAP.

Palavras-Chave

Diagnósitco, Inteligência Artificial, Manutenção Baseada na Condição, Manutenção de Aviões, Machine Learning, Prognóstico, Prognósticos e Gestão de Saúde

This page is intentionally left blank.

Contents

Acronyms

ANN Artificial Neural Networks. 20

ARMA Auto-Regressive Moving Average. 19

CAC Cabin Air Compressor. 38

CACTCS Cabin Air Conditioning and Temperature Control System. 4, 37, 38, 41, 47

CBM Condition Based Maintenance. v, 2, 4

DFT Discrete Fourier Transform. 4, 13, 14, 30, 44, 50

DWNN Dynamic Wavelet Neural Network. 20

FDE Flight Deck Event. 7, 8

FDI Fault Detection and Isolation. 18, 19

FFT Fast Fourier Transformation. 14

FT Fourier Transform. 13

HI Health Indicator. 5, 7–10, 17–21, 24, 27, 28, 36, 37, 39, 42, 43, 47–49, 53, 55

ISHM Integrated Systems Health Management. 22

KF Kalman Filter. 23, 24

MSE Mean Squared Error. xiii–xvi, 5, 14, 22, 28, 29, 42–44, 47–50, 65–73, 75–84

NN Neural Networks. 20, 23

PHM Prognostics and Health Management. v, 17

RBF Radial Basis Function. 12, 29, 41, 43, 44, 47

ReMAP Real-time Condition-based Maintenance for Adaptative Aircraft Maintenance Planning. 1, 3, 4, 55, 56

RMS Root Mean Square. 21

RUL Remaining Useful Life. 4, 5, 7, 8, 17, 20, 22–25, 30, 55, 56

SVM Support Vector Machines. 11, 12, 21, 24, 27

SVR Support Vector Regressor. 12, 28, 41, 43, 44, 47, 48

WNN Wavelet Neural Networks. 20

This page is intentionally left blank.

List of Figures

List of Tables

Chapter 1

Introduction

Diagnosis and Prognosis of Aircraft Systems State is the thesis' theme that was developed during the last academic year. It is inserted in a H2020 European project called Real-time Condition-based Maintenance for Adaptative Aircraft Maintenance Planning (ReMAP) [1] and also in the Master's in Informatics Engineering of the Faculty of Sciences and Technology of the University of Coimbra.

This introductory Chapter includes a brief explanation about the main goals, the context of the problem, as well as, the approach implemented, and the obtained results. With these descriptions a general idea is given about the document content and the respective structure.

1.1 Context and Problem Statement

Nowadays, people make predictions for organizing and planning their lives. These predictions could have an impact that takes the people to spend more or less money, according to the type of forecast made, depending of having a huge or tiny impact on their life. Progressively, the tools that allow making the predictions are more trustworthy, which can help people in their everyday life. They can be applied in different contexts in the life routines either for in perspective personal, for example, to choose the clothes to wear according to the wheater, or in a perspective professional, like good scheduling for the mechanical maintenance for aircraft.

Focusing on mechanical maintenance prediction and, more specifically, in an airline company, these predictions can save money above all when they are related to a system or component failure.

Suppose that a fault occurred, but it was not detected. It is possible that, in future events, this fault can affect other systems or components in the aircraft. But, what would happen if the defect was detected in time? Probably, the damage caused would be less and the costs associated, too.

A statistical study reports some facts about the importance of the predictions in the maintenance of the aircraft and how the prompted detection helps the airline companies save money and extend the lives of the systems of the aircraft [2]. Besides that, the maintenance team has benefits when a failure is detected timely like greater ease to structure the schedules or to organize the inventory, to allow quick maintenance for the airplane [3].

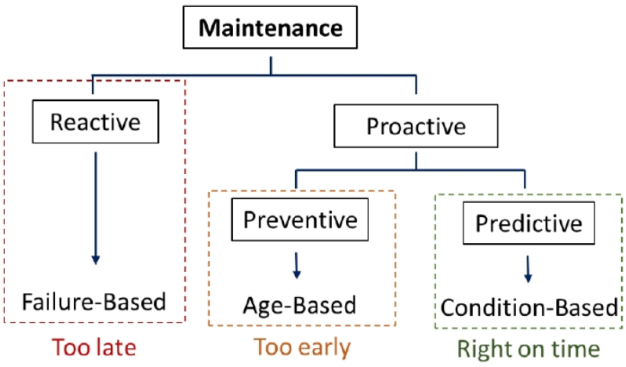

Figure 1.1: Difference between the reactive, preventive and predictive maintenance, extracted from [3].

In Figure 1.1 is shown the difference between the reactive, preventive and predictive maintenance, explaining which one has more (red group), medium (orange group) or less (green group) impact to the aircraft maintenance. Reactive maintenance occurs when a system or component fails and needs to be removed, switching by another identical. The second one, the preventive maintenance, establishes a relation between the age/flight hours and the appearance, knowing that a failure will occur in the next flights. Finally, predictive maintenance consists of finding out the condition of the system at a certain time, concluding when it is necessary to submit the system to the maintenance team. This approach will have less impact on money and the time spent [3]. The work developed falls on predictive maintenance, more specifically, in Condition Based Maintenance (CBM).

CBM helps to predict when a failure will occur based on the condition of the system through the sensor measurements along the time and the respective behavior, providing the diagnosis of the system state.

Nevertheless, even though the number of accidents is minor, they still happen, according to a report that describes aviation accidents [4]. In Figure 1.2 is shown a general decline in accidents, from 1998 until 2017, categorized by *RE* (green line), *SCF* (light brown line), *LOC-I* (dark brown line), *ARC* (yellow line), and *CFIT* (blue line), that are described after the Figure 1.3. This decline could be a result of the knowledge about all systems that compose the airplane, as well as, the technology was improved in this matter.

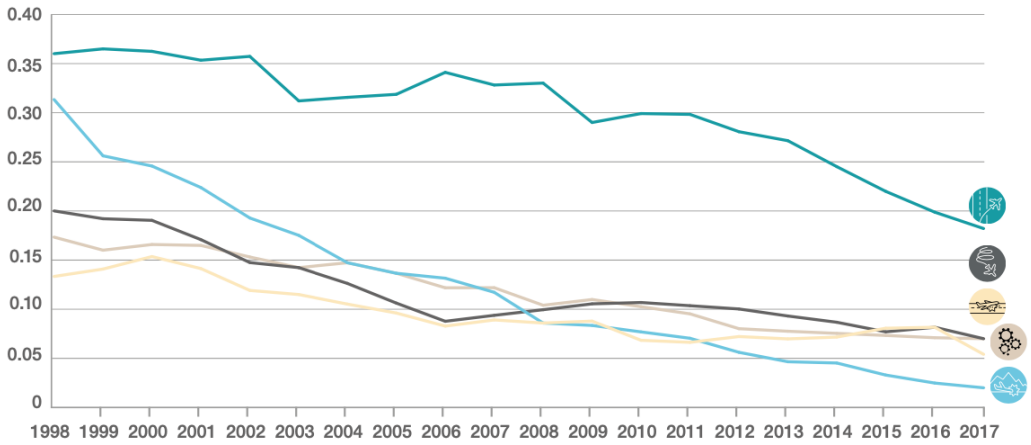

Figure 1.2: Hull loss rate per million flights, by accident category, extracted from [4].

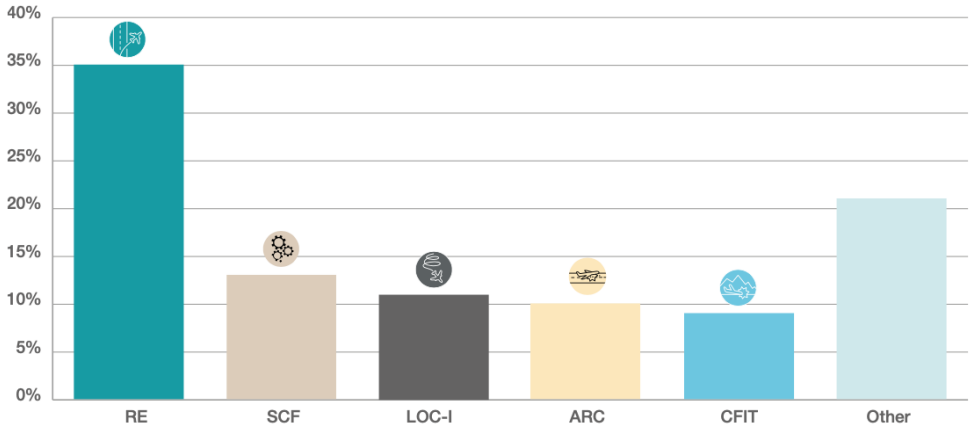

Figure 1.3: Percentage of hull losses accidents by accident category, extracted from [4].

Where each acronym, that corresponds to an accident category, means [4]:

- **RE:** Runaway Execution, which means a lane deviation;
- **SCF:** System/Component Failure or Malfunction, which means that occurred a failure in a system;
- **LOC-I:** Loss of Control in Flight;
- **ARC:** Abnormal Runway Contact, which means that the landing was not normal;
- **CFIT:** Controlled Flight Into Terrain, which means that the accident happened because of a collision with an obstacle (for example);
- **Other:** The accident was caused by a circumstance that is not labeled.

Analyzing the graphic in Figure 1.3, the main cause of their happening is related to the loss of control in flight, and the second one is associated with the no detection of the failure, on time. Despite they are not full losses, they are still losses that are associated with expensive costs to the airline, which results in a problem for them either for the money or time spent on that airplane.

In order to decrease the number of these values, and improve the statistical, is essential to help the maintenance team to find out what is the condition of the system and the possibility to occur a failure, to mitigating the losses caused by no attempted detection.

1.2 Goals

This thesis is integrated on the ReMAP Project, more precisely in Work Package 5 (WP5), which is responsible for developing methodologies for aircraft systems based on prognostics and health management. As such, the main goals of this thesis derive from the WP5 goals.

This work consists mainly of analyzing the health of a system/component (diagnosis) and predict when a failure will occur (prognosis) through the sensor data behavior. In order to obtain these results, there are a few steps that are needed to be processed:

1. **Processing the data:** The outliers were removed, using a sliding window that analysis which data are out of the established boundaries, and attribute a value that is inside of them;

2. **Diagnosis methodology:** The health of the system is predicted, feeding Machine Learning algorithms with sensor data, and predicting the Health Indicator for every 10 minutes;

3. **Prognosis methodology:** The behavior of the sensor data is analysed, in the frequency domain, using DFT;

4. The points **2.** and **3.** are applied in different systems, in order to test and validate these approaches.

More specifically, to have better results, the data is processed, whereby the outliers need to be removed. Then, the data will be used for the diagnosis and prognosis phase that is for the prediction of the health system and the Remaining Useful Life, respectively. After that, the results are analyzed and, to validate the approach, it is applied to another system. After these points are completed, the objective of this thesis is completed. This way, all of these processes will be explained in the next sections.

1.3 Approach

Condition Based Maintenance is the base concept that allows to correspond to the main objectives of this work is involved. The adopted approach takes into account the sensors' measurement over the time and how it can be useful to solve two main issues related to the diagnosis and prognosis.

The diagnosis phase consists in finding out the condition of the system and conclude if it suffered extra degradation. Relatively to prognosis, this phase aims to know how much time left (from the present) until the end of life. In this phase, is important to predict when a failure will occur once the early prediction allows the airline company to avoid expensive costs. So, the adopted approaches for these phases are:

- **Diagnosis:** Machine Learning algorithms were used to develop this phase, in order to predict the state of a system, based on degradation over the time.

- **Prognosis:** The prognosis phase followed an approach based on the frequency and magnitude analysis, with the main objective of anticipating the failure.

In the context of ReMAP, 13 different systems are considered. However, in this work only 2 of them will be treated: Air Bleed system, from Boeing 747, and Cabin Air Conditioning and Temperature Control System, from Boeing 787.

The obtained results in this investigation help to increase the life of a system and decrease the spent money when a failure is detected on time. However, it does not mean that a system is not changed if a fault does not occur. All systems have an established deadline, by the supplier, and when it is time to change, it will be exchanged for another, even if there is no fault. So, this work complements the work of the maintenance team to help to identify possible unusual behaviors and the respective origin, and it will not substitute the defined deadlines.

1.4 Main Achievements

According to the mentioned approaches, there were reached results in both phases.

In diagnosis, was developed a data-driven approach that is based on the health condition, using Machine Learning algorithms. This phase, after testing a set of test cases, and manipulate the data, was found the minimum MSE between the real and predicted health, which means that in most situations the HI predicted is equal to the HI in the target.

In the prognosis phase, the failure was anticipated through three different approaches that analyzed the magnitude, after each flight, in each sensor, over each trajectory.

1.5 Document Structure

Beyond this introductory chapter, this document has six more chapters.

In **Chapter 2** is the Background. There will be explained the important concepts that will be useful to understand the work performed.

In **Chapter 3** is the State of the Art. Here is described and analyzed some performed work related to the computation of the Health Indicator and the Remaining Useful Life, in similar problems, and that could be adapted to this situation.

Chapter 4 explains the approach applied, whether in diagnosis or in prognosis.

In **Chapter 5** is explained how Air Bleed system and CACTCS work and what are the relevant information in the dataset that will be used to obtain the results.

In **Chapter 6** is shown and explained the obtained results.

Finally, the conclusion of this work is described in **Chapter 7**.

This page is intentionally left blank.

Chapter 2

Background

There exist some concepts and a set of themes that are important to support the understanding of the next sections. In order to explain all the process clearly, over the document, these concepts and methods used in this work will be described.

2.1 Main Concepts

Before explaining the methodologies used, there are specific concepts that are essential to understand the adopted approach and help to have a clearly idea about it.

2.1.1 Health Indicator

The Health Indicator (HI) of a given system reflects the combination of flight hours and the deterioration process that increases flight after flight, resulting in a state of health [5].

The unit of the HI can be expressed in percentage [6] or an absolute value [7]. Over this work, the absolute value will be considered, and in ideal conditions, after one hour of flight, the HI value will be increased one hour. But in most cases, this doesn't happen because the systems have more deterioration in relation to the flight hours to which they have been subjected.

2.1.2 Flight Deck Events

A Flight Deck Event (FDE) is a warning that advises the maintenance team that something happened to the system. These warnings are classified depending on their severity.

The FDE is crucial because allows the maintenance team to find out what is wrong and fix the problem to extend the life of the system.

Over this document, a FDE can be named by *failure*.

2.1.3 Remaining Useful Life

The Remaining Useful Life is the time left (from the present) until the end of life of a certain system [8]. The evolution of this concept over the time is shown in Figure 2.1.

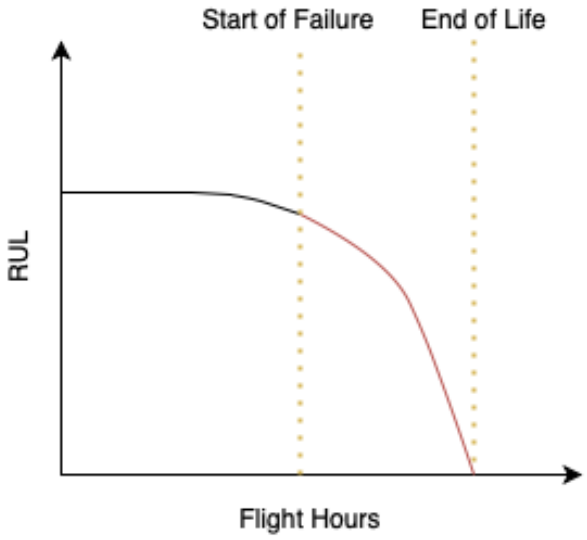

Figure 2.1: RUL evolution, over the time.

In ideal conditions, the RUL is expected to have, after a certain period, a more pronounced decrease, which will indicate that the end of life is near, as shown by the red line in Figure 2.1. This value can be determined in a direct (when a pattern is recognized in the data that could help to estimate the remaining life) or indirect way (where first is estimated the damage, and then it will be compared with the real damage) [9].

However, the available data do not permit to compute this value simulating a real scenario, where the future behavior is not known. So, the approach followed focuses on the prediction of the failure instead of the computation of the RUL.

2.1.4 Removals

The removal matches the end of a trajectory, as well as, the maximum value of the Health Indicator and the minimum value (zero) of the RUL. In order words, the removal corresponds to the extraction of a certain system, when it is at the end of life.

They occur by routine because there is a maximum time, established by the maintenance team, that a system can be a part of the airplane. So, after that, it needs to be removed and substituted by another. However, other situations are responsible for the removals. These situations can be caused by the damage that takes a high degradation of the system or a FDE occurrence that, when the warning is dangerous, jeopardizes the security.

2.1.5 Trajectory

A trajectory is the time interval since a system is inserted in the airplane until it is removed [10]. That way, the end of a trajectory matches with a removal, as well as, the value 0 of the Remaining Useful Life.

2.1.6 Flight Phase and Aggregated Phase

The flight phase is a number (between 1 and 14) that corresponds to a stage of the flight. The number and the respective phase is described below:

1. Power on
2. Engine start
3. Taxi out
4. Unknown
5. Take-off roll
6. Initial climb
7. Climb
8. Cruise
9. Descent
10. Approach
11. Rollout
12. Taxi in
13. Unknown
14. Engine shutdown

Some of these fourteen phases have a similar sensor behavior over time, which means that these flight phases can be processed at the same time. The result of these combinations of the flight phases are designated by aggregated phases and are described in Table 2.1.

Table 2.1: Aggregated phase explanation.

Number of the Aggregated Phase	**Name of the Aggregated Phase**	**Phase Flight**
1	Start	Phase 1
2	Climb	Phase 2, 3, 4, 5, 6, and 7
3	Cruise	Phase 8
4	Descent	Phase 9, 10, 11, 12, and 13
5	Finish	Phase 14

Analyzing the table were assumed five different aggregated phrases that represent the 14 flight phases and, consequently, each flight.

2.2 Machine Learning Algorithms

To predict the HI, in the diagnosis phase, three Machine Learning algorithms (Linear Regression, Support Vector Machines, and Random Forest) were used.

First of all, the Machine Learning algorithms are divided into three main groups [11] [12]:

- **Supervised:** Each instance of the data has a label, and a label is like a tag that will help the training phase of the algorithm, helping to get a correct solution in the testing phase. So, the algorithm result is computed based on data patterns that were associated with the respective labels;
- **Unsupervised:** The algorithm does not need labels. So, it finds the pattern in the data by its own. These methods are used in problems that, in general, are more complex than those that use supervised algorithms;
- **Reinforcement Learning:** The algorithm learns to react to the environment through the feedback given for the right choices. These methods are not labeled, like the supervised algorithms, whereby the decision for a given task is not previously defined.

Moreover, it is possible to divide the problem as follows [13]:

- **Classification:** The model divides the data into labeled classes;
- **Regression:** The model predict continuous values, through the input data, instead of using classes.

So, the algorithms used to perform the diagnosis phase are supervised, once the HI for each instance is defined. As the HI is a real value, that starts in 0 and increases over time, it is a regression problem. Relatively to the prognosis phase, there were not used Machine Learning algorithms. The approach is based on the analysis of behavior of the provided sensor data.

2.2.1 Linear Regression

The technique that allows finding a linear relationship between x and y, which means, between input and output, respectively, is named as Linear Regression. This Machine Learning Algorithm is characterized by the equation explained below [14]:

$$y = \beta_0 + \beta_1 x + \varepsilon \tag{2.1}$$

Where each variable means:

- y**:** Is the dependent variable. In other words, it is the output from the Linear Regression. This variable returns the value that is intended to be predicted;
- β_0**:** Is the value where the line intersects the Y-axis;
- β_1**:** Is the slope of the line;
- x**:** Is the independent variable, that is, the value of the data point;
- ε**:** Is the noise that is generated by unexplained factors.

The Equation 2.1 can be explained, more detailed, in Figure 2.2, where the relationship between all variables is shown.

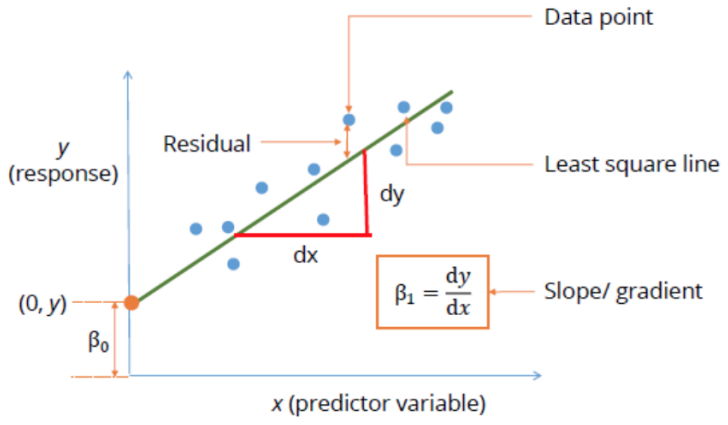

Figure 2.2: Example of how the Linear Regression algorithm works, extracted from [15].

Looking at Figure 2.2, the variables of Equation 2.1 and the data that are analyzed (blue dots) are represented. The main goal of the Linear Regression algorithm is to draw a line between the data that will minimize the error. This line (green line, in Figure 2.2) is mostly defined by the ordinate at the origin, the slope, and the error. This way, β_0 represents the point where the green line intersects the Y-axis. Moreover β_1 is the slope, that is obtained by Equation 2.2, considering the points A = (x_1, y_1) and B = (x_2, y_2).

$$\beta_1 = \frac{d_y}{d_x} = \frac{y_2 - y_1}{x_2 - x_1} \tag{2.2}$$

The ε mentioned in Equation 2.1 is obtained through the sum of the distance of each blue dot to the green line. In short, in order to get a minor error, this algorithm has been trying to identify which is the best line.

2.2.2 Support Vector Machines

On the SVM algorithm the main goal is the drawn line, which is denominated hyperplane, that divides the data into classes in a dimensional space considered. So, this hyperplane separates the data classes allowing the best division between them. For example, considering that exist two classes (A and B), most elements of class A are on one side, and the ones of class B are mainly on the other side. In Figure 2.3, is shown an example of how SVM works.

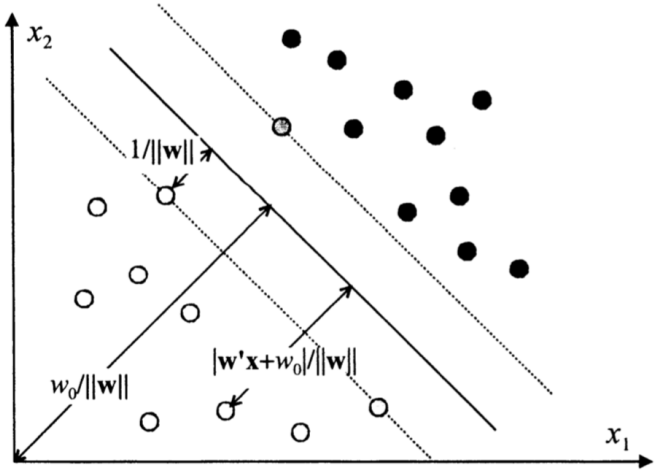

Figure 2.3: SVM example, extracted from [16].

In Figure 2.3 is shown the hyperplane that was drawn with the help of the SVM algorithm through the bold line. This line allows observing that the two classes considered were separated almost correctly. Also, in the same Figure is shown that the distance between the hyperplane and the closest data is given by the expression $\frac{1}{\|w\|}$. So, the Equation 2.3, where w is the vector of values that will be computed, that maximizes the separation, minimizing the distance, is given by:

$$min \frac{1}{2} \|w\|^2 \tag{2.3}$$

However, it is crucial to reduce the cost function, assuring that the weight vector is minimized, too. As this is a quadratic function, there exists a method that will solve this problem based on Lagrangian function, computing the saddle point, as is shown in Equation 2.4:

$$J(w, w_o, \alpha) = \frac{1}{2} \|w\|^2 - \sum_{i=1}^{n} \alpha_i (t_i (w' x_i + w_0) - 1) \tag{2.4}$$

Where w is the vector of values that will be computed, α_i is the Lagrange Multipliers, t_i is the target values, x_i is the vector that correspond to the distance between the data and the hyperplane, and b is the bias defined.

In order to improve the results that could be obtained, it is crucial to change the way the SVM will separate the data. Thus, it is possible to change the way the algorithm does that, changing the functions that are responsible for that, which are below described:

- **Linear:** $K(x_i, x_j) =< x_i, x_j >$;
- **Polynomial:** $K(x_i, x_j) = (\gamma < x_i, x_j > +r)^b$;
- **Radial Basis Function (RBF):** $K(x_i, x_j) = e^{(-\gamma \|x_i - x_j\|^2)}$;
- **Sigmoid:** $K(x_i, x_j) = tanh(\gamma < x_i, x_j > +r)$.

For some of those algorithms, it is feasible to change two values: γ and r. The γ value is responsible for defining how much influence the training examples have. So, how lower is γ's value, more close should be the examples. The r value is a kernel parameter which represents the interception with the y axis.

This algorithm can be applied to the regression problems [17], with Support Vector Regressor (SVR). The difference between the regression and the classification problems had been mainly in the creation of the margin of tolerance (ϵ), shown in figure 2.4, as a way to approximate the SVR to SVM. The remaining algorithm works in the same way, as explained before.

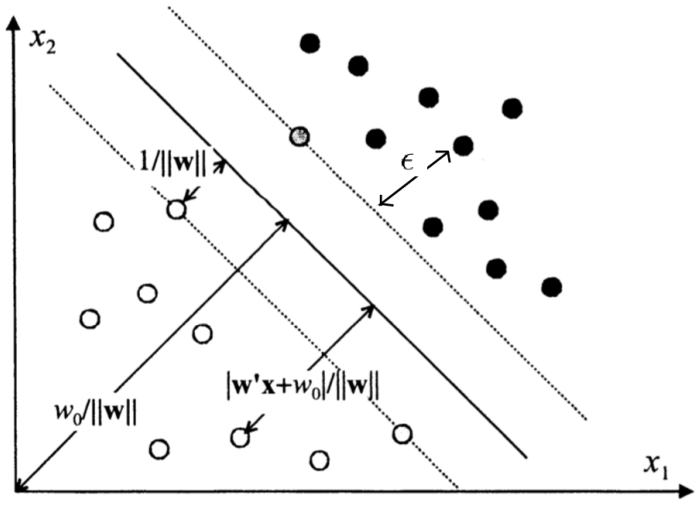

Figure 2.4: SVR example, adapted from [16].

2.2.3 Random Forest

Random Forest is an algorithm that works with decision trees. A decision tree draws all the possible paths with feasible solutions, based on the data features [18] [19]. In Figure 2.5 is shown an example of how is created one tree, using this algorithm.

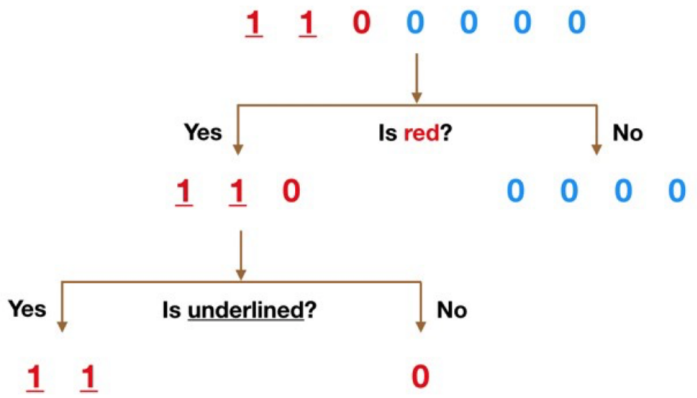

Figure 2.5: Creation of trees with the Random Forest algorithm, extracted from [18].

As demonstrated in Figure 2.5, there are two features in the considered data. The color (that could be red or blue) and the underline (if the considered data is, or not, underlined) are the features that will be analyzed.

First of all, a branch was created, and the target was divided into the colors. So, it was built two branches: one to the red color and another one to the blue color. Then, the two forks that were created are considered, and, for each one, it will be checked if the elements are underlined or not. This way, on the left side, it was created another branch, and on the right side, it wasn't created another branch, because this feature is not presented. The algorithm is stopped here because there are no more features.

After concluding this process, some individual trees will be created and will be operated as an ensemble, once they will be submitted to a voting process, that will decide which tree is the best for the problem/data.

Finally, there is an important topic that distinguishes the difference between the classification and the regression problems on the voting process, with the Linear Regression algorithm. To the classification problem, the voting process is computed using the mode, considering the result for each tree. To the regression problem, the voting process was computed using the mean, that was obtained for all of the output of the trees created [20].

2.3 Discrete Fourier Transform

The Discrete Fourier Transform (DFT) is a mathematical method that is applied in the signal frequency domain [21] and is derived from the Fourier Transform (FT), which is given through Equation 2.5.

$$DFT(w_k) = \sum_{n=0}^{N-1} x(t_n) e^{-iw_k t_n}, \quad k = 0, ..., N-1 \tag{2.5}$$

Where each variable means [21]:

- w_k: Is the k^{th} frequence sample
- N: Number of samples
- t_n: Is the n^{th} sampling instant

The DFT is calculated with the help of the Fast Fourier Transformation (FFT) that allows computing the frequency components simply and efficiently, relatively to other approaches [22] [23].

So, the FFT receives as input the signal, and then it is processed to obtain the respective frequencies, which range starts in 0 until maximum frequency (less than half of the sampling rate) [23].

2.4 Normalization

A way to have better performance with Machine Learning algorithms is the normalization of the data, either the sensor measurements or target. This normalization consists in transforming these values into a small interval, typically between 0 and 1, as is shown in Equation 2.6.

$$Normalization = \frac{X_{max} - X_{min}}{Y_{max} - Y_{min}} * (c_1 - Y_{max}) + X_{min} \tag{2.6}$$

After the normalization, and in order to obtain the original values, is mandatory to invert the normalization, through Equation 2.7.

$$Inv\_Normalization = \frac{Y_{max} - Y_{min}}{X_{max} - X_{min}} * (c_2 - X_{max}) + Y_{min} \tag{2.7}$$

In Equation 2.6, the x values belonged to the new interval that the values will be converted. In this case, X_{min} and X_{max} correspond to the interval between 0 and 1, respectively. The Y_{max} and Y_{min} are correlated to the maximum and minimum values, respectively, on the analyzed array. The element c_1 of Y will be converted into a number between X_{min} and X_{max}.

Relatively to the Equation 2.7, the X and Y means the same relative to Equation 2.6. However, the number c_2 is the analyzed element, and it will be converted into a number between Y_{min} and Y_{max}.

2.5 Mean Square Error

A measure that evaluates the Machine Learning Algorithms is the Mean Squared Error (MSE) and is expressed through Equation 2.8. This metric allows an understanding of how big the error is, between the ground truth and the obtained predicted values, once this distance is squared. This metric allows us to conclude that the greater the result, the

greater the error. For example, when a distance between two points is minimal (less or equal than 1), the distance will be continued minimal. Otherwise, if the distance between two points was significant, it mean that the squared result will be more significant.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(X_i - \widehat{X}_i)^2 \tag{2.8}$$

Where n is the number of the elements of the array, X_i is the ground truth and $\widehat{X}_i$ is the predicted array.

This page is intentionally left blank.

Chapter 3

State of the Art

In this section is described some methodologies based on Prognostics and Health Management (PHM) that allow computing the Health Indicator and the Remaining Useful Life, which will be useful for the improvement of this work, in order to help the maintenance team to estimate how long will take the system to fail.

So, first of all, in Section 3.1, is given a context about the different methods that could be used in PHM. Then, in Sections 2.1.1 and 3.3 are reported some methodologies that were used in other works. Furthermore, this section finishes with a summary that will give an idea about the approach that will be followed, considering the developed work.

3.1 Prognostics and Health Management

Prognostics and Health Management (PHM) help the maintenance team to find and solve some problems through algorithms that detect unusual behaviors. This detection can be helpful because permits to estimate the Remaining Useful Life (explained in Section 2.1.3), and the state of the health of a system, contributing to safety. In addition to these purposes, these methods could help to extend the lifetime of a system that will result in saving money to the airline company as well as, in an easy for the scheduling, for the maintenance team [24].

This way, there exist three different methods that can be used in PHM, which will be described in the next sections.

3.1.1 Model-based method

Model-based methods describe the degradation of a system through mathematical or physical models, doing the update of model parameters through the data that was measured. This degradation can be a reflection of high deterioration, which results in fewer hours for the end of life of a system. So it will help the maintenance team to fix some problems that have occurred, extending the life of a system [25].

Modeling using Physics-of-Failure

This method is applied when occurs a physical failure, like a crack, that will influence the normal condition and behavior of a system.

Marble and Morton et al. [26] defend that a physical failure can be disseminated, over time, with the use. They explain that this propagation can affect the condition of the correspondent system and even others, increasing the damage, as can be seen in Figure 3.1.

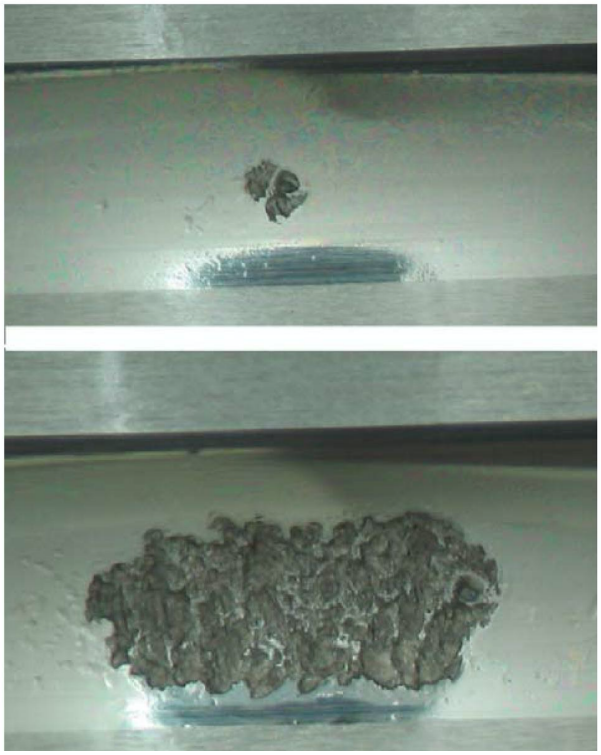

Figure 3.1: Example of a damage propagation, extracted from [26].

The photos shown in Figure 3.1, were taken at different times of the experience described in the mentioned article. The first one was taken earliest than the second, which means that over time the physical failure increased and could become major. Moreover, this failure could affect the performance of the own system as well as, the others that are depending on it.

Model-based Fault Detection and Isolation

Model-based Fault Detection and Isolation (FDI) is an approach whose decision is taken considering a mathematical model that is based on the physical condition.

Frank et al. [27] identified four types of FDI:

- **Parity Space:** Uses a model to simulate the output, based on input data. The fault is calculated based on the computed error between the prediction of the model and the physical state of the system;
- **Dedicated Observer:** Estimates the Health Indicator using the inputs and the outputs of the physical system.
- **Fault Detection Filter:** Is a model that identifies the failures, and isolates them, according to the behavior of the system's input-output.
- **Parameter Identification:** Identifies the parameters of the model, according to the input/output of the physical model.

Abdulhamed and Reza et al. [28] developed an approach based on HI. First of all, they developed a mathematical fault model and, consequently, they designed an observer-based sensor Fault Detection and Isolation. These models, according to the defined equations explained in their article, permit to read the sensor signal and identify where the fault occurs.

3.1.2 Data-driven method

Data-driven methods are methods that predict the system state, based on degradation behavior, obtained from time series. The degradation behavior is typically obtained from the sensors' measurements [8].

Once no priori knowledge is needed, these methods are used to detect a failure because they can establish a relation between the input and the output data and the respective degradation [29]. The following sections topics have some examples where this method can be applied.

ARMA

ARMA is the acronym for Auto-Regressive Moving Average, in which the main goal is to capture the entire history of fault events and predict the next event [30]. It can be useful when the goal is to predict failure, being necessary, previously, to train an algorithm, based on sensor measurements and the respective behavior.

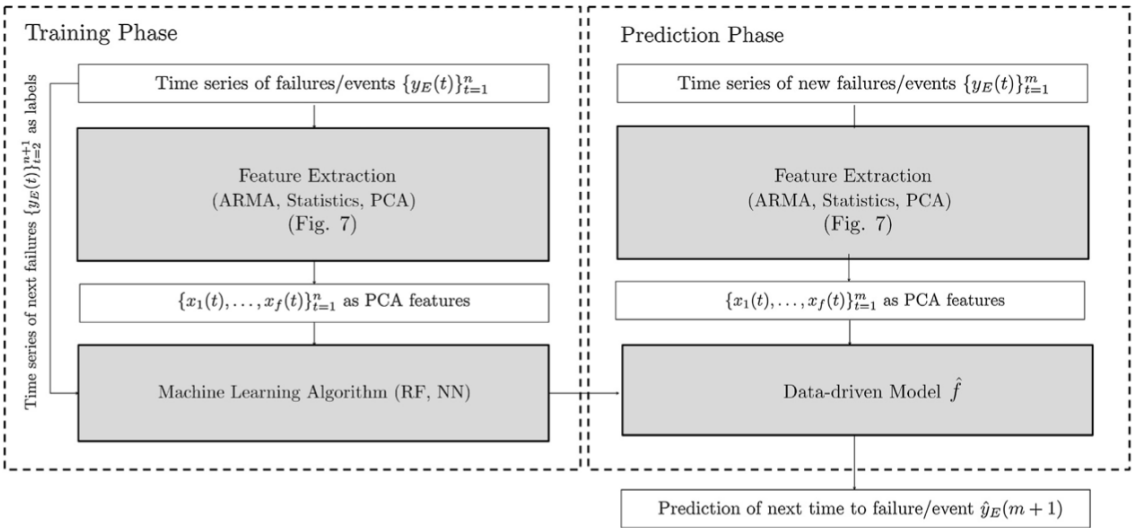

Figure 3.2: Workflow using ARMA model, extracted from [30].

Márcia et al. [30] developed an approach which is detailed in Figure 3.2. First of all, a model is created in the training phase, which will predict failure in the prediction phase. In this phase, the ARMA model is used to obtain the historical data of failures. Then, it will be used to train a Machine Learning Algorithm which target is the current time of the failure. Then, proceeds with the prediction phase, where was extracted the historical data too, and the output is the prediction of the next failure. Finally, it is created a data-driven model, which is fed with the historical data obtained from the ARMA and with the Machine Learning model created in the training phase. The result is the prediction of the next failure of the system.

Artificial Neural Networks

Artificial Neural Networks (ANN), which also are known by Neural Networks (NN), simulate a representation of the brain. These structures are composed of neurons that make connections between them, processing and transmitting the information [31]. In Figure 3.3 is shown an example of the representation of the Neural Networks.

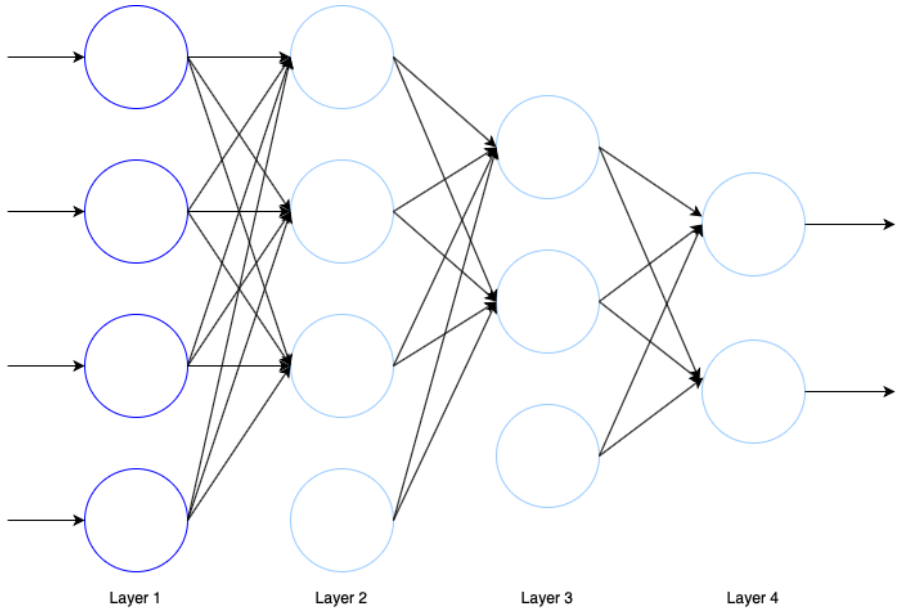

Figure 3.3: Example of a Neural Network.

Interpreting the Figure 3.3, the NN receives the input through Layer 1. This input could be an array or an image, for example, consonant the considered objective. Then, it is processed, based on some mathematical calculations, in the Layers 2 and 3, also called hidden layers, and the Layer 4 returns all the possible outputs.

Wang and Vachtsevanos et al. [32] used two variants of Neural Networks that, in general, could have successful results: the Wavelet Neural Networks (WNN) and the Dynamic Wavelet Neural Network (DWNN). The first step of their work consisted of analyzing and evaluating a physical failure, like a crack, using a WNN. Then, they used a DWNN to predict this failure propagation over time, as well as, to compute the RUL.

3.1.3 Hybrid method

The hybrid method joins the data-driven and model-based approach. More specifically, this method combines the sensor measurements with the state of a system, respectively. So, the results obtained with this method could be an improvement when compared just with the data-driven model, or just with the model-based approach, because the results of these two methods are compared, which help to avoid the false positives and compute a more precise Remaining Useful Life.

3.2 Health Indicator extraction

The Health Indicator is a measure that reflects the age of a system or a component (in flight hours), as was explained in Section 2.1.1. So, in literature, there are many ways to

express this value. As mentioned before, a typical way to compute it is by analyzing the sensor values and the respective behavior, for example. This way, it was identified three different methods that permit to compute the Health Indicator value, as will be described in the following sections.

3.2.1 Physical Health Indicator

The Physical Health Indicator describes the health of a system based on the actual state of the system and the occurred failures. These attributes are analyzed according to statistical or signal processing methods that, in general, will be interpreted typically with the Root Mean Square (RMS) metric. The extra degradation occurs when the obtained result from the RMS is higher than a threshold, defined previously by the system's specification [8] [33].

3.2.2 Probabilistic Health Indicator

The Probabilistic Health Indicator is a probability between 0 and 1 that reflects how healthy is a system, where the meaning of the boundaries depends on the context. Generally, 1 means that the system is healthy, and 0 means the opposite. However, sometimes is the opposite because of the system's properties, being necessary to specify this characteristic. Such as the Physical Health Indicator, the Probabilistic Health Indicator has a threshold, which expresses the degradation that is defined by a statistical confidence level. [8].

3.2.3 Virtual Health Indicator

The difference between Physical Health Indicator and Virtual Health Indicator (VHI) is related to physical degradation. The VHI is computed through the signal of the sensors and doesn't need the physical state, as was observed in Physical Health Indicator. To visualize the behavior of this method, a dimension reduction technique, like principal component analysis, is used [33].

3.2.4 HI Computation

The Health Indicator Computation can be done in different ways, using one of the methods mentioned above. However, the following topics will explain how the Health Indicator is computed, based on what is reported in the literature allied to the described concepts.

Artificial Intelligence

Another way to explore data-driven methods is to use Artificial Intelligence beyond the Neural Networks, as mentioned before. So, many algorithms can be applied for analyzing the sensor behavior, like Support Vector Machines, Random Forest, among others, and obtaining good results with them.

Marcia et al. [34] used an approach based on Artificial Intelligence. The methodology applied in her work consists of two main phases: the training and testing/predicting the failure. The first stage consists of training a Machine Learning Algorithm, with sensor data and which target is the fault events. In this phase, it is created a model that will be

used in the second stage. After that, the second phase starts. Here, the idea is to predict the next failure, analyzing the sensor data with the help of the created model. The results are analyzed considering specific metrics like Mean Squared Error.

Integrated Systems Health Management

The Integrated Systems Health Management (ISHM), with the provided data, information, and knowledge about the systems, can describe the capability that permits to compute the health condition [35]. These systems receive as input the sensor values and, based on their behavior, will help to identify an anomaly occurrence and fix it, to prevent other future failures [36]. The ISHM method is divided into four groups of faults diagnosis that, all together, allows to identify the problem:

- **Fault Detection:** Catching that something is wrong;
- **Fault Isolation:** Find out where the failure occurred;
- **Fault Identification:** Identify what happened to occur the failure;
- **Fault Prognostics:** Determining the usage of the system and anticipate the next failure.

ISHM, consonant to the data and the respective behavior, works with different algorithms. If the data belongs to a problem that uses the model-based approach, the algorithm that will be used with the ISHM is a Finite State Machine. Otherwise, if the data belongs to a problem that uses the data-driven approach, the algorithms that will be used could be the Linear Regressions or the Decision Trees.

Figueroa et al. [35] developed an architecture whose main goal is to develop a credible ISHM, that could be used in complex systems, for the industrial community.

3.3 Remaining Useful Life Computation

In literature, exist many ways to find the Remaining Useful Life. However, most of them are depending on the information of the entire trajectory. So, in the following topics are explained two methods that follow a data-based model, where the behavior of the data measurements is analyzed and interpreted, and there is not needed to know all trajectory behavior.

Elbow Point

The normal degradation behavior of a system consists of a linear and smooth degradation behavior until a certain time of flight hours/cycles, and after that, the deterioration is accentuated. The point from which the degradation is more emphasized is the elbow point, as is illustrated in Figure 3.4.

Elsevier et al. [37] uses an approach that permits to detect the change-point of the degradation of a system. The identification of this point is important because it is possible to anticipate a future failure that could be caused by the detected degradation that is

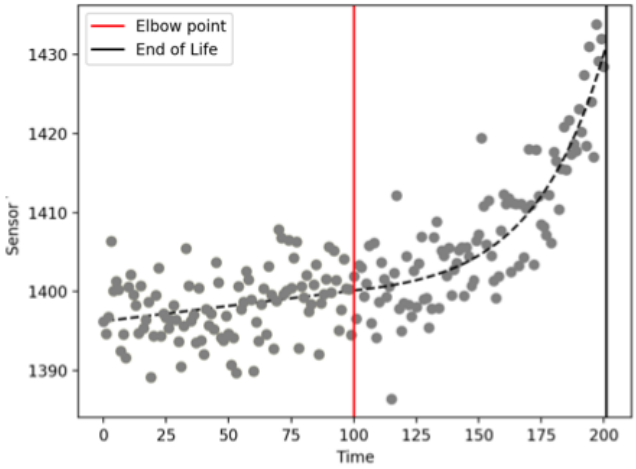

Figure 3.4: Example of RUL, extracted from [37].

reflected in the sensors measurements. The estimation of the Remaining Useful Life, from that point, is made based on the conditions that were detected.

This is a classification problem, where there are two different classes in the target: 0 and 1. So, first of all, the elbow point is detected using a sliding window with a Z-test. Then, the target that will be used is composed of 0 (until the elbow point detection) and 1 (from the elbow point until the end of life). After that, the sensor data are normalized according to the standard rule in Equation 3.1.

$$s(t)^{norm} = \frac{s(t) - \mu}{\sigma} \tag{3.1}$$

Where $s(t)^{norm}$ is the normalized signal resulted from the division of the subtraction of the $s(t)$, that is the signal at time t, and the μ, that is the mean of the signal, by the standard deviation of the values of the signal σ.

After that, a model that permits to detect the elbow point is created. This model is trained with the sensors data, and the ground truth is the obtained target, in the first step. Then, the acquired model (that was obtained through the training of a Neural Networks) is used to estimate the RUL, considering that the end of life (EoL) is known, through the Equation 3.2.

$$RUL(tp) = EOL - tp \tag{3.2}$$

Where tp is the time where the RUL is calculated, and the EOL is the end of life of the system, which is defined by a value that characterizes the limit of flight hours/cycles that a system could suffer.

Combining Data-Driven and Kalman Filtering

The Kalman Filter (KF) is an algorithm that estimates the measurements from the noisy time series [38].

Márcia et al. [39] compares the computing of the RUL using this method with Machine Learning algorithms (like the Support Vector Machines or Random Forest) and without this method. In this work was created a framework that transforms each RUL observed into a RUL KF-Estimation, as is shown in Figure 3.5.

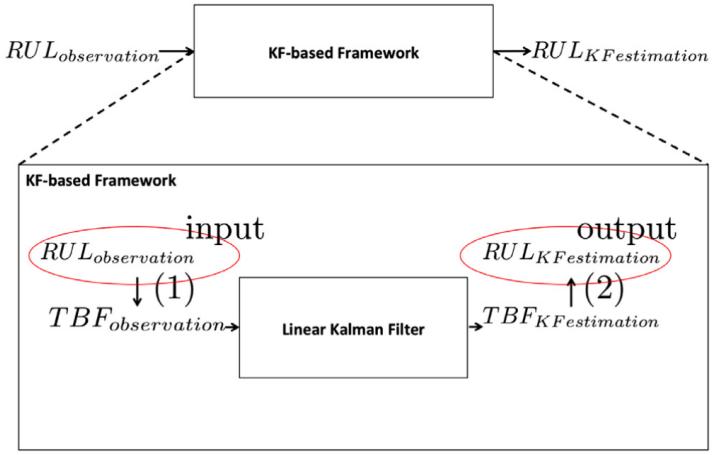

Figure 3.5: Example of Kalman Filter, extracted from [39].

After this processing, a Machine Learning algorithm is used, which is feed with the sensor measurements, and the target is the RUL predicted through the Kalman Filter. Then, it is predicted the RUL of the system, over time, and the results are compared when the Kalman Filter is used, or not, existing, in general, better results using this filter.

3.4 Summary

Over this section, some used approaches were briefly described that permit to compute the Health Indicator or the Remaining Useful Life of a certain system or component.

To identify which procedure will be followed, first of all, it is needed to recognize the type of the available data, that will help to compute the HI and the RUL. As mentioned, there are three types of approaches that are depending on the kind of data: model-based, data-driven, and hybrid approach. As the available data is just the sensors measurements, that were extracted during flights, for each trajectory, this means that it will be followed a data-driven method. The remaining approaches could not be used because there is no access to the physical state of the system.

As explained above, this thesis is divided into two main phases: diagnosis and prognosis. In the diagnosis phase, the healthy of a system will be estimated, more specifically, the Health Indicator, and in the prognosis phase, the main goal is to compute the Remaining Useful Life over trajectory time.

To compute the Health Indicator, there were proposed two approaches: one related to Artificial Intelligence, where were used Machine Learning algorithms, and another one that uses Integrated Systems Health Management. However, the approach chosen follows an artificial intelligence because is the one that will match the data available (sensor data and the respective Health Indicator, over time). Of note that the available HI was obtained from a previous thesis work [10] and then were provided to improve this dissertation.

For the prognosis phase was described two different advances: the Combining Data-Driven

and Kalman Filtering and the elbow point. The Combining Data-Driven and Kalman Filtering method was rejected because the final Remaining Useful Life is unknown. However, this problem remains in the elbow point approach. So, the idea of the adopted approach in this phase is inspired in this approach with a little difference: instead of calculating the Remaining Useful Life, it will be predicted the occurrence of a failure, as will be described in the next section.

This page is intentionally left blank.

Chapter 4

Methodologies

The datasets used, as was refered before, have the sensor measurements over each trajectory. Furthermore, the datasets have the timestamp for each measurement even as the information that described each flight, like the flight ID or the flight phase, enabling a detailed study. According to the available data were identified a different number of trajectories, for each system. Thus, it was found 8 trajectories for the Air Bleed system and 9 for the CACTCS system, which corresponds, respectively, to 8 and 9 removals.

The following sections explain, in detail, the approaches adopted for the Diagnosis and Prognosis phases. Despite the datasets belong to different systems and the respective structure are not the same, it was possible to apply the same methodology in both situations.

4.1 Diagnosis Approach

The diagnosis phase consists of predicting the health of a system, analyzing if its behavior suffered an extra degradation that could be caused by an anomaly or generate one. This information is important for the maintenance team because these predictions, combined with the knowledge of a maintenance team about a system, will allow to understanding if the HI is increasing quickly, helping to understand if a failure will occur in a near future.

The followed approach takes into account the sensor measurements and the respective HI considered in the timestamp of the extraction. Considering that the sensor measurements and the respective HI are normalized, the first step is to aggregate data, according to the aggregated phase. After that, the data is again aggregated, considering a sampling period. These two steps are important because there are thousands (and millions, in some cases) of instances to analyze that will influence the execution time.

In Figure 4.1 is shown the differences, before and after applying the aggregation data. The first column, in both of the data examples (ap), is the aggregated phase (that was explained in Section 2.1.6) that is considered. So, analyzing the left side of the scheme is exemplified the original data, and on the right side is the result of this aggregation. The result of the process is the computation of the mean, the standard deviation, for each sensor, and the variation of the Health Indicator for the interval time considered.

The next step consists of training the Machine Learning Algorithms (Linear Regression, Support Vector Machines, and Random Forest) with the means and standard deviations obtained, whose target is the variation of the Health Indicator, as is explained in Figure 4.2.

ap	S_1	S_2	...	Sn	HI
3	1	1	...	1	0
3	2	2	...	2	10
3	3	3	...	3	15
3	4	4	...	4	20
3	5	5	...	5	25
3	6	6	...	6	30
3	7	7	...	7	35
3	8	8	...	8	40

→

ap	μ S_1	σ S_1	μ S_2	σ S_2	...	μ S_n	σ S_n	ΔHI
3	1.5	0.5	1.5	0.5	...	1.5	0.5	10
3	3.5	0.5	3.5	0.5	...	3.5	0.5	5
3	5.5	0.5	5.5	0.5	...	5.5	0.5	5
3	7.5	0.5	7.5	0.5	...	7.5	0.5	5

Figure 4.1: Example of how data is aggregated.

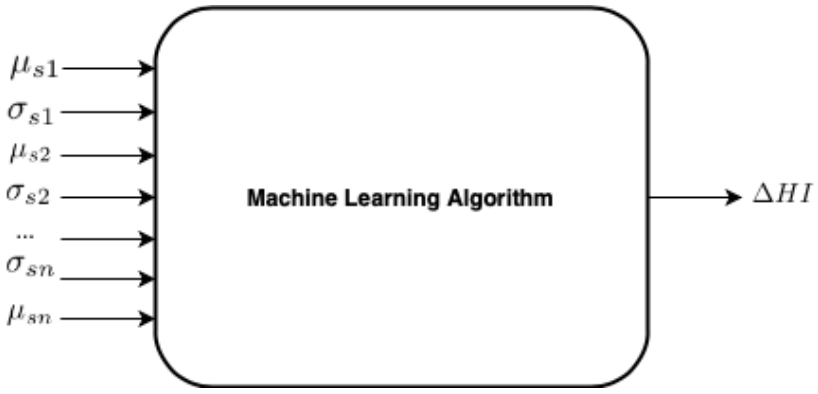

Figure 4.2: Input and output scheme of the Machine Learning Algorithm.

This approach was chosen, because this type of approaches has associated low costs of development and there is no need of specific previous knowledge about the system, to perform a solution that could analyze the health of the system. However, this is a black box methodology, which means that, after applying the sensors measurements in the Machine Learning algorithm, there is no way to know what happens during the HI prediction process.

Before applying this approach to the Machine Learning algorithms, it was verified different sampling period, in order to know in which one the Mean Squared Error is minor, as is detailed in the Figure 4.3.

The best results were obtained when the sampling period is 10 minutes. This happens because, in this situation, is extracted more information than the in remaining ones.

Another important subject is related to the best Mean Squared Error obtained. In some algorithms, like Support Vector Regressor or Random Forest is possible to change some arguments, that will have a huge impact on the obtained results. The Linear Regression does not have a way to change the parameters, using Sklearn, in Python.

One possible parameter to change in the Support Vector Regressor is the *gamma* (γ) value, as explained in Section 2.2.2. This variable has a different performance according to the kernel function used. In this Machine Learning algorithm exists four different kernel functions and, some of them, could be influenced by this parameter as is detailed below:

- **Linear:** This function is given by the equation $K(x_i, x_j) =< x_i, x_j >$, where there

is no influence of γ parameter.

- **Polynomial:** $K(x_i, x_j) = (\gamma < x_i, x_j > +a)^b$ is the equation that describes the polynomial function. Here, γ is multiplying by a number so, how biggest is γ, biggest will be the base of the power. Moreover, b is the degree of the function, which means that if the base if higher the result will be higher too. So, to get a better final result, that minimizes the distances between i and j, the γ value would be as small as possible.

- **Radial Basis Function (RBF):** The RBF is given through the equation $K(x_i, x_j) = e^{(-\gamma \|x_i - x_j\|^2)}$. Here, the γ value defines the influence that a single training has on the result. So, higher γ value means that the model has no much tolerance when a value is predicted. However, a lower γ value means that has more tolerance, so the predicted values can be more "distant" than the true values. This way, this parameter needed to be tested for some different values.

- **Sigmoid:** The equation of Sigmoid function is $K(x_i, x_j) = tanh(\gamma < x_i, x_j > +r)$, where the γ value is multiplying by a norm. The computation of this function is made applying a hyperbolic tangent, which allows to conclude that the greater the γ, the greater the result. So, the value chosen should be lower.

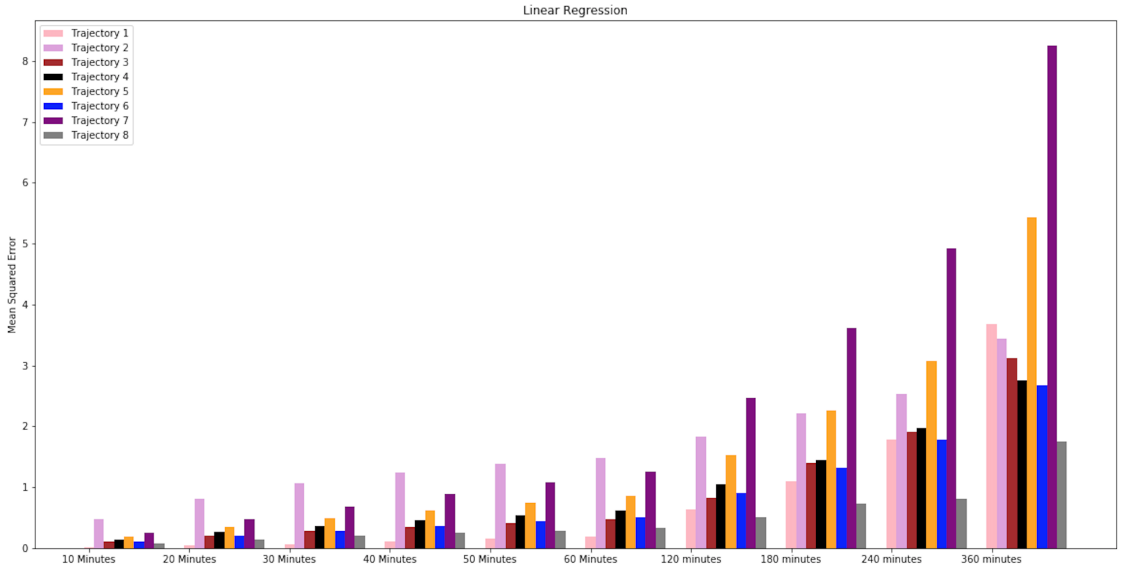

Figure 4.3: Mean Squared Error with different sampling period, using Linear Regression.

Taking into account these conditions, a battery tests were made to both systems. The γ values for each kernel function are between 0.001 and 3 because when is lower than 0.001 or higher than 3, the computation time increases, consonant the kernel function that is analyzed.

Relatively to the Random Forest, the parameter that could be changed is the number of estimators, which means, the number of trees that are created. The number of estimators varied between 2 and 100, however there are illustrated just 3 different numbers for this parameter. The number of estimators stops at 100 because, after that, performance stagnates and the obtained MSE is the same.

4.2 Prognosis Approach

In the prognosis phase, the main goal is to compute the RUL. However, the information available does not allow to estimate this value over the trajectory or when the slope curve degradation is higher. Moreover, the founded solutions depend on the knowledge of the entire trajectory. So, the solution proposed is to predict when a failure will occur based on the sensor data, using the Discrete Fourier Transform. For this analysis was not considered all data sensors, because some aggregated phases have some noise, that will produce outliers and could affect the final result. This way, it was just analyzed the data measured during the third aggregated phase (cruise), for all flights, in each trajectory, because this phase is more stable and produce fewer outliers. The considered process for the prognostics phase is detailed in Figure 4.4.

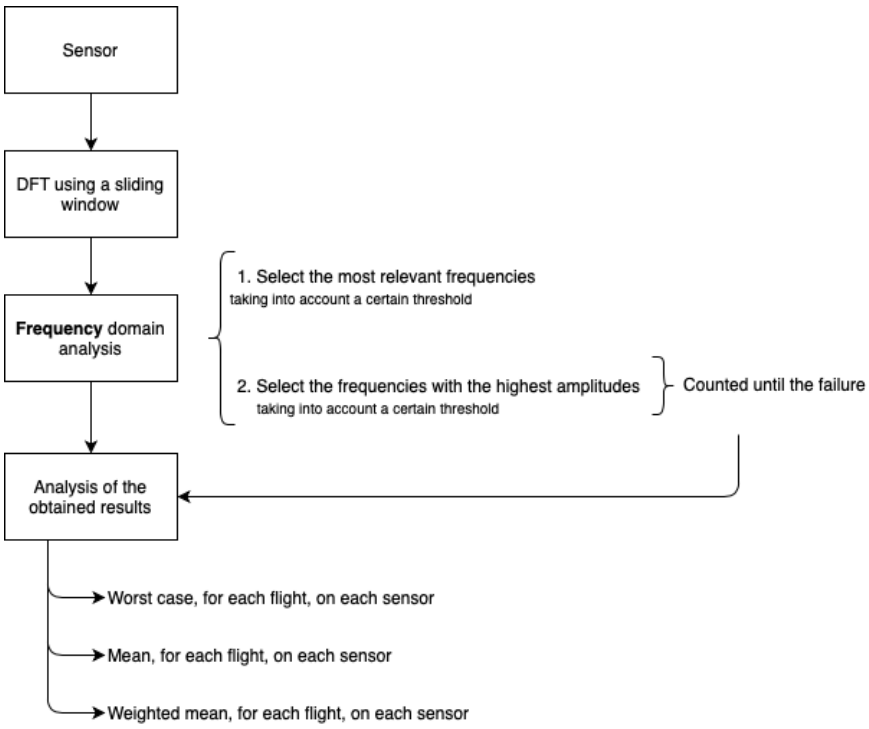

Figure 4.4: Approach flowchart.

For each sensor, in each flight, will be applied a DFT using a sliding window, with 10 minutes of the sampling period. The DFT function returns the transformation of the signal into the frequencies, and then the 80% highest are selected. After that, from the 80% highest peaks selected, the most relevant frequencies are chosen, and the ones that have higher amplitudes. Then, the amplitudes that were selected are counted based on a defined threshold, either for the frequencies or amplitudes.

This threshold was determined according to the behavior in each flight. So, in both situations, were computed the maximum of the frequency and the maximum of the amplitude obtained. Then, these values were multiplied by 0.25 and 0.5 (empiric values, after a few tests), respectively, selecting just a small number of frequencies and amplitudes. Finally, the number of highest amplitudes are counted and normalized, because some trajectories have more noise than others, which could return scattered values. So, the final normalized result returns 0, 1 or 2, according to the following rules:

- **0:** Which means that there is no highest amplitude for the highest frequencies, under the defined threshold, detected;
- **1:** If there is 1 or 2 amplitudes for the highest frequencies found, above the threshold;
- **2:** If there are more than 2 amplitudes for the highest frequencies, above the threshold, detected;

Then, all the counter results are saved (one value for each flight) and are analyzed until the occurrence of the first failure, sensor by sensor, taking into account the three different approaches mentioned in Figure 4.4, and that will be described below.

Worst Case

In Figure 4.5 is shown what happens when is considered the Worst Case approach. For each sensor, in each flight, the value of the normalized counter obtained (blue points) is compared. So, the major value between each sensor, for each flight, is considered as the worst situation. Then, this value is saved (orange points) until the failure. The result of this approach is the sum of the orange points.

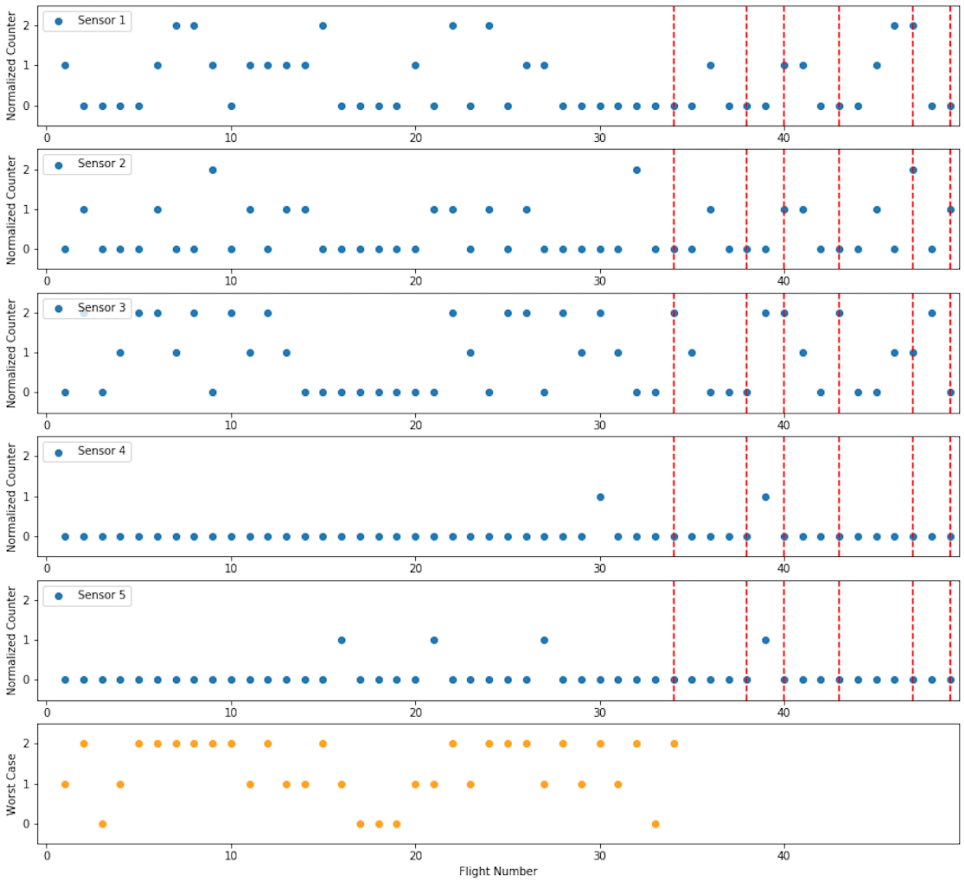

Figure 4.5: Worst case approach, for the first trajectory, in Air Bleed System.

Mean

In Figure 4.6 is described what is the procedure when the Mean Approach is considered. For each sensor, it will be seen the obtained counter for each flight, and then the mean is computed. The result, one for each flight, is saved (which is represented through the orange dots), and the final result is the sum of these orange dots, until the occurrence of the first failure.

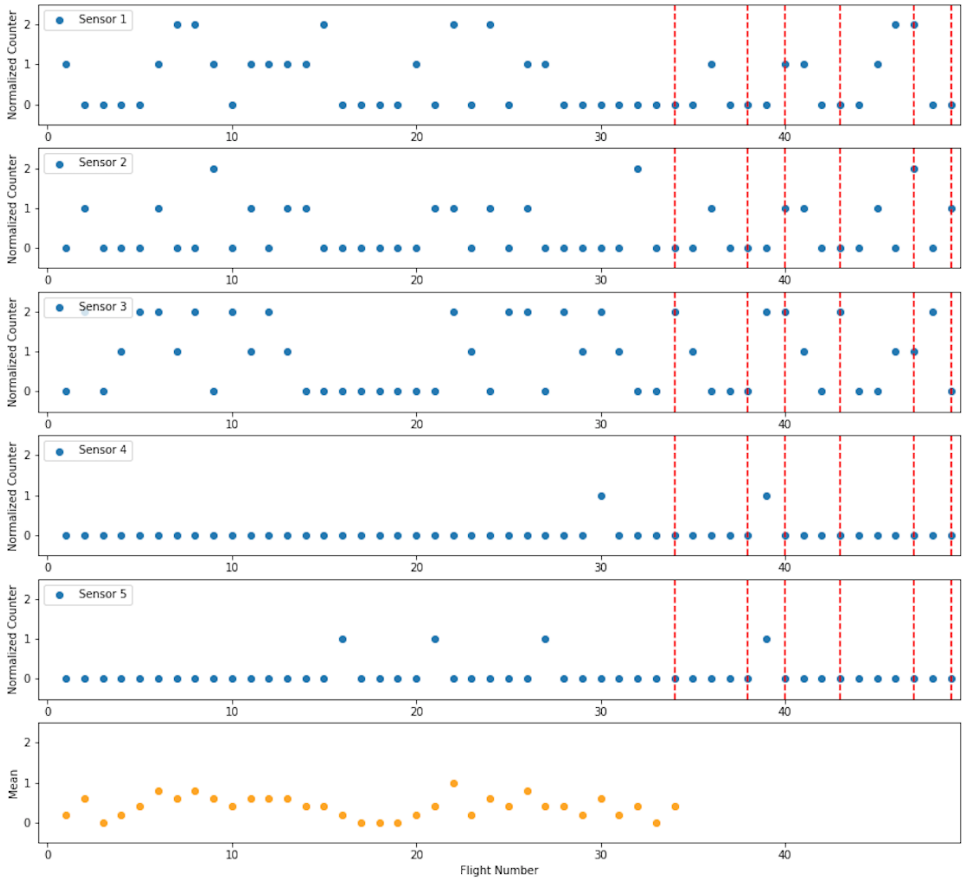

Figure 4.6: Mean approach, for the first trajectory, in Air Bleed System.

Weighted Mean

The result obtained for the weighted mean approach is detailed in Figure 4.7. For each flight, it will be analyzed the result for each sensor. Then, the number of ones and the number of twos will be counted and attributed a weight. The weight for the number of occurrences of numbers two is different from the number of occurrences of ones because when is obtained the number two, it could be a reflection of a major degradation. So, for each flight, each orange point is obtained by Equation 4.1.

$$WeightedMean = \frac{n_{ones}}{150} + \frac{n_{twos}}{115} \tag{4.1}$$

The final results, described in Chapter 6, are computed through the sum of the orange points. Of note that these weights are a result from several tests, but they are not validated by the maintenance team.

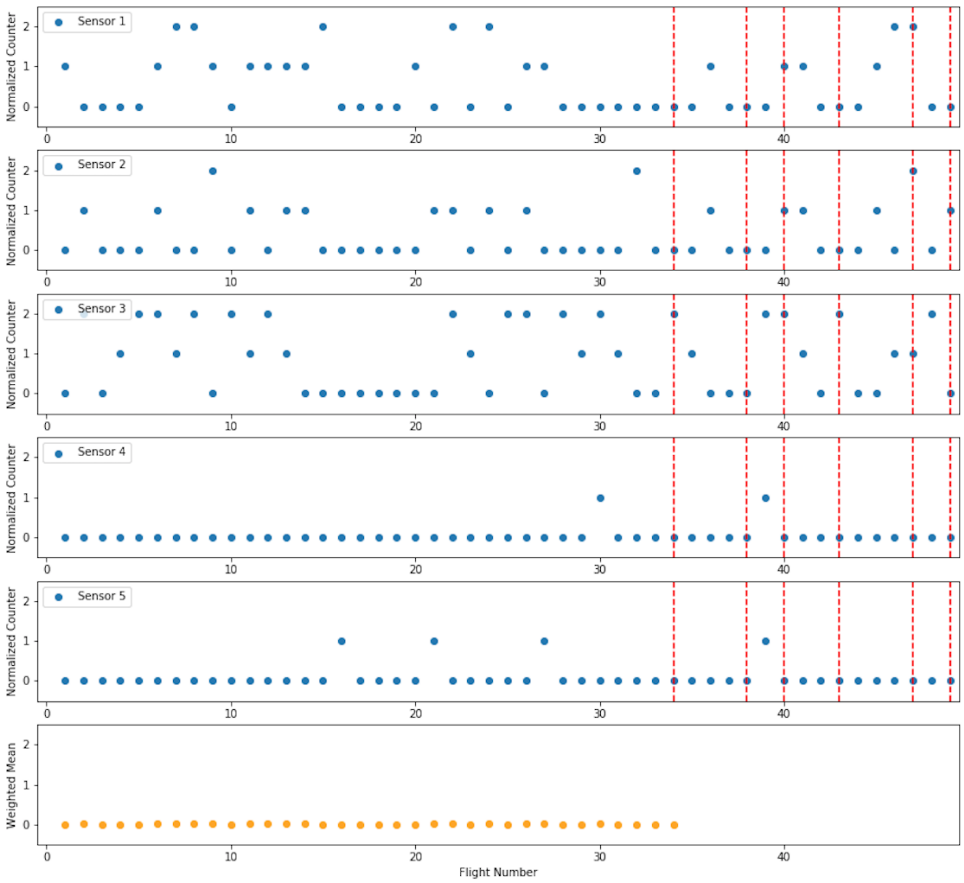

Figure 4.7: Weighted mean approach, for the first trajectory, in Air Bleed System.

This page is intentionally left blank.

Chapter 5

Case Studies

This chapter describes the systems where was implemented the approach described in Chapter 4. Both of them are responsible for bringing the air from the outside to inside of the airplane. However, they are used in different airplanes. Moreover, they have some differences in the process of how the air is transported and processed, as will be explained in the following sections.

Besides that, this section reports the structure of the datasets used to develop the solutions to the problems, and the respective pre-processing done to obtain better results.

5.1 Air Bleed System

The Air Bleed system dataset belongs to the Boeing 747 and is one of the systems that was submitted to this case study. The following Sections will describe this system as well as, describe what were the necessary data used to develop the approach.

5.1.1 How does Air Bleed system works?

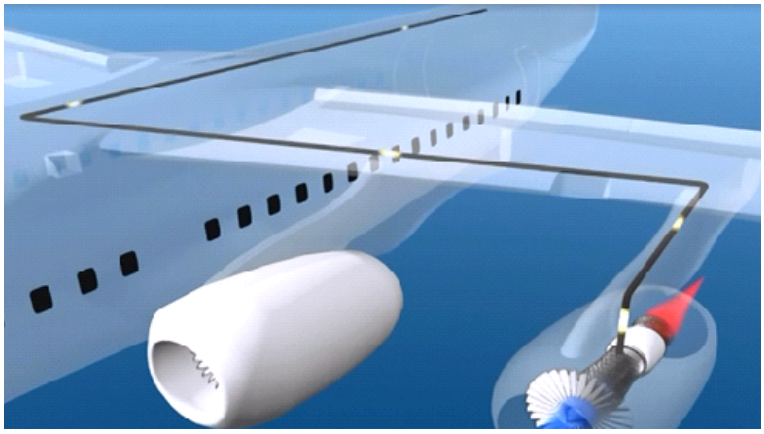

Figure 5.1: Air Bleed System example.

Figure 5.1 shows the place, in the airplane, where the Air Bleed system is. This system has a fan that extracts the air from the outside and conducts it to the places where it is necessary. As the air is carried to different places, its characteristics are changed with the

help of the ducts, valves, and regulators. These components will adjust the temperature and the pressure of the air, consonant the final destiny [40] [41].

The Air Bleed System is responsible for carrying the air that was previously processed to the engine start, air conditioning, water system pressurization, among others.

5.1.2 Dataset Struture

The dataset that belongs to the Air Bleed system has eight trajectories, and each trajectory has a different number of flights. However, all of them have the same information, which will be described below:

- **H51394154:** Index value for each flight;
- **H20790882:** It's a number that is associated to the flight phase:

  1. Power on
  2. Engine start
  3. Taxi out
  4. Unknown
  5. Take-off roll
  6. Initial climb
  7. Climb
  8. Cruise
  9. Descent
  10. Approach
  11. Rollout
  12. Taxi in
  13. Unknown
  14. Engine shutdown

- **H13768180:** It's the date that the data file was created. This file contains the data that was extracted from sensors;
- **H62778170:** It's the tail number (plane ID);
- **H64936356:** Filename where the values of sensors were saved;
- **H73670103:** It represents a time;
- **H22124930:** It represents a date;
- **H55759866:** It corresponds to a second identifier, like the flight ID;
- **custom_id:** Is the flight number;
- **my_flight_counter:** Flight counter that is restarted on each trajectory;
- **System:** Each airplane has 4 Air Bleed systems. So, this column is a number between 1 and 4, and represents what system that corresponds the data analyzed.
- **HI:** Health Indicator value. This value will increase over time, starting in zero, in each trajectory;
- **presence_FDE:** This column indicates if occurred, or not, a failure or was emitted a warning. If one of these situations happened, it will appear the number 1. Otherwise, it will appear 0.
- **s_1:** Sensor values for the shaft 1;
- **s_2:** Sensor values for the shaft 2;

- **s_3:** Sensor values for the oil temperature;
- **s_4:** Sensor values for the air pressure;
- **s_5:** Sensor values for the air temperature;
- **aggregated_phase:** It's a number, between 1 and 5, that is corresponded to one, or more, phase flights. These phases were aggregated taken into account the sensor values and the respective variations over the time. In Table 2.1 was explained which aggregated phase corresponds to each phase flight;
- **timestamp:** The timestamp that the data sensors was measured.

For this work, it was not necessary to use all of these data. So, the most important ones considered are:

- **timestamp:** Because is important to know the duration of a flight and the behavior during it, in order to analyze the results;
- **s_i:** Where i is between 1 and 5. This corresponds to the sensor measurements; they are important, because they are used either in diagnosis or prognosis;
- **aggregated_phase:** Because is essential to divide the dataset in the diagnosis phase, according to the approach described in Section 2.1.6;
- **HI:** Once is needed to the diagnosis phase.

The remaining ones were not used, so the pre-processing of data will focus on these ones, more specifically, in sensor data and HI values.

5.1.3 Preprocessing of the Air Bleed data

Before using the datasets either diagnosis or prognosis approach, the data was processed. First of all, the sensor measurements, as well as the variation of the Health Indicator, were normalized into the interval between 0 and 1.

Specifically for the prognosis phase, the outliers were removed, because the behavior of the sensors has some values unusual (outliers), taking into account the pattern of the sensor in general, over the aggregated phase 3.

5.2 Cabin Air Conditioning and Temperature Control System

The Cabin Air Conditioning and Temperature Control System (CACTCS), belongs to Boeing 787, and is similar to the Air Bleed system. However, this system suffered an improvement from the previous systems (like the Air Bleed System) to guarantee safety and loyalty for the airline and the respective costumers.

5.2.1 How does CACTCS works?

The difference between this system and the Air Bleed System is mainly the way the air is extracted and used in the different places in the airplane. CACTCS has an architecture that allows distributing the air to the cabin in an efficient way, that does not use so many sources during the flight, decreasing the fuel spent, as well as, maintaining engine power during cruise phase [42] [43] [44].

CACTCS has compressors that are activated in an electrical form, instead of a mechanical form that needs and spends more fuel for the same objective. The final purpose of this system is the same of Air Bleed system.

5.2.2 Dataset Struture

This case study focuses on Air Conditioning Pack which appearance is shown in Figure 5.2.

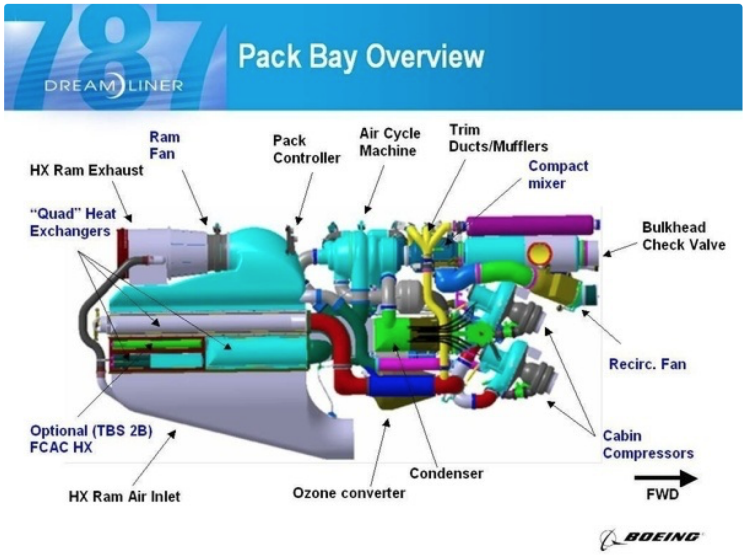

Figure 5.2: Air Conditioning Pack, extracted from [42]

Each Boeing 787 has two packs of Air Conditioning packs: one on the left side (left pack) and another on the right side (right pack), under the airplane.

The sensors that were studied and analyzed concern to Cabin Air Compressor (CAC) system. As was shown in Figure 5.2, the Air Conditioning Pack has two CACs and, each CAC has 11 sensors, which result in a total of 22. Moreover, the Air Condition Pack has 23 other sensors that do not belong to the CAC, whereby they were not considered.

Beyond these 45 sensors (22 for the CAC and 23 for the other components), the dataset has 12 informative features like the timestamp, flight ID, flight phase, among others, that are useful to understand the behavior of the trajectory, as well as the problem. This way, the dataset has 102 anonymous features - twelve informative features plus 90 sensor measurements (45 for each pack).

5.2.3 Preprocessing of the CACTCS data

The results in the diagnostic phase are depending on the Machine Learning algorithms. They are very sensitive to the inputs and, because the input data and the target have a huge range, the better way to train these algorithms is to process them. So, to estimate the variation of HI, the data will be normalized in an interval between 0 and 1, as was done in Air Bleed data.

Another consideration for this dataset is that there are no data available that permits to construct complete trajectory. Thus, the trajectories have more or less 3000 hours. This value corresponds to the minimum flight hours considered that a system is submitted, defined by the maintenance team. In total, exist 9 trajectories that will be analyzed. The removal of the outliers was made, as in the Air Bleed system.

This page is intentionally left blank.

Chapter 6

Results

This chapter describes the obtained results for each case study, as well as, for each approach. The first section explains the obtained results either for diagnosis or prognosis, for Air Bleed system. The second section describes the same results for CACTCS system.

6.1 Air Bleed System

The following sections describes the obtained results for the Air Bleed system, applying the diagnosis and prognosis approach, described in Chapter 4.

6.1.1 Diagnosis

In the diagnosis phase is analyzed how healthy is a system, considering the degradation that is presented through the data. One of the most important steps, as mentioned over this document, is to predict how will be the health of the system in future flights, considering the past and the actual data, analyzed until then. This way, the adopted approach aims to predict this state and was divided into two different ways:

- **Train and Test Trajectories:** A trajectory is used to train a Machine Learning algorithm, and then, a model is obtained that will be used to predict the Health Indicator in the remaining trajectories.
- **1 Trajectory to Train and Test:** A percentage (30%, 40%, 50%, 60%, or 70%) of a trajectory is used to train an algorithm, and then, the created model is used to predict the health in the remaining trajectory.

Note that the results obtained, in this section, were analyzed in the same conditions, which means that the chosen parameters for each algorithm are the same for both situations. This way, the Linear Regression has no changed parameters because there is no parameter to change with the library used. In SVR, when the kernel function is linear, there is no parameter to change. However, when the kernel function is Polynomial, RBF, or Sigmoid, the γ value is 0.2, 0.2, and 0.01, respectively. In the Random Forest Algorithm, the parameter that was changed was the $n_{estimators}$, that will have the values 2, 32, and 100.

Train and Test Trajectories

The approach *Train and Test Trajectories* creates a model that was previously trained, with a complete trajectory. Then, this model is used to predict the Health Indicator for the remaining trajectories. The obtained results, applying the Mean Squared Error for analyze the quality of prediction of the Health Indicator, are shown in Table 6.1. The created model that is responsible for these results was constructed based on the data for the trajectory 1. Then, it was applied to all trajectories, to know if the predicted Health Indicator was close to the real.

Table 6.1: MSE obtained for the model created with Trajectory 1, for Air Bleed system.

Trajectory	LR	SVR - Linear	SVR - Poly	SVR - RBF	SVR - Sigmoid	RF - 2	RF - 32	RF - 100
1	0.004	0.004	0.004	0.003	0.004	0.001	0.001	0.001
2	0.007	0.007	0.026	0.006	0.006	0.009	0.007	0.007
3	0.003	0.004	0.011	0.004	0.004	0.005	0.004	0.004
4	0.014	0.013	0.016	0.012	0.013	0.015	0.009	0.011
5	0.047	0.044	0.048	0.039	0.044	0.048	0.034	0.039
6	0.004	0.004	0.004	0.004	0.004	0.006	0.004	0.004
7	0.023	0.021	0.025	0.021	0.022	0.029	0.027	0.026
8	0.002	0.002	0.002	0.002	0.002	0.004	0.002	0.002

As was expected, the minor MSE obtained is when is predicted the Health Indicator for trajectory 1, because it was the trajectory responsible for the creation of the model. Despite does not know what is happening inside of each Machine Learning algorithm, during the predicting phase, it was expected than some Machine Learning algorithms had a better performance than others. These results are a reflection of many conditions like the algorithm efficiency, the chosen parameters, or the data that were used to training the algorithm that are good for the training phase. This way, these situations produce many different conditions that are reflected in the remaining trajectories, which conduct a higher ability model to predict the Health Indicator.

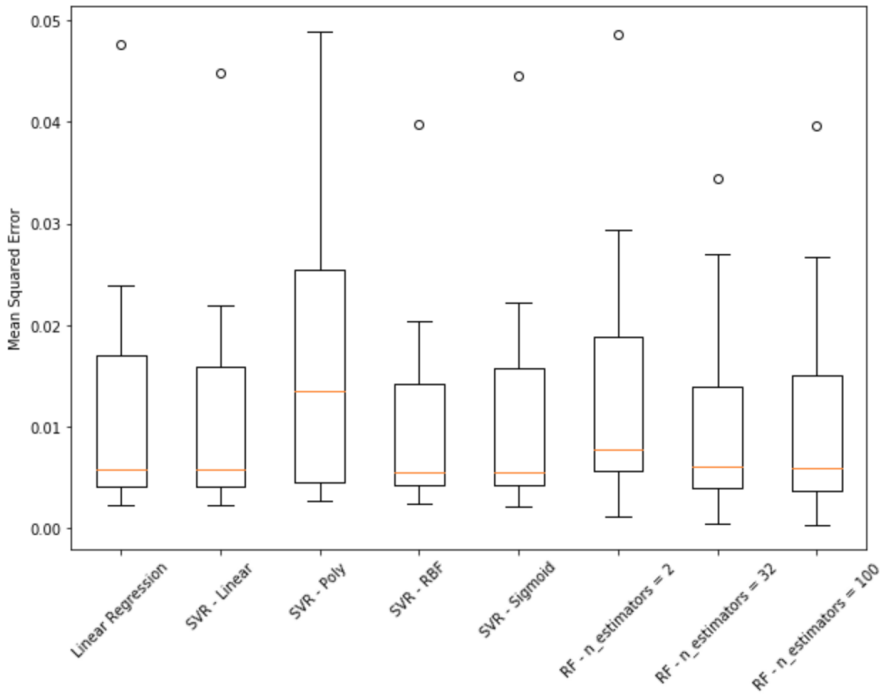

Figure 6.1: Boxplot that compares MSE obtained, for each algorithm, when the model is created with Trajectory 1, for Air Bleed system.

As explained, the results of the algorithms are depending not only on the data quality but also on the algorithms. The conjugation of these two factors is the key to the success of the results. In Figure 6.1 is shown a boxplot that describes for which algorithm was obtained the best MSE.

For the considered model, the best result was obtained with the SVR algorithm, the kernel function RBF, and the $\gamma = 0.2$. One algorithm with similar results is also the SVR algorithm but with kernel function Sigmoid, and $\gamma = 0.01$. As mentioned, the γ values were chosen empirically.

There were created more models with the remaining trajectories, and then, they were applied to the rest of trajectories, as was explained in this topic, whose results obtained are in Appendix A. Analyzing these results, it is verified that the obtained Mean Squared Error is similar to the results that were described above. For most situations, the algorithm SVR has the best MSE, with the RBF kernel function. However, the Random Forest with a number of estimators equal to 32 and 100 have good performance, too. The results in Random Forest are very similar because after a certain number of estimators the final result tends to stagnate.

1 Trajectory to Train and Test

The approach *1 Trajectory to Train and Test* creates a model that was previously trained, with the first 30% of instances of a trajectory. Then, this model is used to predict the Health Indicator for the remaining trajectory. In Table 6.2 is shown the obtained MSE when is used the first 30% instances of each trajectory to train a Machine Learning algorithm. The created model is used to predict the Health Indicator in the left 70% of the data of each trajectory.

Table 6.2: MSE obtained with the models created, for each trajectory, for the first 30% instances, for Air Bleed system.

Trajectory	LR	SVR - Linear	SVR - Poly	SVR - RBF	SVR - Sigmoid	RF - 2	RF - 32	RF - 100
1	0.004	0.005	0.005	0.005	0.005	0.007	0.006	0.006
2	0.006	0.006	0.007	0.006	0.006	0.007	0.006	0.006
3	0.003	0.004	0.004	0.003	0.004	0.005	0.004	0.003
4	0.009	0.009	0.011	0.011	0.009	0.011	0.010	0.010
5	0.009	0.009	0.011	0.010	0.011	0.010	0.008	0.008
6	0.004	0.004	0.004	0.004	0.004	0.006	0.004	0.004
7	0.010	0.008	0.014	0.007	0.010	0.008	0.006	0.006
8	0.002	0.002	0.045	0.002	0.002	0.002	0.002	0.002

The obtained MSE is similar in all algorithms, for all trajectories. It could mean that the data that was used to train the algorithm is good enough to predict what will be the future HI, in the remaining trajectory. Moreover, the main conclusion of these results is that the set of circumstances, like the duration flights, or the behavior of the data in each flight, who make part of the creation of the model, were reached and considered in the future situations (in the remaining 70% of each trajectory).

Observing the Figure 6.2 the algorithms whose had better performance were the SVR algorithm with the kernel function RBF, and the $\gamma = 0.2$, and the SVR algorithm, but with kernel function Sigmoid, and $\gamma = 0.01$. These results, as happened previously, are depending on the data and the chosen parameters.

Moreover, there were created more models with 40%, 50%, 60%, and 70% of the first instances, for all trajectories, which results are in Appendix A.

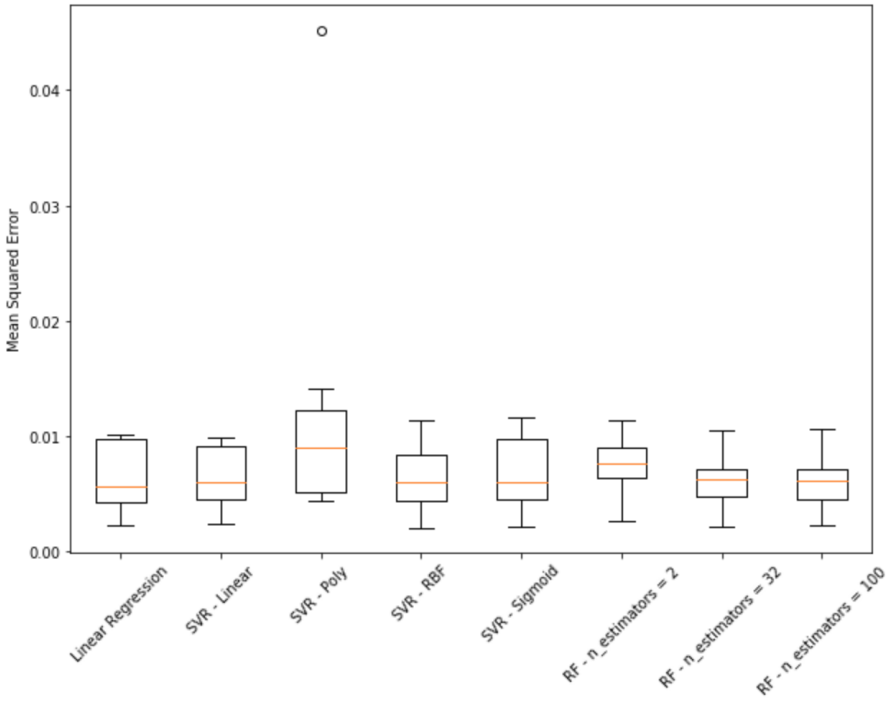

Figure 6.2: Boxplot that compares the MSE obtained, for each algorithm, when the model is created with 30% of data, for each trajectory, for Air Bleed system.

Analyzing these results, it was expected that the MSE would decrease the amount of data the model has. However, this did not happen. In most situations the MSE remained the same or increases. The reason why this value does not decrease could be related to the fact that the data have, more or less, the same duration of flights (when the MSE is the same), or the data have noise that will affect the performance and, consequently, the results (when the MSE increases). Furthermore, inspecting the boxplots of these results, for most situations, the Random Forest algorithm with a number of estimators equals to 32, and 100 was the one that had better results. Nevertheless, the SVR algorithm with the RBF kernel function was also good.

6.1.2 Prognosis

The prognosis phase consists of predict when a failure will occur. This prediction is made taking into account the signal frequency and magnitude analysis, using a DFT. After defining a certain threshold, the highest peaks above it are counted, normalized and analyzed, using three different approaches:

- **Worst Case:** For each sensor, in each flight, is analyzed what was the highest normalized value obtained, and it is saved. The result of this approach is the sum of the highest values for each flight.
- **Mean:** For this approach is computed the mean of the normalized result, for each sensor, in each flight. The result of this approach is the sum of the mean obtained for each flight.
- **Weighted Mean:** In this approach is attributed weight to the normalized result, for each sensor, in each flight. The result of this approach is the sum of the weighted results obtained for each flight.

These three approaches are detailed in Section 4.2. Table 6.3 shows the results for each prognosis approach adopted.

Table 6.3: Sum of normalized counter, for each approach, considering the trajectory until the failure, using Air Bleed system.

Trajectory	**Worst Case**	**Mean**	**Weighted Mean**
1	46	13.6	0.365
2	147	50.4	1.313
3	74	25.8	0.688
4	37	13.8	0.367
5	160	62.0	1.570
6	126	39.4	1.035
7	74	29.4	0.720
8	55	17.2	0.443

The obtained results are inconsistent in the three approaches adopted, which leads to conclude that this is not a good approach to follow. However, this result can be different for each trajectory because it could be dependent on the trajectory dimension, the flight that occurs the first failure, or some noise in sensor measurements. As the outliers were removed in a previous step, the remaining hypothesis will be tested.

In order to know if these results are depending on the related topics, it was verified which trajectory has the first occurrence of the failure, and then, the dimension of this trajectory will be considered in the remaining ones. So, as in trajectory 1 occurs the first failure at flight 34, earlier than others, will be considered the 34 flights before the failure in other ones, because it is necessary to guarantee that the trajectories start at the same time. This could result in a problem, once with the available data is impossible to know if the trajectory comes from a new system or a repaired system, being unfair to compare two different systems that will have different behavior before the occorrence of a failure. The results are shown in Table 6.4.

Table 6.4: Sum of normalized counter, for each approach, considering the minor trajectory, using Air Bleed system.

Trajectory	**Worst Case**	**Mean**	**Weighted Mean**
1	46	13.6	0.365
2	42	15.4	0.411
3	35	11.2	0.303
4	36	13.2	0.347
5	43	19.8	0.493
6	41	12.6	0.317
7	42	14.6	0.375
8	39	13.4	0.533

To analyze the difference of the results for each approach, between the first trajectory and the remaining others, the relative error was computed. This way, it is possible to understand if the results of these approaches are too far from the initial trajectory, concluding if they reflect or not a good approach to follow. In Table 6.5 are represented the obtained results for the relative error.

The obtained percentages indicate that, in general, the results are very close between Trajectory 1 and the remaining ones, concluding that the three approaches, apparently,

Table 6.5: Comparison of approaches, using Air Bleed system, with minor trajectory dimension until the failure.

Trajectory	Worst Case	Mean	Weighted Mean
1	0%	0%	0%
2	$\frac{\lvert 42-46\rvert}{46} * 100 = 8.695\%$	$\frac{\lvert 15.4-13.6\rvert}{13.6} * 100 = 13.235\%$	$\frac{\lvert 0.411-0.365\rvert}{0.365} * 100 = 12.603\%$
3	$\frac{\lvert 35-46\rvert}{46} * 100 = 23.913\%$	$\frac{\lvert 11.2-13.6\rvert}{13.6} * 100 = 17.647\%$	$\frac{\lvert 0.303-0.365\rvert}{0.365} * 100 = 16.986\%$
4	$\frac{\lvert 36-46\rvert}{46} * 100 = 21.739\%$	$\frac{\lvert 13.2-13.6\rvert}{13.6} * 100 = 2.941\%$	$\frac{\lvert 0.347-0.365\rvert}{0.365} * 100 = 4.931\%$
5	$\frac{\lvert 43-46\rvert}{46} * 100 = 6.521\%$	$\frac{\lvert 19.79-13.6\rvert}{13.6} * 100 = 30.808\%$	$\frac{\lvert 0.493-0.365\rvert}{0.365} * 100 = 35.068\%$
6	$\frac{\lvert 41-46\rvert}{46} * 100 = 10.869\%$	$\frac{\lvert 12.6-13.6\rvert}{13.6} * 100 = 7.352\%$	$\frac{\lvert 0.317-0.365\rvert}{0.365} * 100 = 13.15\%$
7	$\frac{\lvert 42-46\rvert}{46} * 100 = 8.695\%$	$\frac{\lvert 14.6-13.6\rvert}{13.6} * 100 = 7.352\%$	$\frac{\lvert 0.375-0.365\rvert}{0.365} * 100 = 2.739\%$
8	$\frac{\lvert 39-46\rvert}{46} * 100 = 15.217\%$	$\frac{\lvert 13.39-13.6\rvert}{13.6} * 100 = 1.544\%$	$\frac{\lvert 0.533-0.365\rvert}{0.365} * 100 = 46.027\%$

could be used to anticipate the failure.

The ideal for the maintenance team is to predict failure with some antecedence, scheduling the maintenance of the aircraft timely. This way, for each approach, will be defined a threshold, assuming that the real value of the prediction is the value obtained for the first trajectory. A way to suppose this threshold could be 3 flights before the occurrence of a failure. So, if the value of failure for trajectory 1, in the worst case approach, is 46, and considering the worst scenario, the threshold defined for this situation could be 40, supposing that three flights before the failure returned the value 2. The upper bound considered is the value obtained for this trajectory plus 3 flights, in the same conditions that were defined the lower bound, for each situation. The following analyses the occurrence of true positives and false positives that are identified, considering the threshold defined bellow, for each approach.

- **Worst Case**
  - **Minimum Threshold:** 40;
  - **Maximum Threshold:** 52;
  - **True Positives:** Trajectories 1, 2, 5, 6, and 7;
  - **True Negatives:** 0;
  - **False Positives:** Trajectories 3, 4, and 8;
  - **False Negatives:** 0.
- **Mean**
  - **Minimum Threshold:** 7.6
  - **Maximum Threshold:** 20.6
  - **True Positives:** Trajectories 1, 2, 3, 4, 5, 6, 7 and 8;
  - **True Negatives:** 0;
  - **False Positives:** 0;
  - **False Negatives:** 0.
- **Weighted Mean**
  - **Minimum Threshold:** 0.245

- **Maximum Threshold:** 0.485
- **True Positives:** Trajectories 1, 2, 3, 4, 6, and 7;
- **True Negatives:** 0;
- **False Positives:** Trajectories 5 and 8;
- **False Negatives:** 0.

As explained, these results are dependent on the defined threshold. The value considered is empiric, so it is needed to be confirmed for the maintenance team and tested as well. However, in this situation, and considering the thresholds defined, the best approach was the worst case, followed by mean and weighted mean. This was an expected result, because the relative error from the first trajectory, in this approach, was the lowest in most of the situations, relatively to the remaining ones.

6.2 CACTCS System

The next sections describe the obtained results for the CACTCS system, using the approach described in Chapter 4.

6.2.1 Diagnosis

In the diagnosis phase is analyzed how healthy is a system, considering the degradation that is presented through the data. This way, the adopted approach that aims to predict this state was divided into two different ways, like in the Air Bleed system:

- **Train and Test Trajectories:** It is used a trajectory to train a Machine Learning algorithm, and then, a model is obtained that will be used to predict the Health Indicator in the remaining trajectories.
- **1 Trajectory to Train and Test:** A percentage (30%, 40%, 50%, 60%, or 70%) of a trajectory is used to train an algorithm, and then, the created model is used to predict the health in the remaining trajectory.

Of note that the results obtained, in this section, were analyzed in the same conditions, which means that the chosen parameters for each algorithm are the same for both ways. This way, the Linear Regression has no changed parameters because there is no parameter to change using the library used. In SVR, when the linear function is linear, there is no parameter to change. However, when the kernel function is Polynomial, RBF, or Sigmoid, the γ value is 0.2, 0.2, and 0.01, respectively. In the Random Forest Algorithm, the parameter that was changed was the $n_{estimators}$, that will have the values 2, 32, and 100.

Train and Test Trajectories

The approach *Train and Test Trajectories* creates a model that was previously trained, with a complete trajectory. Then, this model is used to predict the Health Indicator for the own and remaining trajectories. In Table 6.6 is shown the obtained Mean Squared Error when is used the Trajectory 1 to train a Machine Learning algorithm.

Table 6.6: MSE obtained for the model created with Trajectory 1, for CACTCS.

Trajectory	LR	SVR - Linear	SVR - Poly	SVR - RBF	SVR - Sigmoid	RF - 2	RF - 32	RF - 100
1	0.0002	0.002	0.006	0.006	0.005	0.0	0.0	0.0
2	35.882	3.252	3.185	3.209	3.288	3.055	2.921	3.059
3	50.004	3.494	3.372	3.457	3.570	3.222	3.445	2.878
4	6.467	3.607	3.275	3.309	3.377	3.298	2.918	2.568
5	19.857	3.430	3.284	3.302	3.379	3.401	2.970	2.976
6	33.748	3.303	3.247	3.263	3.352	1.938	3.145	3.227
7	0.0004	0.001	0.006	0.006	0.005	2e-05	2e-05	2e-05
8	22.395	3.288	3.094	3.112	3.195	2.823	3.214	2.967
9	19.797	3.270	3.129	3.147	3.229	3.551	3.018	3.257

Analyzing the results, the worst algorithm, when is applied to the mentioned approach, is the Linear Regression. As was expected, the best MSE obtained was for trajectory 1 because is the trajectory that trained the algorithm and that created the model. However, stands out the prediction results for trajectory 4. The MSE here is lower because the characteristics of the first trajectory and the fourth are similar regarding the flights duration. The remaining algorithms have, in general, good results because the parameters could help them have a better performance.

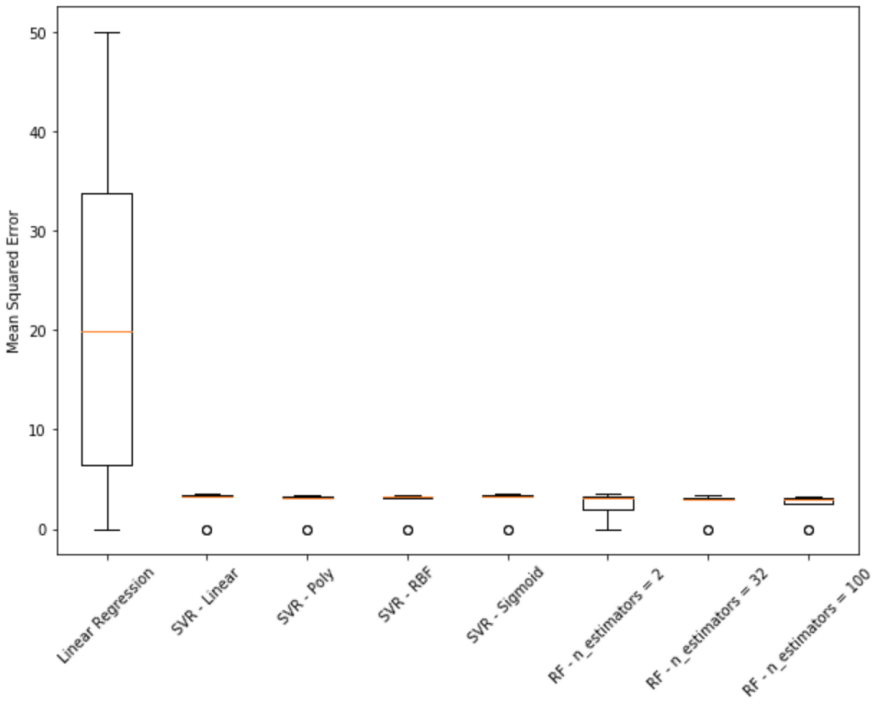

Figure 6.3: Boxplot that compares MSE obtained, for each algorithm, when the model is created with Trajectory 1, for CACTCS.

Examining the boxplot shown in Figure 6.3, the SVR and the Random Forest algorithms have better performance than the Linear Regression, as observed. However, the adjustment of the parameters could have a higher impact but, as mentioned, in the Linear Regression algorithm, there are no parameters to change in the library used.

Relatively to the remaining results, they are in Appendix B. Analyzing them, the conclusion is that the best algorithms that predict the Health Indicator are mainly the SVR with Sigmoid kernel function and the Random Forest, where the number of estimators seems not to have a significant impact. However, analyzing the tables in the same Appendix, the conclusion is that there are models that are better to predict the Health Indicator than

others. This could be caused by the diversity of characteristics of each trajectory, which means that exist trajectories with more consistent flight hours and data behavior, for each flight, and others where these characteristics are not consistent. So, when a trajectory is consistent in his behavior, and the model is created according to it, the result of the HI prediction for the trajectories where the duration of flights is not consistent will be not so good. However, the chosen parameters could improve these situations.

1 Trajectory to Train and Test

The approach *1 Trajectory to Train and Test* creates a model that was previously trained, with the first 30% of instances of a trajectory. Then, this model is used to predict the Health Indicator for the remaining trajectory. In Table 6.7 is shown the obtained Mean Squared Error when is used the first 30% instances of all trajectories to train a Machine Learning algorithm. The created models are used to predict the Health Indicator in the left 70% of data, of each trajectory.

Table 6.7: MSE obtained with the models created, for each trajectory, for the first 30% instances, for CACTCS.

Trajectory	LR	SVR - Linear	SVR - Poly	SVR - RBF	SVR - Sigmoid	RF - 2	RF - 32	RF - 100
1	0.001	0.001	0.001	0.004	0.003	2e-05	0.0001	0.0001
2	0.102	0.101	0.100	0.093	0.108	0.102	0.070	0.069
3	0.155	0.155	0.101	0.091	0.175	0.120	0.079	0.078
4	0.305	0.115	0.080	0.085	0.123	0.095	0.058	0.056
5	0.115	0.100	0.081	0.082	0.109	0.082	0.053	0.050
6	0.089	0.090	0.084	0.079	0.099	0.088	0.060	0.059
7	0.0003	0.004	0.005	0.004	0.004	5e-05	4e-05	3e-05
8	0.109	0.100	0.089	0.088	0.104	0.087	0.056	0.056
9	0.067	0.070	0.056	0.055	0.076	0.071	0.045	0.045

The obtained MSE is similar in all algorithms, for all trajectories. It could mean that the data that trained the algorithm is good enough to predict what will be the future HI, in the remaining trajectories. Moreover, the main conclusion of these results is that the set of circumstances, like the duration flights, or the behavior of the data in each flight, who make part of the creation of the model, were reached and considered in the future situations (in the remaining 70% of each trajectory).

Analyzing Figure 6.4, the algorithm which had better performance was the Random Forest with the number of estimators equals to 32 and 100. The results in these situations are very similar because, after a certain number of estimators, the performance of the algorithm tends to stagnate. Moreover, these results, as happened previously, are depending on the data and the chosen parameters.

Moreover, there were created more models with 40%, 50%, 60%, and 70% of the first instances, for all trajectories, which results are in Appendix B.

Inspecting the results in Appendix B, for all of the considered situations, the best MSE obtained is when the Random Forest algorithm, with the number of estimators is equal to 32 and 100, is applied, because is an algorithm that works based on the evaluation of all the possible solutions and finds a result that is better than the others. Relatively to the tables in the same Appendix, it was expected that the MSE was decreasing over time, because if the algorithm has more data to train, the results should be better. So, analyzing the obtained results, the MSE was decreasing, despite the difference is not significant, what permit to conclude that the data that were added to the algorithm, when it was trained, has infomation that will make the model skillful to predict the Health Indicator.

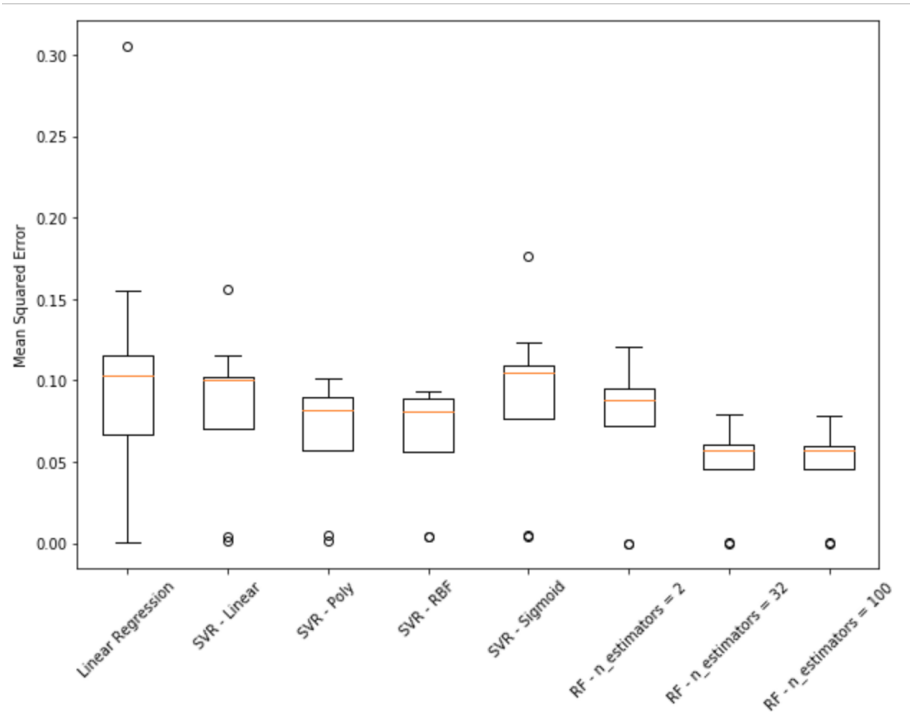

Figure 6.4: Boxplot that compares the MSE obtained, for each algorithm, when the model is created with 30% of data, for each trajectory, for CACTCS.

6.2.2 Prognosis

The prognosis phase consists of predict when a failure will occur. This prediction is made taking into account the signal frequency and magnitude analysis, using a DFT. After defining a certain threshold, the highest peaks above it are counted, normalized and analyzed, using three different approaches:

- **Worst Case:** For each sensor, in each flight, is analyzed what was the highest normalized value obtained, and it is saved. The result of this approach is the sum of the highest values for each flight.
- **Mean:** For this approach is computed the mean of the normalized result, for each sensor, in each flight. The result of this approach is the sum of the mean obtained for each flight.
- **Weighted Mean:** In this approach is attributed weight to the normalized result, for each sensor, in each flight. The result of this approach is the sum of the weighted results obtained for each flight.

These three approaches are detailed in Section 4.2. Table 6.8 shows the results for each prognosis approach adopted.

The obtained results are inconsistent in the three adopted approaches as happened with the results of the prognostic in the Air Bleed system. Moreover, as mentioned before, this result can be different for each trajectory because it could be dependent on the trajectory dimension, the flight where the first failure occurs, or some noise in sensor measurements.

As was done for the Air Bleed system, the shortest trajectory was verified. That is, the trajectory with fewer flights from the beginning to the appearance of the failure was

Table 6.8: Sum of normalized counter, for each approach, considering the trajectory until the failure, using CACTCS.

Trajectory	Worst Case	Mean	Weighted Mean
1	89	16.727	0.971
2	146	30.909	1.853
3	185	38.818	2.234
4	119	20.090	1.246
5	111	20.454	1.235
6	229	22.272	2.509
7	288	57.818	3.210
8	145	28.540	1.708
9	169	27.818	1.585

considered, and this dimension will be regarded in the others. Thus, following this idea, the 181 flights until the occurrence of the fault will be considered in all trajectories, because this is the smallest number of flights between the beginning of the trajectory and the fault, considering the 9 available trajectories. This way, it is simulated that all trajectories start at the same time and the results are shown in the Table 6.9.

Table 6.9: Sum of normalized counter, for each approach, considering the minor trajectory dimension until the failure, using CACTCS.

Trajectory	Worst Case	Mean	Weighted Mean
1	89	16.272	0.971
2	100	20.272	1.278
3	115	24.181	1.397
4	99	16.818	1.047
5	85	15.636	0.937
6	147	27.909	1.606
7	175	35.454	1.955
8	82	15.818	0.955
9	156	26.636	1.457

The obtained results indicate that, in general, the values obtained are not close between trajectory 1 and the remaining ones. It could happen because the flight duration is more inconstant than the other dataset. To confirm this statement, in the Table 6.10, is computed the relative error, between the minor trajectory - Trajectory 1 - and the remaining ones.

As could be seen, the relative error from the first trajectory to the remaining ones is very high, which means that after defining a threshold for each approach, it could be very complicated to detect a failure in time. However, a description of when a failure is detected correctively or not, according to a defined threshold - considering 3 days before the occurrence of failure - is described below.

Table 6.10: Relative error, for each approach, considering the minor trajectory dimension until the failure, using CACTCS.

Trajectory	Worst Case	Mean	Weighted Mean
1	0%	0%	0%
2	$\frac{\lvert 100-89\rvert}{89} * 100 = 12.359\%$	$\frac{\lvert 20.272-16.272\rvert}{16.272} * 100 = 24.982\%$	$\frac{\lvert 1.278-0.971\rvert}{0.971} * 100 = 31.616\%$
3	$\frac{\lvert 115-89\rvert}{89} * 100 = 29.213\%$	$\frac{\lvert 24.181-16.272\rvert}{16.272} * 100 = 48.604\%$	$\frac{\lvert 1.397-0.971\rvert}{0.971} * 100 = 43.872\%$
4	$\frac{\lvert 99-89\rvert}{89} * 100 = 11.23\%$	$\frac{\lvert 16.818-16.272\rvert}{16.272} * 100 = 3.355\%$	$\frac{\lvert 1.047-0.971\rvert}{0.971} * 100 = 12.976\%$
5	$\frac{\lvert 85-89\rvert}{89} * 100 = 4.494\%$	$\frac{\lvert 15.636-16.272\rvert}{16.272} * 100 = 3.908\%$	$\frac{\lvert 0.937-0.971\rvert}{0.971} * 100 = 3.501\%$
6	$\frac{\lvert 147-89\rvert}{89} * 100 = 65.168\%$	$\frac{\lvert 27.909-16.272\rvert}{16.272} * 100 = 71.515\%$	$\frac{\lvert 1.606-0.971\rvert}{0.971} * 100 = 65.396\%$
7	$\frac{\lvert 175-89\rvert}{89} * 100 = 96.629\%$	$\frac{\lvert 35.454-16.272\rvert}{16.272} * 100 = 117.883\%$	$\frac{\lvert 1.955-0.971\rvert}{0.971} * 100 = 101.338\%$
8	$\frac{\lvert 82-89\rvert}{89} * 100 = 7.865\%$	$\frac{\lvert 15.818-16.272\rvert}{16.272} * 100 = 2.79\%$	$\frac{\lvert 0.955-0.971\rvert}{0.971} * 100 = 1.647\%$
9	$\frac{\lvert 156-89\rvert}{89} * 100 = 75.280\%$	$\frac{\lvert 26.636-16.272\rvert}{16.272} * 100 = 63.692\%$	$\frac{\lvert 1.457-0.971\rvert}{0.971} * 100 = 50.051\%$

- **Worst Case**
  - **Minimum Threshold:** 83
  - **Maximum Threshold:** 95
  - **True Positives:** Trajectory 1;
  - **True Negatives:** 0;
  - **False Positives:** Trajectory 1 2, 3, 4, 5, 6, 7, 8, and 9;
  - **False Negatives:** 0;
- **Mean**
  - **Minimum Threshold:** 10.727
  - **Maximum Threshold:** 22.727
  - **True Positives:** Trajectory 1, 2, 4, 5, and 8;
  - **True Negatives:** 0;
  - **False Positives:** Trajectory 3, 6, 7 and 9;
  - **False Negatives:** 0;
- **Weighted Mean**
  - **Minimum Threshold:** 0.851
  - **Maximum Threshold:** 1.901
  - **True Positives:** Trajectory 1, 2, 3, 4, 5, 6, 8, and 9;
  - **True Negatives:** 0;
  - **False Positives:** Trajectory 7;
  - **False Negatives:** 0.

These results are dependent on the defined threshold. The value considered is empiric, so it is needed to be confirmed for the maintenance team and tested as well. However, in this situation, and considering the thresholds defined, the best situation was the mean approach. As was expected, the results were not good, in general, because the relative error from the first trajectory, in all approaches, is very high, which impossibilities predicting the error before it occurs with some confidence. The reason why this happens is that the time of each flight has a significant variation in the same trajectory.

6.3 Summary of Results

The mentioned approaches, for both datasets, have promising results.

For the diagnosis contribution, in general, the MSE obtained, for all algorithms, is low. This means that this approach could be good to predict the HI, over a trajectory. More specifically, if the HI will have an unexpected degradation or if the system, after a certain moment, will have a more accentuated degradation. For this situation, as mentioned, a battery of tests were made empirically.

Despite the results for the prognosis contribution are interesting, the results are dependent on the trajectory dimension as well as the occurrence of the first failure. This approach is also dependent on the chosen thresholds. Regardless that they were defined taking into account the same conditions, the approach to defining it can be improved.

However, the fact that it is not know whether a system is new or repaired, is a problem for both approaches, which can be reflected in the final results, such as the evolution of the HI or the prediction of the failure.

This page is intentionally left blank.

Chapter 7

Conclusion

This work, as mentioned several times over this document, is inserted in a European Project - ReMAP - which the main goal is to develop solutions, based on the condition of the system, replacing the fixed inspections by adaptive interventions.

As was detailed, this thesis is divided into two main contributions: diagnosis and prognosis. For the diagnosis, the main goal was to predict the Health Indicator every 10 minutes, using 3 different Machine Learning algorithms - Linear Regression, Support Vector Regressor, and Random Forest. With this approach could happen false positives either for a prediction that will mislead the maintenance team or for a prediction that will not advise the maintenance team timely, causing more costs for the airline company. For the prognosis phase, the major challenge was to find a way that could predict the RUL, without the knowledge about the future. So, to solve this problem, an approach that anticipates the failure was adopted, considering the behavior of the sensor data. The biggest challenge in this contribution was to obtain consistent results between trajectories. When each approach is analyzed, for each trajectory, exists some discrepancy between the obtained values. These differences occur because is possible that some trajectories are associated with new or repaired systems. So, it is possible that reflects a different behavior and consequently early or late detection of the failure. However, it is not possible to be certain which situation is regarded for each trajectory. Also, the results in this approach could be affected according to the considered thresholds, either for the extraction of the relevant frequencies or the anticipation of a failure. As each sensor has a different range of values, it means that the thresholds are dependent on the sensor and the system.

Analysis of the Schedule Plane

The proposed schedule plane for the second semester, is illustrated in Figure 7.1.

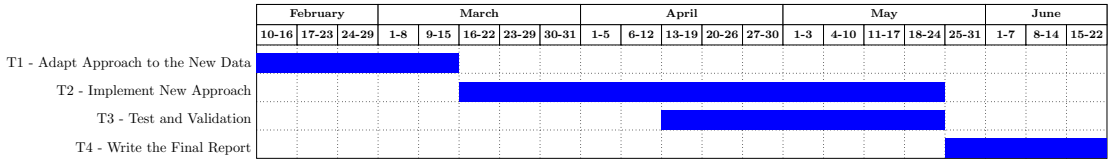

Figure 7.1: Proposed Gantt Chart of the Second Semester.

However, there happened some delays, because to improve the approach, it was necessary

to research the way that the RUL could be calculated. So, after the validation of the results of the first semester, occurred some research that was not in the plans about different ways to solve the found problems.

Another reason for the delay was the fact that parallel with the thesis existed a course related to this master. Moreover, the problem relative to the pandemic made work less productive, causing a delay in the established plans. So, in the face of these events, the schedule plan was changed, suffering some delays, as is demonstrated in Figure 7.2.

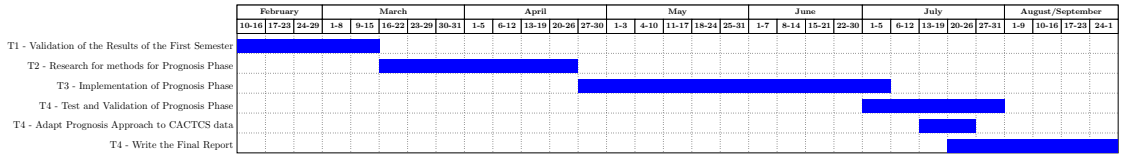

Figure 7.2: Gantt Chart of the Second Semester.

They are detailed in Appendix C.

Analysis of the Risks

For the intermediate defense, were identified two risks. However, it was suggested in this defense, add one more. So, the Risk Matrix and their impact on this work are described in Table 7.1.

Table 7.1: Risk Matrix.

Probability/Impact	Negligible	Marginal	Moderate	Critical	Catastrophic
Improbable					
Remote					
Occasional				**Risk 2**	
Probable			**Risk 1**	**Risk 3**	
Frequent					

- **Risk 1 - High Execution Time:**

  The execution time, mostly in the last part of this work, was high. However, it didn't take long more than 4 days. During these days, were made another work, like the writing of this document, so this has no huge impact on the developed work. Thus, it is important to take into account this risk in the future, in order to mitigate it using machines with more computational power or use a part of a dataset, when it could be possible.

- **Risk 2 - Poor Data Quality:**

  According to the information transmitted through the ReMAP team and with the done analysis, the conclusion was that the data has relevant information that is important to explore, in order to improve the results. This way, there was not required a mitigation action.

- **Risk 3 - Huge Number of Data:**

  The quantity of the available data was not too huge, so was possible to solve the problem, without problems of execution. Thus, it is not a risk for these datasets whereby there is no needed mitigation action.

Future Work

In the future, should be done an analysis that combines the diagnosis phase with the prognosis phase, investigating the health of a system together with the prediction of the failure, and the duration of the flight. This approach is proposed because believes that the duration of flights could influence the system's deterioration.

Also, the way that the thresholds are defined should be improved because the outcomes are dependent on them. When is chosen a minor threshold are selected too many frequencies and, consequently, too many magnitudes. However, when is chosen a major threshold, the opposite happens. So, is relevant to guarantee that, for different systems, the frequencies and magnitudes are selected in similar conditions.

Moreover, the obtained results in the diagnosis phase could be improved if there is an improvement of a combination of parameters for each Machine Learning algorithm used.

This page is intentionally left blank.

References

[1] Remap - real-time condition-based maintenance for adaptive aircraft maintenance planning. https://h2020-remap.eu. Accessed: 2020-01-02.

[2] Serdar Dalkilic. Improving aircraft safety and reliability by aircraft maintenance technician training. *Engineering Failure Analysis*, 2017.

[3] Big data in aviation - reduce costs thru predictive maintenance. https://www.exsyn.com/blog/big-data-in-aviation-predictive-maintenance. Accessed: 2020-07-15.

[4] Airbus. A statistical analysis of commercial aviation accidents 1958-2017. 2018.

[5] Yaguo Lei, Naipeng Li, Liang Guo, Ningbo Li, Tao Yan, and Jing Lin. Machinery health prognostics: A systematic review from data acquisition to rul prediction. *Mechanical Systems and Signal Processing*, 104:799–834, 2018.

[6] T. Wang, Jianbo Yu, D. Siegel, and J. Lee. A similarity-based prognostics approach for remaining useful life estimation of engineered systems. *2008 International Conference on Prognostics and Health Management*, pages 1–6, 2008.

[7] A. Saxena. Prognostics, the science of prediction. *Annual Conference of the PHM Society (PHM2010)*, 2010.

[8] Tianyi Wang. Trajectory similarity based prediction for remaining useful life estimation. 2010.

[9] Abhinav Saxena Kai Goebel, Bhaskar Saha. A comparison of three data-driven techniques for prognostics. 2008.

[10] Daniel Azevedo. *Condition-Based Maintenance for Diagnosis and Prognosis in Aircraft Systems*. Master's thesis, University of Coimbra, Coimbra, 2019.

[11] Supervised vs unsupervised vs reinforcement. https://www.aitude.com/supervised-vs-unsupervised-vs-reinforcement/. Accessed: 2020-07-14.

[12] Supervised vs unsupervised learning: Key differences. https://www.guru99.com/supervised-vs-unsupervised-learning.html. Accessed: 2020-08-05.

[13] Difference between classification and regression in machine learning. https://machinelearningmastery.com/classification-versus-regression-in-machine-learning/. Accessed: 2019-12-28.

[14] A beginner's guide to linear regression in python with scikit-learn. https://towardsdatascience.com/a-beginners-guide-to-linear-regression-in-python-with-scikit-learn-83a8f7ae2b4f. Accessed: 2019-11-27.

[15] An intuitive perspective to linear regression. https://hackernoon.com/an-intuitive-perspective-to-linear-regression-7dc566b2c14c. Accessed: 2019-11-27.

[16] J. P. Marques de Sá. *Pattern Recognition Concepts, Methods and Applications*. 2001.

[17] Support vector machine - regression (svr). https://www.saedsayad.com/support_vector_machine_reg.htm. Accessed: 2020-01-08.

[18] Understanding random forest. https://towardsdatascience.com/understanding-random-forest-58381e0602d2. Accessed: 2019-11-27.

[19] Decision trees - a simple way to visualize a decision. https://medium.com/greyatom/decision-trees-a-simple-way-to-visualize-a-decision-dc506a403aeb. Accessed: 2020-08-06.

[20] Random forest regression. https://towardsdatascience.com/random-forest-and-its-implementation-71824ced454f. Accessed: 2020-01-08.

[21] Introduction to the dft. https://www.dsprelated.com/freebooks/mdft/Introduction_DFT.html. Accessed: 2020-07-23.

[22] Explained: The discrete fourier transform. http://news.mit.edu/2009/explained-fourier. Accessed: 2020-07-23.

[23] What is a good explanation of the discrete fourier transform? https://www.quora.com/What-is-a-good-explanation-of-the-discrete-Fourier-transform. Accessed: 2020-07-23.

[24] Thamo Sutharssan, Stoyan Stoyanov, Chris Bailey, and Chunyan Yin. Prognostic and health management for engineering systems: a review of the data-driven approach and algorithms. 2015.

[25] Yaguo Lei, Naipeng Li, Szymon Gontarz, Jing Lin, Stanislaw Radkowski, and Jacek Dybala. A model-based method for remaining useful life prediction of machinery. 2017.

[26] S. Marble and B. P. Morton. Predicting the remaining life of propulsion system bearings. In *2006 IEEE Aerospace Conference*, pages 8 pp.–, 2006.

[27] Paul M. Frank. Fault diagnosis in dynamic systems using analytical and knowledge-based redundancy. *Automatica*, 26(3):459–474, may 1990.

[28] A. Hwas and R. Katebi. Model-based fault detection and isolation for wind turbine. In *Proceedings of 2012 UKACC International Conference on Control*, pages 876–881, 2012.

[29] Marcia Batista, João Pedro Malere, Cairo Nascimento Jr., and Elsa Henriques. Comparative case study of life usage and data-driven prognostics techniques using aircraft fault messages. 2017.

[30] Marcia Batista, Shankar Sankararaman, Ivo Medeiros, Cairo Nascimento Jr., Helmut Prendinger, and Elsa Henriques. Forecasting fault events for predictive maintenance using data-driven techniques with arma modeling. 2017.

[31] First neural network for beginners explained (with code). https://towardsdatascience.com/first-neural-network-for-beginners-explained-with-code-4cfd37e06eaf. Accessed: 2020-08-10.

[32] P. Wang and G. Vachtsevanos. Fault prognosis using dynamic wavelet neural networks. 2001.

[33] Yaguo Lei, Naipeng Li, Liang Guo, Ningbo Li, Tao Yan, and Jing Lin. Mechanical systems and signal processing. 2017.

[34] Márcia Lourenço Batista. *Machine Learning and Deep Learning for Prognostics and Predictive Maintenance of Aeronautical Equipment.* PhD thesis, Instituto Superior Técnico, Lisboa, 2018.

[35] Fernando Figueroa, Randy Holland, John Schmalzel, and Dan Duncavage. Integrated system health management (ishm): Systematic capability implementation. 2006.

[36] Mark Schwabacher and Kai Goebel. A survey of artificial intelligence for prognostics. 2007.

[37] Elsevier. Prognostics based on elbow point detection using neural networks. 2018.

[38] Kalman filter. https://en.wikipedia.org/wiki/Kalman_filter. Accessed: 2020-07-27.

[39] Marcia Batista, Elsa M. P. Henriques, Ivo P. de Medeiros, Joao P. Malere, Cairo L. Nascimento Jr., and Helmut Prendinger. Remaining useful life estimation in aeronautics: Combining data-driven and kalman filtering. 2018.

[40] Aircraft bleed air systems. https://www.skybrary.aero/index.php/Aircraft_Bleed_Air_Systems. Accessed: 2020-07-21.

[41] How it works: Bleed air. https://www.flyingmag.com/how-it-works-bleed-air/. Accessed: 2020-07-21.

[42] How does the boeing 787 pressurize? https://www.quora.com/How-does-the-Boeing-787-pressurize. Accessed: 2020-07-22.

[43] Inside the 747-8 new environmental control system. https://www.boeing.com/commercial/aeromagazine/articles/2012_q1/4/. Accessed: 2020-07-22.

[44] 787 no-bleed systems: Saving fuel and enhancing operational efficiencies. https://www.boeing.com/commercial/aeromagazine/articles/qtr_4_07/article_02_1.html. Accessed: 2020-07-22.

This page is intentionally left blank.

Appendices

This page is intentionally left blank.

Appendix A

Air Bleed Results

The following sections will present the remaining results that were described in Section 6.1.

Train and Test Trajectories

The next tables and figures will show the results for each situation considered. More specifically, below are the results when a model is created with the trajectories 2, 3, 4, 5, 6, 7, and 8, and then the Health Indicator is predicted for the remaining ones.

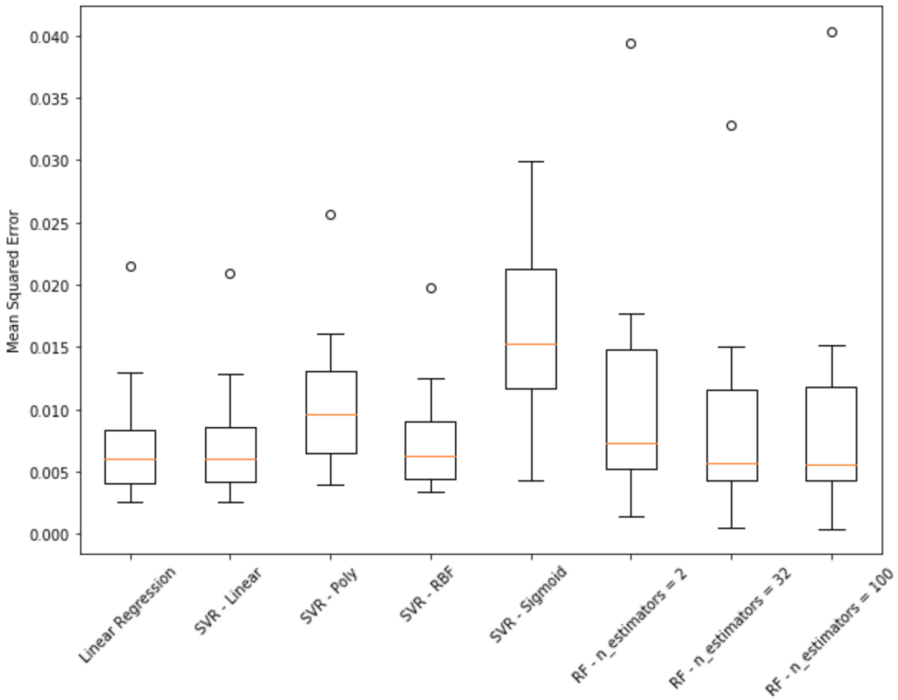

Figure A.1: Boxplot that compares MSE obtained, for each algorithm, when the model is created with Trajectory 2, for Air Bleed system.

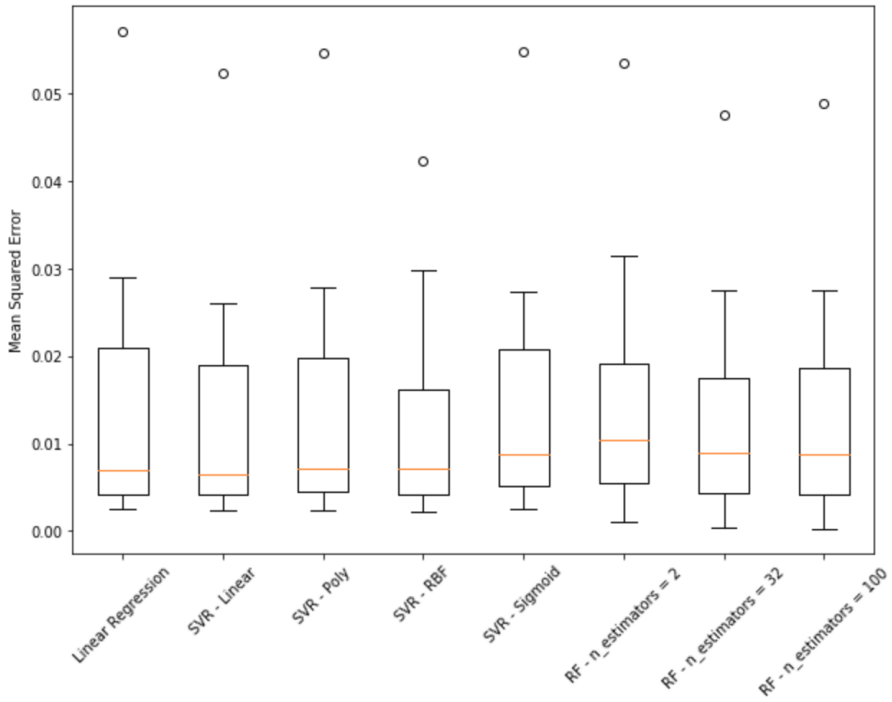

Figure A.2: Boxplot that compares MSE obtained, for each algorithm, when the model is created with Trajectory 3, for Air Bleed system.

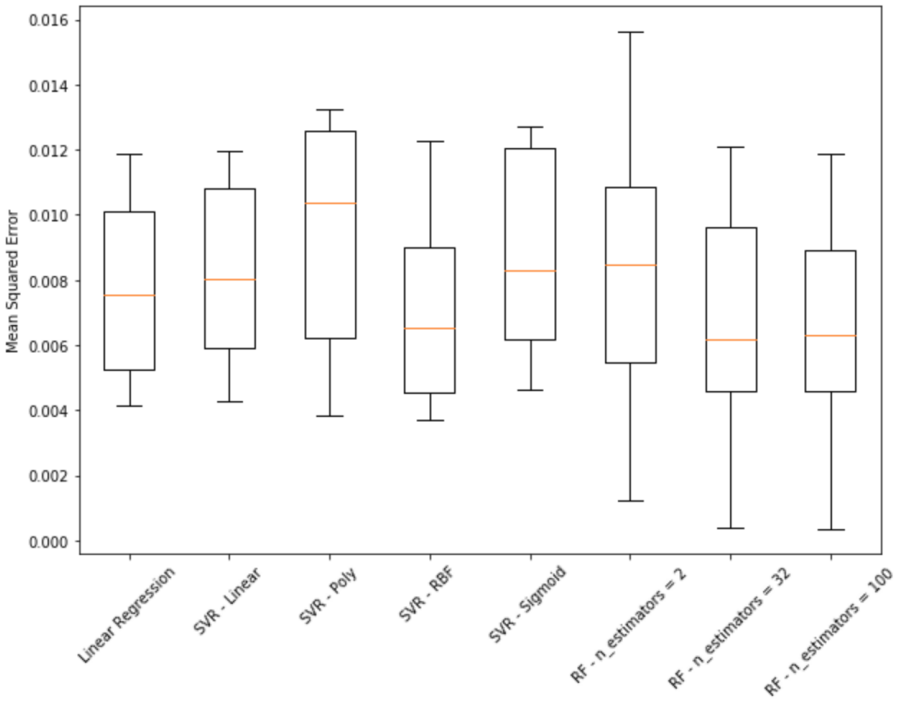

Figure A.3: Boxplot that compares MSE obtained, for each algorithm, when the model is created with Trajectory 4, for Air Bleed system.

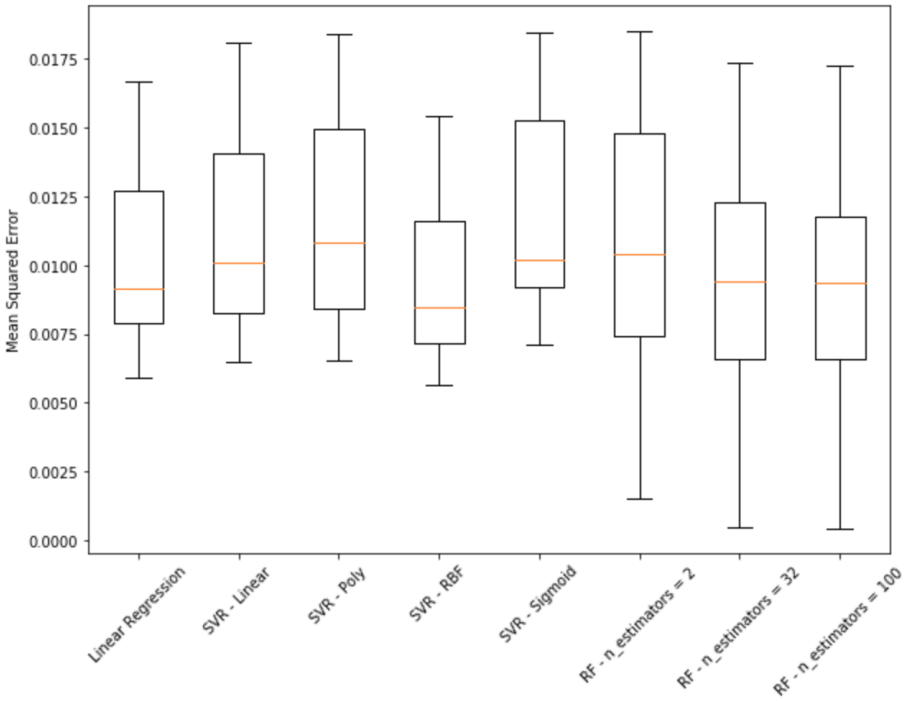

Figure A.4: Boxplot that compares MSE obtained, for each algorithm, when the model is created with Trajectory 5, for Air Bleed system.

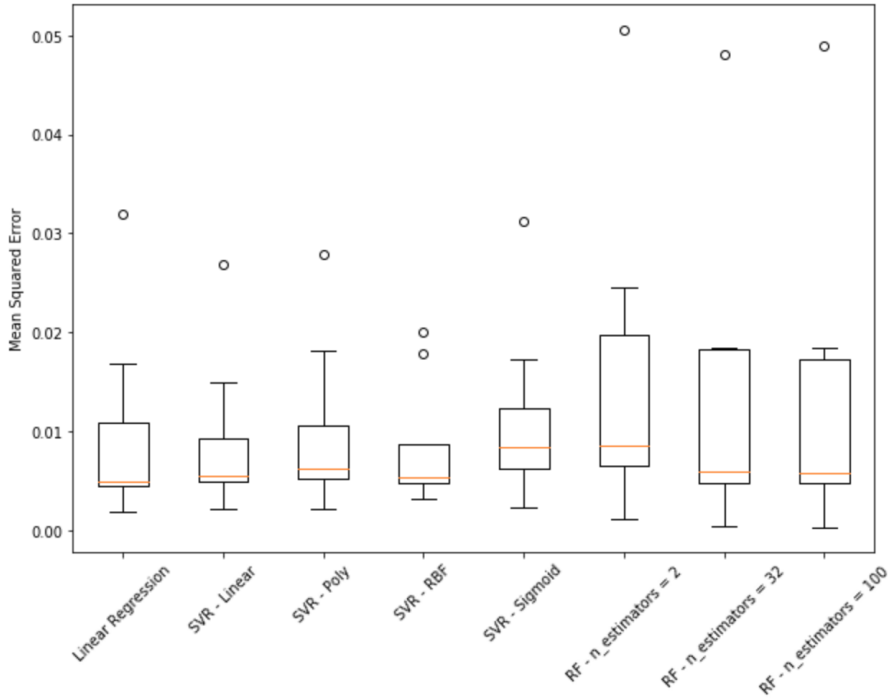

Figure A.5: Boxplot that compares MSE obtained, for each algorithm, when the model is created with Trajectory 6, for Air Bleed system.

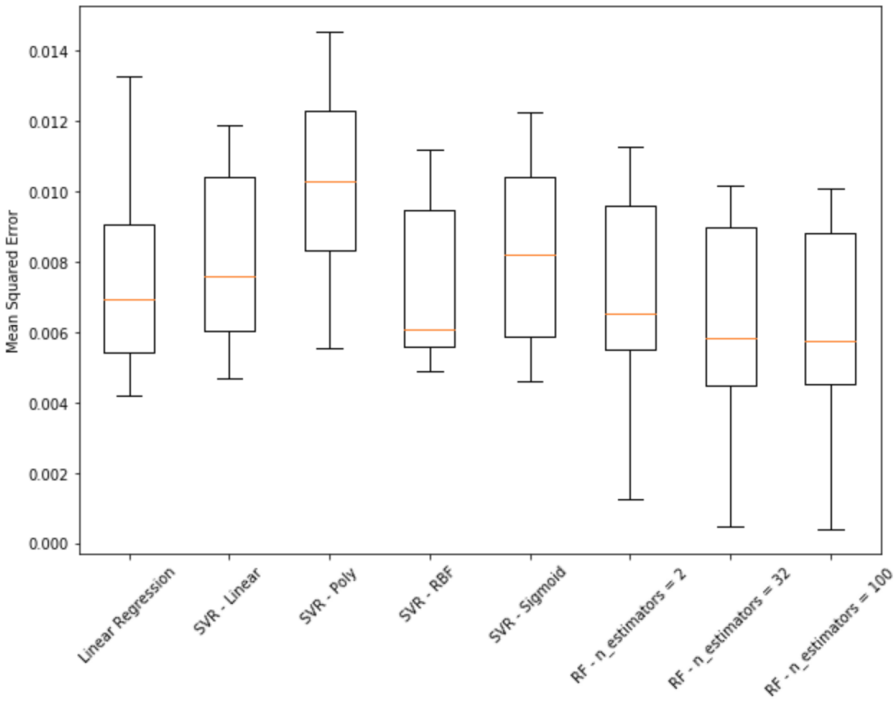

Figure A.6: Boxplot that compares MSE obtained, for each algorithm, when the model is created with Trajectory 7, for Air Bleed system.

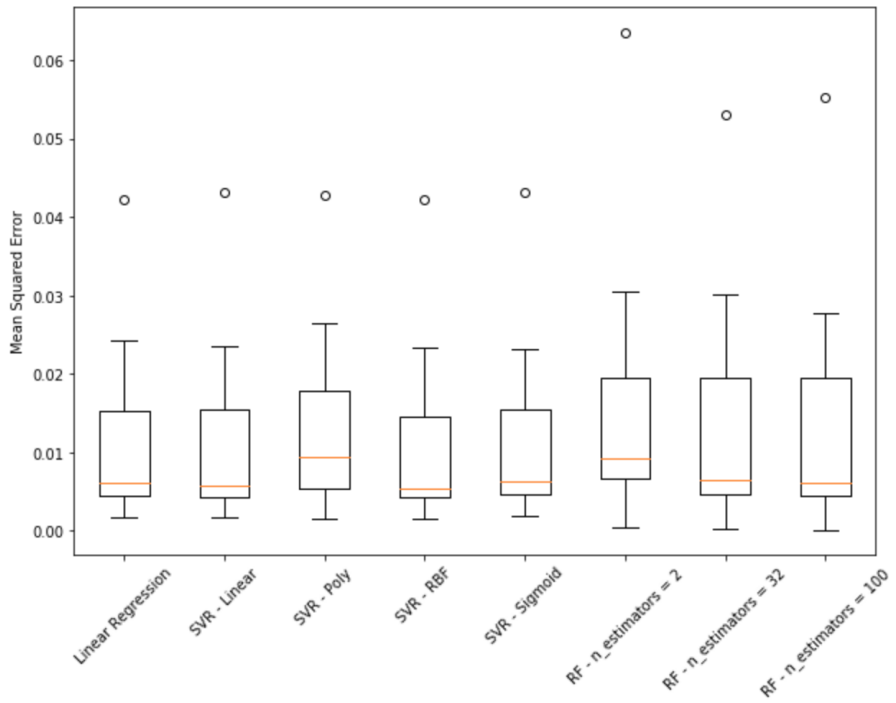

Figure A.7: Boxplot that compares MSE obtained, for each algorithm, when the model is created with Trajectory 8, for Air Bleed system.

Table A.1: MSE obtained for the model created with Trajectory 2, for Air Bleed system.

Trajectory	LR	SVR - Linear	SVR - Poly	SVR - RBF	SVR - Sigmoid	RF - 2	RF - 32	RF - 100
1	0.005	0.006	0.007	0.006	0.011	0.005	0.004	0.004
2	0.004	0.004	0.003	0.003	0.024	0.001	0.004	0.004
3	0.006	0.007	0.012	0.007	0.012	0.007	0.006	0.006
4	0.006	0.005	0.011	0.005	0.018	0.013	0.010	0.010
5	0.021	0.020	0.025	0.019	0.029	0.039	0.032	0.040
6	0.004	0.004	0.007	0.004	0.011	0.007	0.005	0.005
7	0.012	0.012	0.016	0.012	0.020	0.017	0.015	0.015
8	0.002	0.002	0.004	0.003	0.004	0.003	0.003	0.002

Table A.2: MSE obtained for the model created with Trajectory 3, for Air Bleed system.

Trajectory	LR	SVR - Linear	SVR - Poly	SVR - RBF	SVR - Sigmoid	RF - 2	RF - 32	RF - 100
1	0.004	0.004	0.004	0.004	0.005	0.006	0.004	0.004
2	0.008	0.008	0.008	0.009	0.011	0.014	0.012	0.012
3	0.003	0.003	0.003	0.002	0.004	0.001	0.001	0.001
4	0.018	0.016	0.017	0.011	0.018	0.014	0.014	0.015
5	0.057	0.052	0.054	0.042	0.054	0.053	0.047	0.048
6	0.005	0.004	0.005	0.004	0.005	0.006	0.005	0.005
7	0.028	0.025	0.027	0.029	0.027	0.031	0.027	0.027
8	0.002	0.002	0.002	0.002	0.002	0.003	0.002	0.002

Table A.3: MSE obtained for the model created with Trajectory 4, for Air Bleed system.

Trajectory	LR	SVR - Linear	SVR - Poly	SVR - RBF	SVR - Sigmoid	RF - 2	RF - 32	RF - 100
1	0.009	0.009	0.009	0.008	0.009	0.009	0.007	0.007
2	0.005	0.006	0.011	0.004	0.006	0.007	0.004	0.004
3	0.011	0.011	0.012	0.008	0.011	0.010	0.009	0.008
4	0.004	0.004	0.003	0.003	0.005	0.001	0.001	0.001
5	0.011	0.011	0.013	0.012	0.012	0.015	0.012	0.011
6	0.005	0.006	0.006	0.005	0.006	0.005	0.004	0.004
7	0.009	0.010	0.012	0.010	0.012	0.011	0.009	0.009
8	0.004	0.004	0.004	0.004	0.004	0.005	0.004	0.003

Table A.4: MSE obtained for the model created with Trajectory 5, for Air Bleed system.

Trajectory	LR	SVR - Linear	SVR - Poly	SVR - RBF	SVR - Sigmoid	RF - 2	RF - 32	RF - 100
1	0.013	0.014	0.014	0.013	0.015	0.014	0.014	0.014
2	0.009	0.010	0.011	0.007	0.010	0.012	0.009	0.009
3	0.016	0.018	0.018	0.015	0.018	0.018	0.017	0.017
4	0.005	0.006	0.006	0.006	0.007	0.007	0.006	0.006
5	0.008	0.008	0.008	0.005	0.009	0.001	0.001	0.001
6	0.009	0.009	0.010	0.009	0.010	0.008	0.008	0.008
7	0.012	0.013	0.015	0.011	0.015	0.015	0.011	0.010
8	0.006	0.007	0.007	0.007	0.007	0.006	0.006	0.006

Table A.5: MSE obtained for the model created with Trajectory 6, for Air Bleed system.

Trajectory	LR	SVR - Linear	SVR - Poly	SVR - RBF	SVR - Sigmoid	RF - 2	RF - 32	RF - 100
1	0.004	0.005	0.005	0.005	0.006	0.0080	0.005	0.005
2	0.005	0.005	0.006	0.005	0.010	0.008	0.006	0.005
3	0.004	0.005	0.006	0.005	0.006	0.007	0.005	0.005
4	0.008	0.007	0.007	0.005	0.010	0.018	0.018	0.016
5	0.031	0.026	0.027	0.020	0.031	0.050	0.048	0.048
6	0.003	0.003	0.003	0.003	0.005	0.001	0.001	0.001
7	0.016	0.015	0.018	0.017	0.017	0.024	0.018	0.018
8	0.001	0.002	0.002	0.003	0.002	0.004	0.003	0.003

Table A.6: MSE obtained for the model created with Trajectory 7, for Air Bleed system.

Trajectory	LR	SVR - Linear	SVR - Poly	SVR - RBF	SVR - Sigmoid	RF - 2	RF - 32	RF - 100
1	0.008	0.010	0.010	0.009	0.010	0.009	0.008	0.008
2	0.005	0.006	0.013	0.005	0.006	0.006	0.005	0.005
3	0.010	0.011	0.014	0.011	0.012	0.010	0.010	0.010
4	0.004	0.004	0.006	0.004	0.004	0.006	0.004	0.004
5	0.013	0.011	0.011	0.010	0.011	0.011	0.010	0.009
6	0.005	0.006	0.008	0.006	0.006	0.005	0.005	0.005
7	0.007	0.008	0.009	0.005	0.009	0.001	0.001	0.001
8	0.004	0.004	0.005	0.006	0.004	0.004	0.004	0.004

Table A.7: MSE obtained for the model created with Trajectory 8, for Air Bleed system.

Trajectory	LR	SVR - Linear	SVR - Poly	SVR - RBF	SVR - Sigmoid	RF - 2	RF - 32	RF - 100
1	0.004	0.004	0.005	0.004	0.005	0.006	0.004	0.004
2	0.007	0.006	0.015	0.006	0.007	0.011	0.008	0.007
3	0.004	0.004	0.005	0.004	0.004	0.005	0.004	0.004
4	0.012	0.012	0.005	0.011	0.012	0.015	0.015	0.016
5	0.042	0.043	0.042	0.042	0.043	0.063	0.053	0.055
6	0.004	0.004	0.005	0.003	0.004	0.007	0.004	0.004
7	0.024	0.023	0.026	0.023	0.023	0.030	0.030	0.027
8	0.001	0.001	0.001	0.001	0.001	0.000	0.001	0.001

1 Trajectory to Train and Test

The next tables and figures will show the results for each situation considered. More specifically, below are the results when a model is created with a percentage of the first instances of a trajectory and then is predicted, for the remaining trajectory, the Health Indicator.

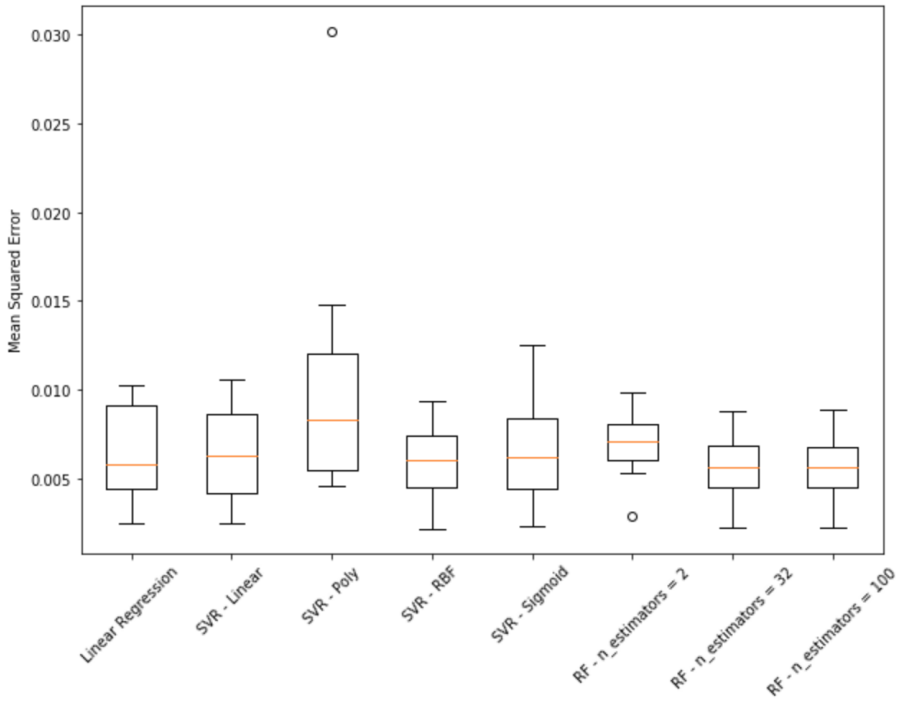

Figure A.8: Boxplot that compares the MSE obtained, for each algorithm, when the model is created with 40% of data, for each trajectory, for Air Bleed system.

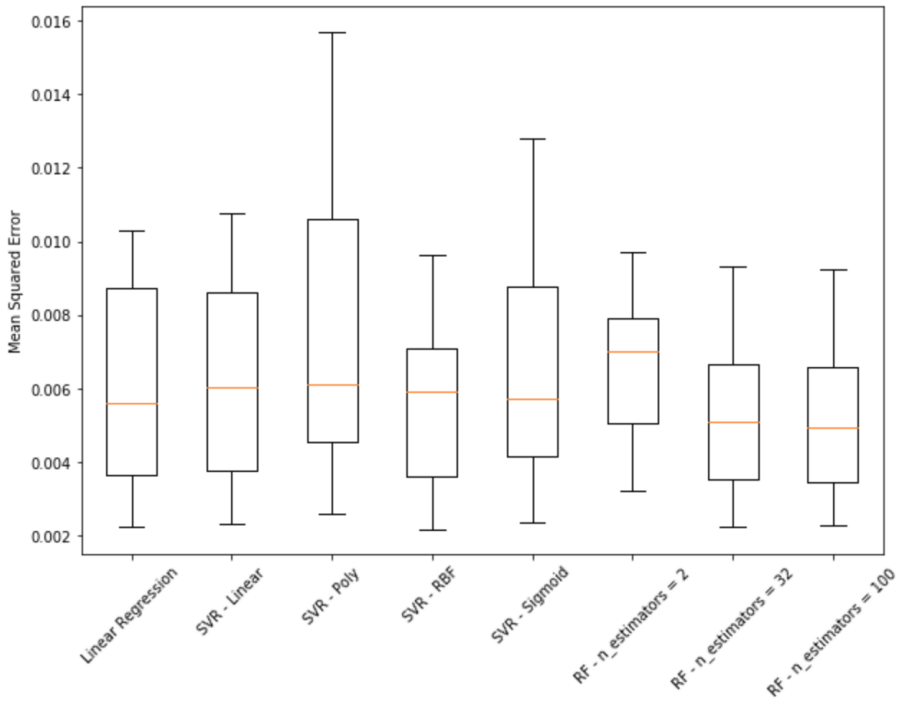

Figure A.9: Boxplot that compares the MSE obtained, for each algorithm, when the model is created with 50% of data, for each trajectory, for Air Bleed system.

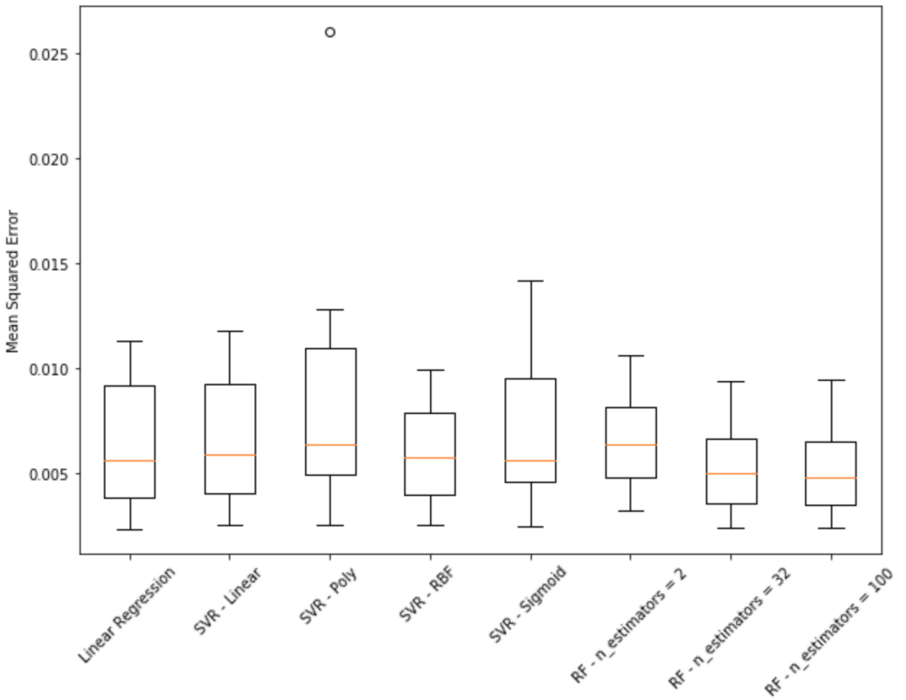

Figure A.10: Boxplot that compares the MSE obtained, for each algorithm, when the model is created with 60% of data, for each trajectory, for Air Bleed system.

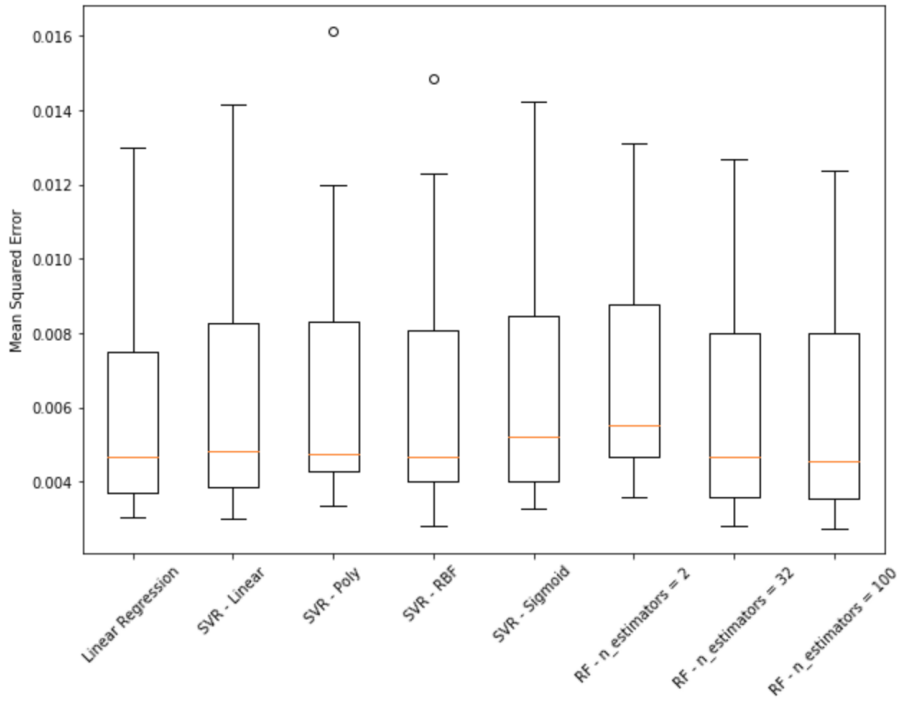

Figure A.11: Boxplot that compares the MSE obtained, for each algorithm, when the model is created with 70% of data, for each trajectory, for Air Bleed system.

Table A.8: MSE obtained with the models created, for each trajectory, for the 40% instances, for Air Bleed system.

Trajectory	LR	SVR - Linear	SVR - Poly	SVR - RBF	SVR - Sigmoid	RF - 2	RF - 32	RF - 100
1	0.005	0.005	0.005	0.005	0.005	0.007	0.005	0.006
2	0.006	0.006	0.007	0.006	0.006	0.007	0.006	0.006
3	0.003	0.004	0.004	0.003	0.004	0.005	0.003	0.003
4	0.009	0.009	0.009	0.009	0.009	0.009	0.008	0.008
5	0.010	0.010	0.014	0.009	0.012	0.009	0.007	0.007
6	0.004	0.004	0.004	0.004	0.004	0.006	0.004	0.004
7	0.009	0.008	0.011	0.006	0.008	0.007	0.005	0.005
8	0.002	0.002	0.030	0.002	0.002	0.002	0.002	0.002

Table A.9: MSE obtained with the models created, for each trajectory, for the first 50% instances, for Air Bleed system.

Trajectory	LR	SVR - Linear	SVR - Poly	SVR - RBF	SVR - Sigmoid	RF - 2	RF - 32	RF - 100
1	0.005	0.005	0.005	0.005	0.005	0.006	0.005	0.005
2	0.006	0.006	0.006	0.006	0.006	0.007	0.006	0.006
3	0.003	0.004	0.005	0.003	0.004	0.005	0.003	0.003
4	0.009	0.010	0.010	0.009	0.010	0.009	0.009	0.009
5	0.010	0.010	0.015	0.009	0.012	0.009	0.007	0.007
6	0.002	0.002	0.002	0.003	0.003	0.004	0.002	0.002
7	0.008	0.008	0.011	0.006	0.008	0.007	0.004	0.004
8	0.002	0.002	0.002	0.002	0.002	0.003	0.002	0.002

Table A.10: MSE obtained with the models created, for each trajectory, for the first 60% instances, for Air Bleed system.

Trajectory	LR	SVR - Linear	SVR - Poly	SVR - RBF	SVR - Sigmoid	RF - 2	RF - 32	RF - 100
1	0.005	0.005	0.005	0.005	0.005	0.005	0.004	0.004
2	0.005	0.006	0.006	0.006	0.005	0.007	0.006	0.006
3	0.004	0.004	0.006	0.004	0.005	0.005	0.003	0.003
4	0.01	0.01	0.01	0.009	0.011	0.01	0.009	0.009
5	0.011	0.011	0.026	0.009	0.014	0.011	0.008	0.008
6	0.002	0.002	0.002	0.002	0.003	0.004	0.002	0.002
7	0.008	0.008	0.012	0.007	0.009	0.007	0.005	0.004
8	0.002	0.002	0.002	0.002	0.002	0.003	0.002	0.002

Table A.11: MSE obtained with the models created, for each trajectory, for the first 70% instances, for Air Bleed system.

Trajectory	LR	SVR - Linear	SVR - Poly	SVR - RBF	SVR - Sigmoid	RF - 2	RF - 32	RF - 100
1	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004
2	0.004	0.004	0.004	0.004	0.006	0.006	0.004	0.004
3	0.003	0.003	0.003	0.003	0.004	0.005	0.003	0.003
4	0.006	0.007	0.007	0.006	0.007	0.007	0.006	0.006
5	0.012	0.014	0.016	0.012	0.014	0.013	0.012	0.012
6	0.003	0.003	0.003	0.002	0.003	0.003	0.002	0.002
7	0.009	0.011	0.011	0.014	0.012	0.012	0.012	0.012
8	0.003	0.003	0.004	0.004	0.003	0.003	0.003	0.003

This page is intentionally left blank.

Appendix B

CACTCS Results

The next sections will show the remaining results that were described in Section 6.2.

Train and Test Trajectories

The following tables and figures will show the results for each situation considered. More specifically, below are the results when a model is created with the trajectories 2, 3, 4, 5, 6, 7, and 8, and then the Health Indicator is predicted for the remaining ones.

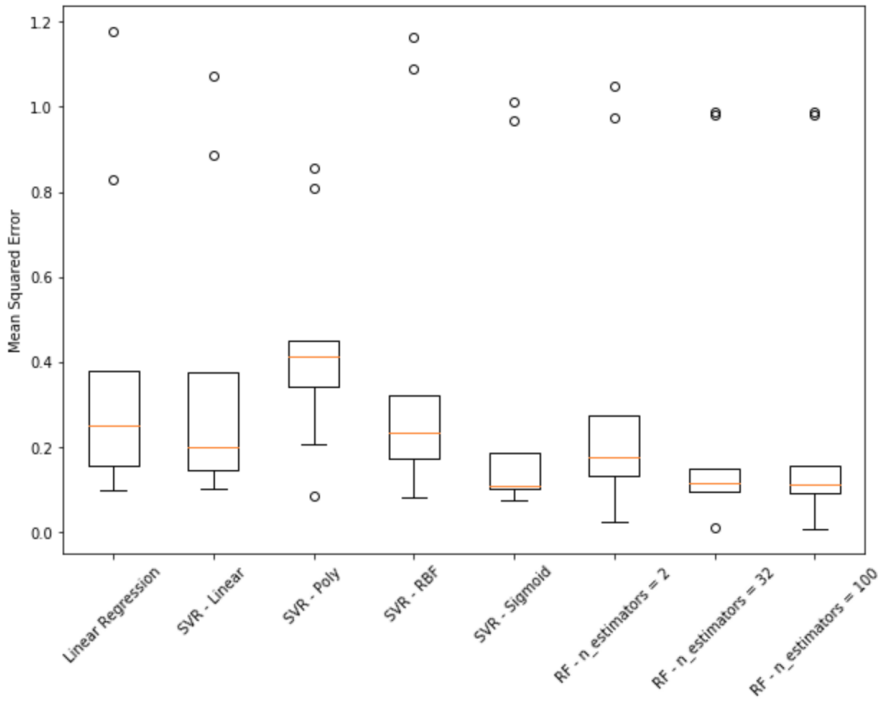

Figure B.1: Boxplot that compares MSE obtained, for each algorithm, when the model is created with Trajectory 2, for CACTCS.

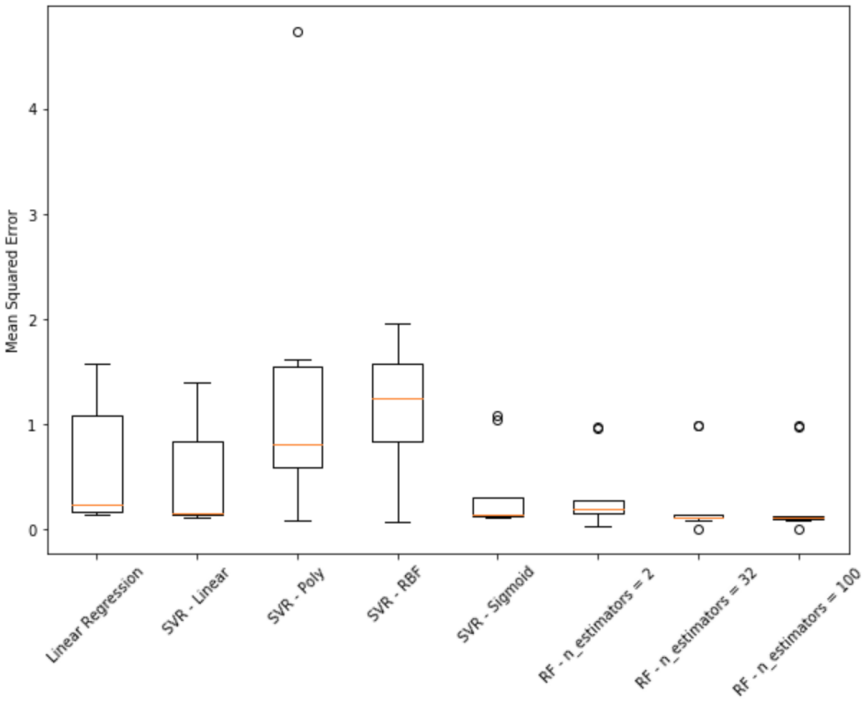

Figure B.2: Boxplot that compares MSE obtained, for each algorithm, when the model is created with Trajectory 3, for CACTCS.

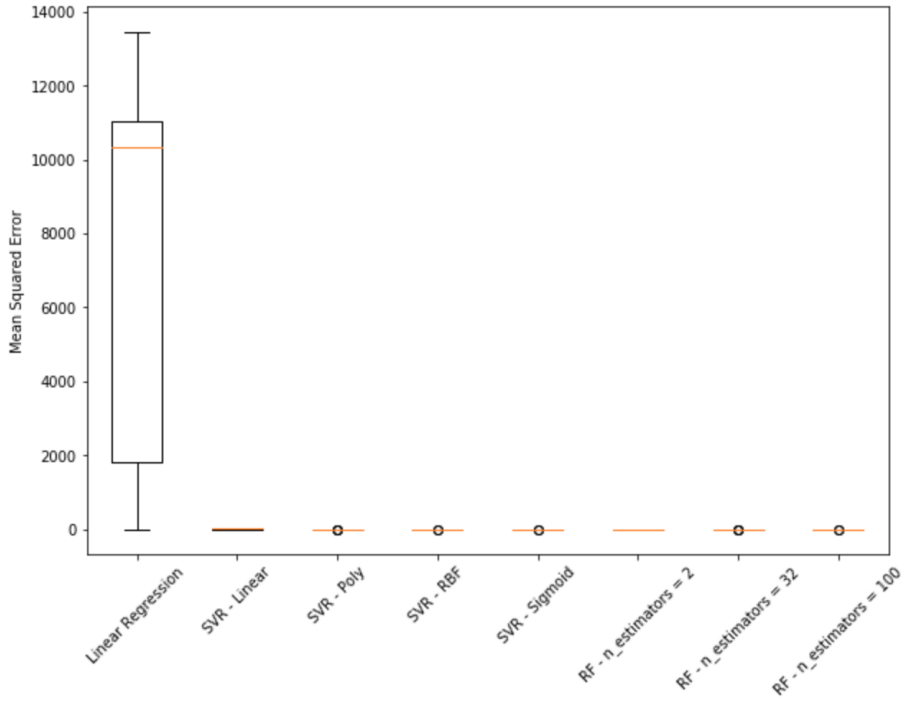

Figure B.3: Boxplot that compares MSE obtained, for each algorithm, when the model is created with Trajectory 4, for CACTCS.

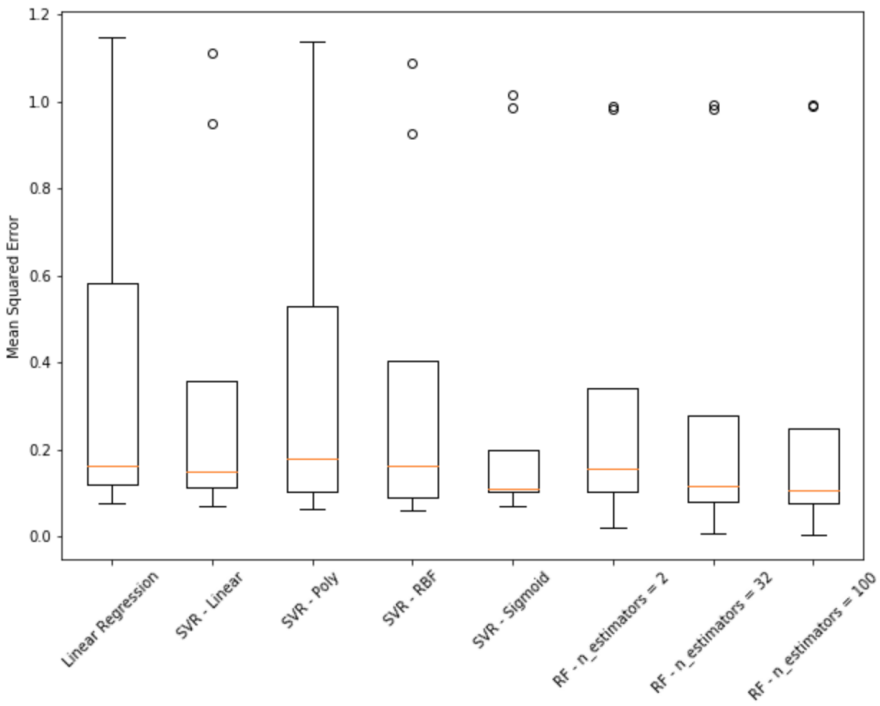

Figure B.4: Boxplot that compares MSE obtained, for each algorithm, when the model is created with Trajectory 5, for CACTCS.

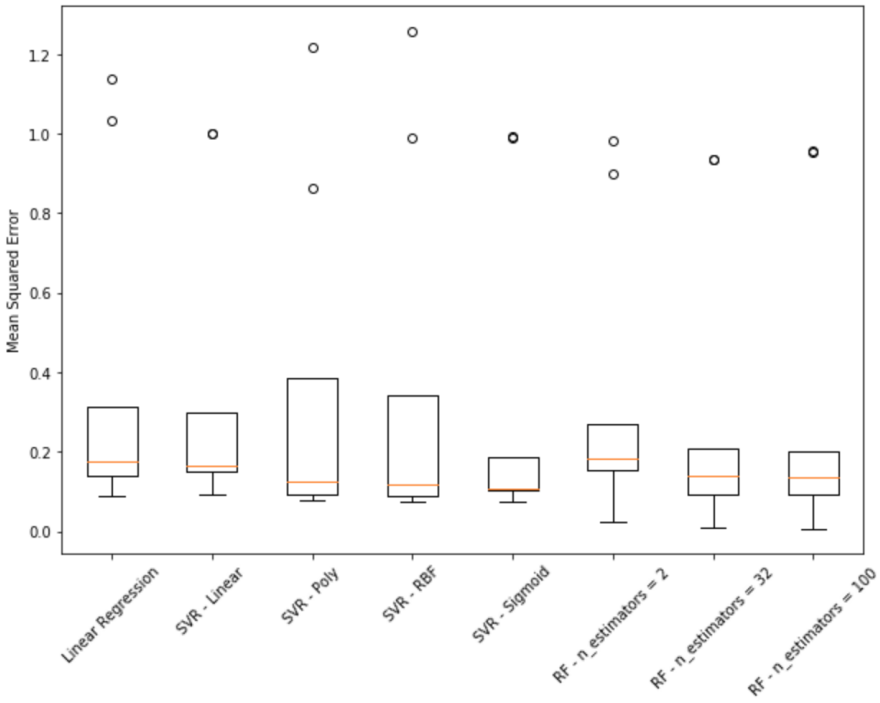

Figure B.5: Boxplot that compares MSE obtained, for each algorithm, when the model is created with Trajectory 6, for CACTCS.

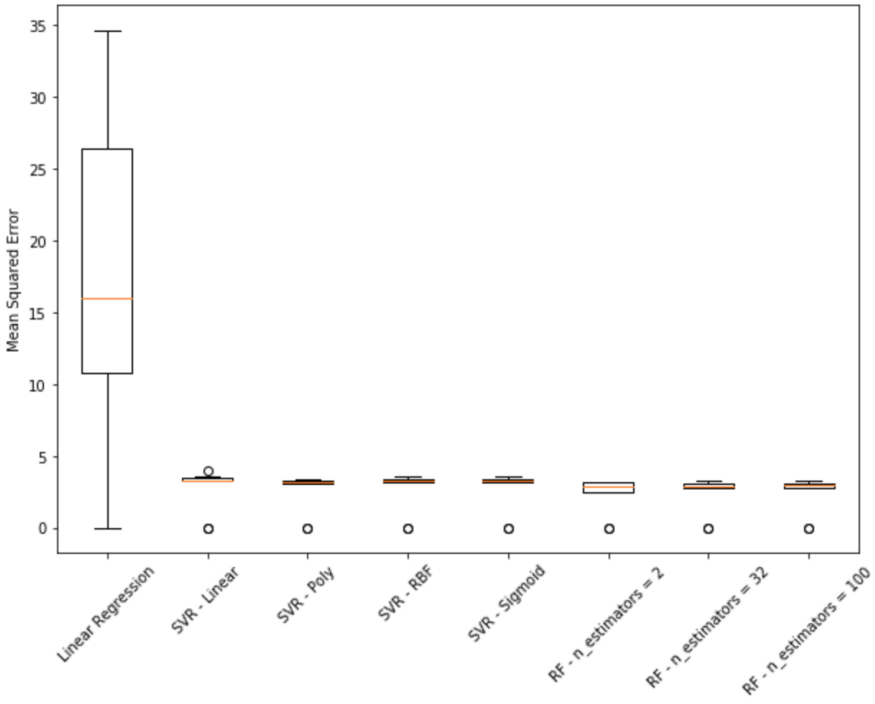

Figure B.6: Boxplot that compares MSE obtained, for each algorithm, when the model is created with Trajectory 7, for CACTCS.

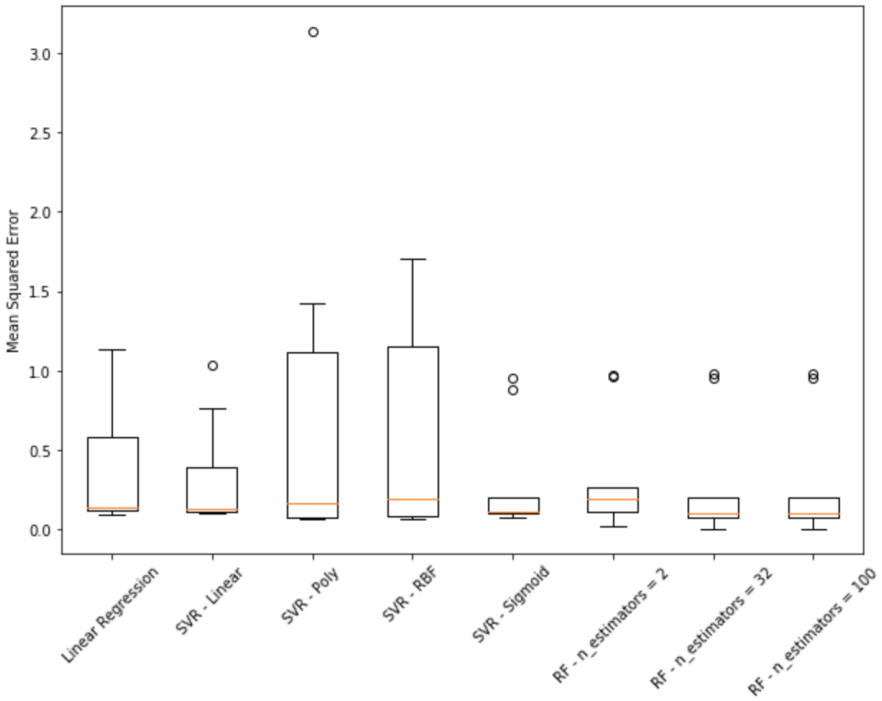

Figure B.7: Boxplot that compares MSE obtained, for each algorithm, when the model is created with Trajectory 8, for CACTCS.

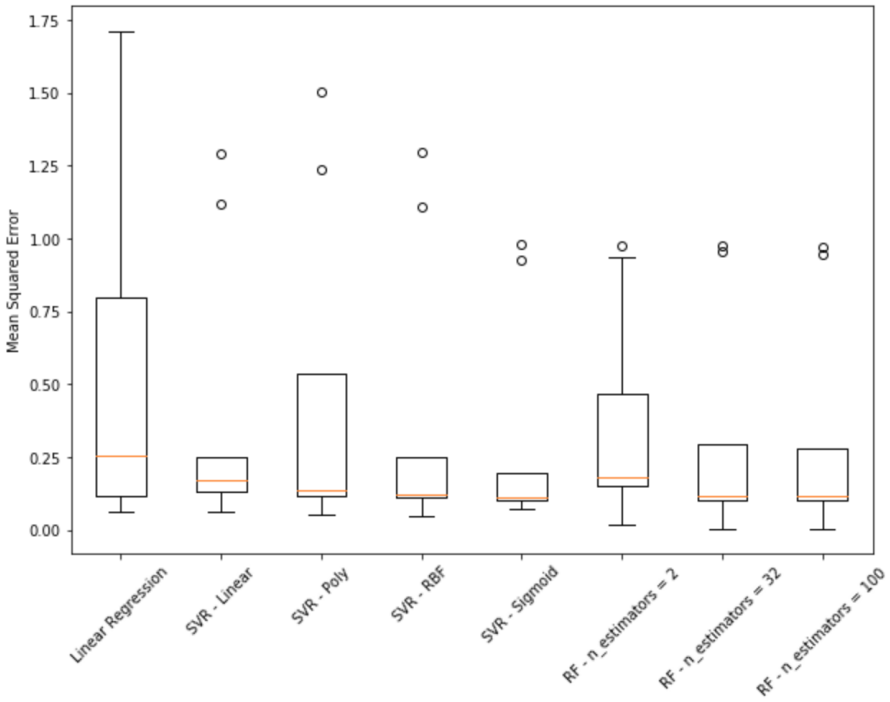

Figure B.8: Boxplot that compares MSE obtained, for each algorithm, when the model is created with Trajectory 9, for CACTCS.

Table B.1: MSE obtained for the model created with Trajectory 2, for CACTCS.

Trajectory	LR	SVR - Linear	SVR - Poly	SVR - RBF	SVR - Sigmoid	RF - 2	RF - 32	RF - 100
1	1.178	1.073	0.807	1.089	1.011	0.974	0.988	0.981
2	0.098	0.099	0.084	0.080	0.105	0.023	0.007	0.007
3	0.378	0.276	0.450	0.224	0.186	0.274	0.146	0.146
4	0.361	0.373	0.412	0.319	0.168	0.217	0.141	0.154
5	0.211	0.196	0.429	0.232	0.107	0.174	0.114	0.111
6	0.148	0.138	0.206	0.153	0.098	0.132	0.093	0.090
7	0.828	0.887	0.855	1.165	0.968	1.049	0.982	0.988
8	0.156	0.146	0.377	0.173	0.102	0.131	0.101	0.103
9	0.249	0.199	0.339	0.288	0.073	0.116	0.093	0.088

Table B.2: MSE obtained for the model created with Trajectory 3, for CACTCS.

Trajectory	LR	SVR - Linear	SVR - Poly	SVR - RBF	SVR - Sigmoid	RF - 2	RF - 32	RF - 100
1	1.578	1.400	1.546	1.544	1.088	0.955	0.980	0.979
2	0.135	0.112	0.361	0.404	0.113	0.173	0.110	0.111
3	0.151	0.157	0.080	0.076	0.169	0.024	0.008	0.007
4	1.089	0.837	4.738	1.848	0.309	0.273	0.132	0.124
5	0.237	0.171	0.814	1.240	0.144	0.233	0.135	0.126
6	0.210	0.145	0.779	1.163	0.137	0.194	0.111	0.102
7	1.381	1.246	1.611	1.581	1.046	0.977	0.984	0.983
8	0.171	0.133	0.583	0.839	0.120	0.156	0.115	0.114
9	0.236	0.146	0.942	1.955	0.111	0.120	0.089	0.086

Table B.3: MSE obtained for the model created with Trajectory 4, for CACTCS.

Trajectory	LR	SVR - Linear	SVR - Poly	SVR - RBF	SVR - Sigmoid	RF - 2	RF - 32	RF - 100
1	1349.372	0.788	0.683	0.758	0.795	1.268	0.984	0.977
2	11032.632	33.891	0.444	0.179	0.123	0.206	0.172	0.165
3	9958.635	15.480	0.355	0.164	0.195	1.549	0.240	0.255
4	0.0919	0.105	0.066	0.067	0.117	0.017	0.006	0.006
5	10991.173	25.302	0.492	0.258	0.116	0.318	0.155	0.167
6	13457.056	27.7168	0.452	0.163	0.110	0.561	0.181	0.182
7	1824.079	0.592	0.737	0.942	0.840	1.324	1.102	1.022
8	10316.339	26.603	0.414	0.279	0.118	0.331	0.153	0.149
9	11805.140	29.911	0.424	0.223	0.083	0.190	0.128	0.120

Table B.4: MSE obtained for the model created with Trajectory 5, for CACTCS.

Trajectory	LR	SVR - Linear	SVR - Poly	SVR - RBF	SVR - Sigmoid	RF - 2	RF - 32	RF - 100
1	1.148	1.112	0.793	0.925	1.014	0.990	0.983	0.987
2	0.163	0.148	0.180	0.161	0.109	0.184	0.115	0.107
3	0.581	0.359	0.295	0.263	0.198	0.342	0.279	0.248
4	0.206	0.217	0.529	0.405	0.133	0.151	0.144	0.136
5	0.091	0.094	0.064	0.061	0.104	0.020	0.006	0.005
6	0.123	0.123	0.118	0.097	0.101	0.155	0.106	0.103
7	0.886	0.949	1.138	1.088	0.987	0.982	0.993	0.992
8	0.119	0.114	0.104	0.091	0.102	0.103	0.08	0.077
9	0.078	0.070	0.065	0.062	0.071	0.086	0.065	0.062

Table B.5: MSE obtained for the model created with Trajectory 6, for CACTCS.

Trajectory	LR	SVR - Linear	SVR - Poly	SVR - RBF	SVR - Sigmoid	RF - 2	RF - 32	RF - 100
1	1.139	0.999	0.863	0.988	0.993	0.982	0.936	0.955
2	0.173	0.184	0.113	0.099	0.105	0.141	0.085	0.083
3	0.312	0.297	0.386	0.330	0.185	0.269	0.209	0.200
4	0.301	0.150	0.385	0.343	0.143	0.235	0.140	0.144
5	0.140	0.166	0.091	0.087	0.106	0.184	0.139	0.134
6	0.089	0.091	0.076	0.074	0.097	0.023	0.008	0.007
7	1.033	1	1.215	1.258	0.989	0.898	0.936	0.953
8	0.145	0.162	0.126	0.116	0.102	0.164	0.093	0.094
9	0.112	0.140	0.093	0.085	0.073	0.152	0.114	0.101

Table B.6: MSE obtained for the model created with Trajectory 7, for CACTCS.

Trajectory	LR	SVR - Linear	SVR - Poly	SVR - RBF	SVR - Sigmoid	RF - 2	RF - 32	RF - 100
1	0.0006	0.002	0.006	0.005	0.005	4e-05	2e-05	3e-05
2	26.385	3.238	3.189	3.280	3.283	2.507	2.774	2.733
3	34.665	3.555	3.385	3.544	3.564	2.906	3.101	3.102
4	10.789	3.921	3.297	3.418	3.441	3.201	2.881	2.765
5	12.213	3.435	3.292	3.380	3.376	3.166	2.834	2.974
6	27.280	3.326	3.245	3.344	3.342	2.533	2.977	3.014
7	0.0002	0.001	0.005	0.004	0.005	1e-05	0.0	0.0
8	16	3.303	3.099	3.184	3.184	3.057	3.196	3.091
9	16.349	3.261	3.125	3.223	3.216	3.116	3.239	3.274

Table B.7: MSE obtained for the model created with Trajectory 8, for CACTCS.

Trajectory	LR	SVR - Linear	SVR - Poly	SVR - RBF	SVR - Sigmoid	RF - 2	RF - 32	RF - 100
1	1.134	1.036	1.112	1.151	0.949	0.963	0.953	0.953
2	0.116	0.109	0.164	0.191	0.109	0.196	0.1	0.098
3	0.582	0.333	0.298	0.339	0.199	0.262	0.202	0.201
4	0.243	0.388	3.137	1.707	0.163	0.197	0.142	0.141
5	0.127	0.126	0.077	0.08	0.106	0.111	0.077	0.070
6	0.107	0.103	0.149	0.152	0.101	0.144	0.087	0.089
7	0.697	0.760	1.419	1.340	0.880	0.972	0.975	0.976
8	0.096	0.098	0.073	0.069	0.101	0.022	0.007	0.006
9	0.141	0.113	0.069	0.076	0.073	0.096	0.069	0.066

Table B.8: MSE obtained for the model created with Trajectory 9, for CACTCS.

Trajectory	LR	SVR - Linear	SVR - Poly	SVR - RBF	SVR - Sigmoid	RF - 2	RF - 32	RF - 100
1	1.711	1.291	1.237	1.109	0.980	0.935	0.954	0.947
2	0.256	0.173	0.137	0.122	0.111	0.180	0.115	0.115
3	0.785	0.251	0.268	0.248	0.197	0.468	0.291	0.278
4	0.799	0.215	0.538	0.221	0.124	0.203	0.148	0.137
5	0.115	0.107	0.117	0.109	0.106	0.131	0.089	0.088
6	0.198	0.156	0.119	0.114	0.101	0.148	0.110	0.106
7	1.018	1.116	1.500	1.296	0.926	0.974	0.973	0.967
8	0.108	0.131	0.106	0.099	0.103	0.148	0.101	0.100
9	0.060	0.062	0.050	0.048	0.071	0.019	0.005	0.005

1 Trajectory to Train and Test

In the following tables and figures the results will be shown for each situation considered. More specifically, below are the results when a model is created with a percentage of the first instances of a trajectory and then is predicted, for the remaining trajectory, the Health Indicator.

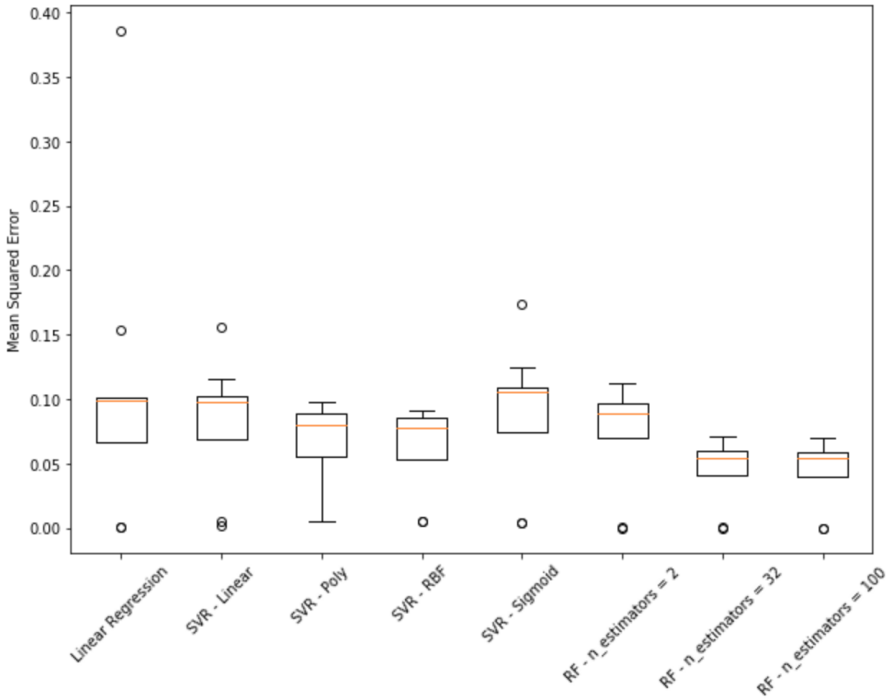

Figure B.9: Boxplot that compares the MSE obtained, for each algorithm, when the model is created with 40% of data, for each trajectory, for CACTCS.

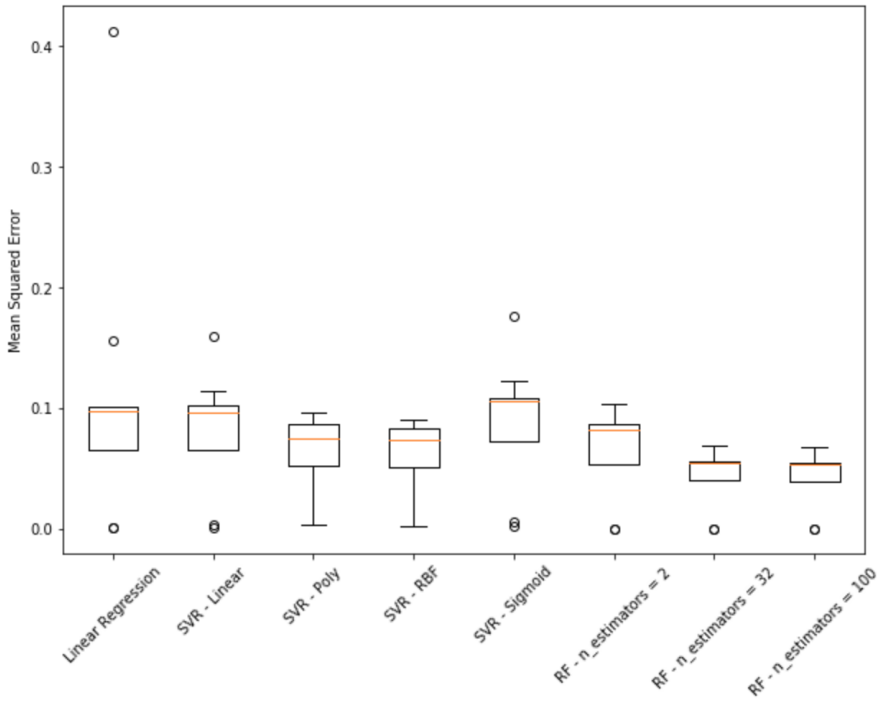

Figure B.10: Boxplot that compares the MSE obtained, for each algorithm, when the model is created with 50% of data, for each trajectory, for CACTCS.

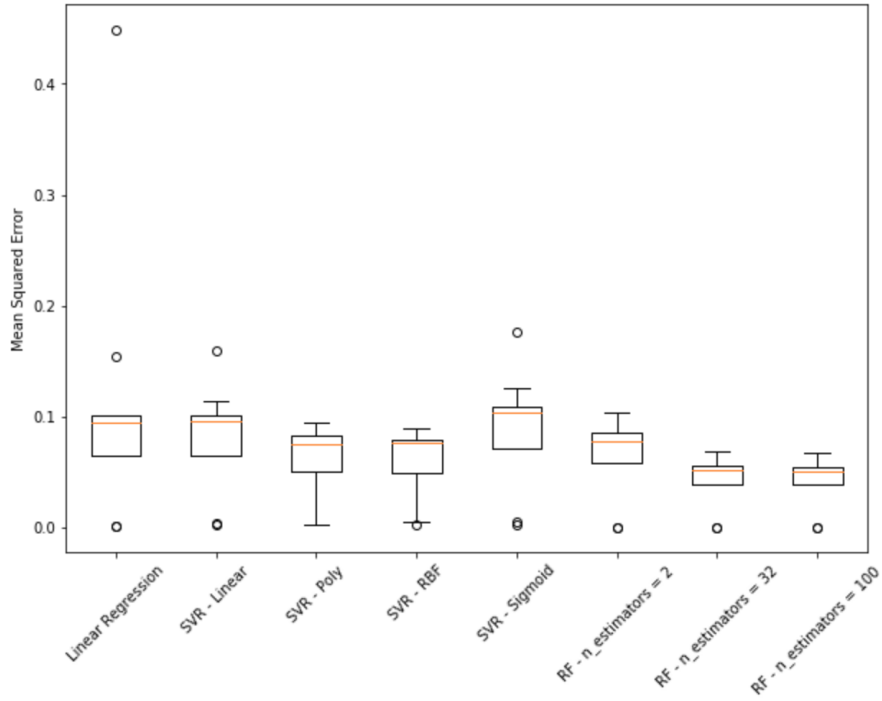

Figure B.11: Boxplot that compares the MSE obtained, for each algorithm, when the model is created with 60% of data, for each trajectory, for CACTCS.

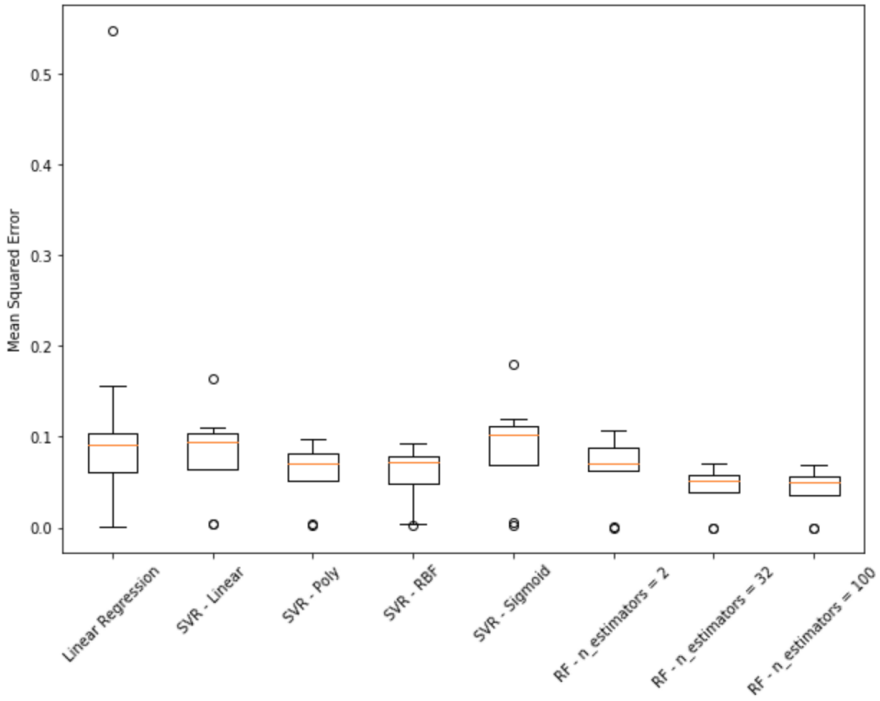

Figure B.12: Boxplot that compares the MSE obtained, for each algorithm, when the model is created with 70% of data, for each trajectory, for CACTCS.

Table B.9: MSE obtained with the models created, for each trajectory, for the first 40% instances, for CACTCS.

Trajectory	LR	SVR - Linear	SVR - Poly	SVR - RBF	SVR - Sigmoid	RF - 2	RF - 32	RF - 100
1	0.0004	0.001	0.005	0.004	0.004	0.0001	0.0001	7e-05
2	0.100	0.101	0.097	0.091	0.108	0.096	0.069	0.068
3	0.153	0.155	0.093	0.087	0.173	0.111	0.070	0.070
4	0.386	0.115	0.080	0.084	0.124	0.098	0.057	0.056
5	0.098	0.097	0.072	0.071	0.107	0.073	0.048	0.049
6	0.081	0.088	0.082	0.077	0.096	0.091	0.059	0.058
7	0.0003	0.004	0.005	0.004	0.004	2e-05	2e-05	2e-05
8	0.100	0.102	0.089	0.085	0.105	0.089	0.053	0.054
9	0.066	0.068	0.055	0.053	0.074	0.069	0.040	0.039

Table B.10: MSE obtained with the models created, for each trajectory, for the first 50% instances, for CACTCS.

Trajectory	LR	SVR - Linear	SVR - Poly	SVR - RBF	SVR - Sigmoid	RF - 2	RF - 32	RF - 100
1	0.0001	0.003	0.002	0.004	0.004	2e-05	2e-05	2e-05
2	0.100	0.100	0.096	0.090	0.108	0.102	0.068	0.067
3	0.155	0.159	0.093	0.088	0.176	0.102	0.067	0.065
4	0.412	0.113	0.074	0.077	0.122	0.086	0.055	0.053
5	0.097	0.096	0.071	0.069	0.106	0.074	0.048	0.047
6	0.085	0.086	0.077	0.073	0.092	0.086	0.055	0.054
7	0.0002	0.0007	0.003	0.002	0.002	4e-05	5e-05	5e-05
8	0.100	0.101	0.086	0.082	0.105	0.081	0.054	0.053
9	0.064	0.065	0.051	0.050	0.072	0.053	0.040	0.039

Table B.11: MSE obtained with the models created, for each trajectory, for the first 60% instances, for CACTCS.

Trajectory	LR	SVR - Linear	SVR - Poly	SVR - RBF	SVR - Sigmoid	RF - 2	RF - 32	RF - 100
1	0.0001	0.003	0.003	0.004	0.005	2e-05	2e-05	2e-05
2	0.100	0.100	0.093	0.088	0.108	0.094	0.068	0.066
3	0.154	0.159	0.092	0.088	0.176	0.103	0.067	0.064
4	0.449	0.114	0.074	0.077	0.125	0.085	0.055	0.054
5	0.094	0.096	0.070	0.068	0.105	0.071	0.047	0.046
6	0.087	0.087	0.080	0.073	0.093	0.084	0.054	0.054
7	0.0002	0.002	0.002	0.002	0.002	5e-05	7e-05	5e-05
8	0.097	0.100	0.082	0.078	0.103	0.076	0.051	0.050
9	0.064	0.064	0.050	0.048	0.070	0.058	0.038	0.038

Table B.12: MSE obtained with the models created, for each trajectory, for the first 70% instances, for CACTCS.

Trajectory	LR	SVR - Linear	SVR - Poly	SVR - RBF	SVR - Sigmoid	RF - 2	RF - 32	RF - 100
1	0.0001	0.004	0.003	0.004	0.005	0.0005	2e-05	3e-05
2	0.103	0.103	0.095	0.091	0.111	0.096	0.070	0.068
3	0.155	0.163	0.097	0.092	0.179	0.106	0.069	0.065
4	0.547	0.110	0.070	0.072	0.120	0.071	0.051	0.050
5	0.091	0.093	0.064	0.062	0.102	0.065	0.043	0.043
6	0.089	0.090	0.082	0.078	0.096	0.088	0.057	0.056
7	0.0002	0.003	0.002	0.002	0.002	4e-05	4e-05	4e-05
8	0.098	0.100	0.081	0.078	0.103	0.081	0.051	0.051
9	0.060	0.064	0.051	0.048	0.069	0.063	0.039	0.035

Appendix C

Gantt Charts

In the next two pages the Gantt Charts, which are referred to Chapter 7, will be shown.

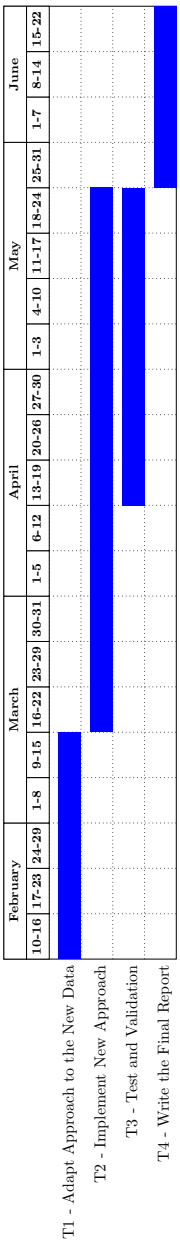

Figure C.1: Proposed Gantt Chart for the Second Semester.

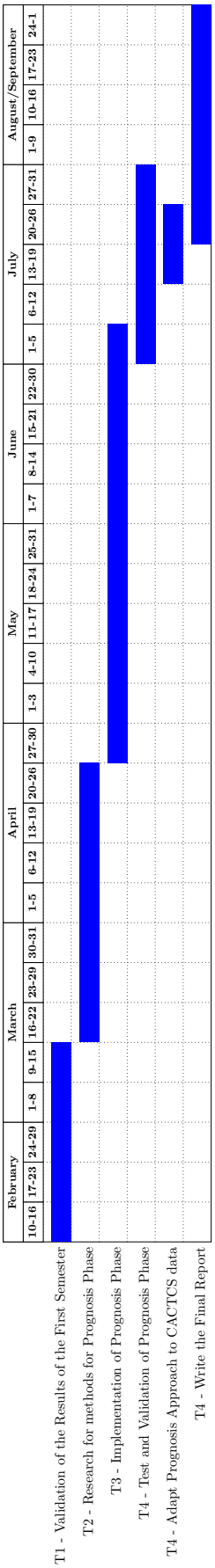

Figure C.2: Gantt Chart for the Second Semester.