UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA MECÁNICA

ANALYSIS OF AERONAUTICAL ENGINES BASED ON MACHINE LEARNING

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL MECÁNICO

ALEJANDRO FLAVIO SOTO ESPINOSA

PROFESOR GUÍA:
ENRIQUE LÓPEZ DROGUETT

MIEMBROS DE LA COMISIÓN:
VIVIANA MERUANE NARANJO
FELIPE BRAVO-MARQUEZ

SANTIAGO DE CHILE
2021

RESUMEN DE LA MEMORIA PARA OPTAR
AL TÍTULO DE INGENIERO CIVIL MECÁNICO
POR: **ALEJANDRO FLAVIO SOTO ESPINOSA**
FECHA: 2021
PROF. GUÍA: ENRIQUE LÓPEZ DROGUETT

ANALYSIS OF AERONAUTICAL ENGINES BASED ON MACHINE LEARNING

The aviation industry depends heavily on the availability of planes and their mechanical components, specially their engines. In addition, the fuel consumption of the engine is the highest operating expense for airlines.

In the last decade *Machine Learning* (ML) has had a big impact in *Prognostics and Health Management* (PHM), hence making it a suitable option for creating models that deliver interesting outputs related to the maintenance of machinery.

The main objective of this thesis is to create a model able to identify when the features of the engine are similar to the ones obtained when the engine needs a maintenance. Data is analyzed during the take off of the plane, being that during this window of time the consumption rate of fuel reaches its maximum. This model is conceived using machine learning techniques.

The specific objectives accomplished in this thesis are:

The identification of sections of interest from the original data set. In this sections an estimation of the fuel consumption of the engine is performed. Finally an anomaly detector that identifies differences between predicted and true fuel consumption is presented. This anomaly detector labels as anomalies the bigger differences between predicted and true fuel consumption.

Results show a clear relation between degradation over time and fuel consumption in the engines. For the prediction of fuel consumption 4 models were used: polynomial regression, decision trees, gradient boosting and long short term memory. The best results were obtained by gradient boosting, with an error below 0.7%.

Evaluation metrics show accuracies of 92 % for engines AIY 2 and AHD 1 and 83 % for engine AHD 2.

RESUMEN DE LA MEMORIA PARA OPTAR
AL TÍTULO DE INGENIERO CIVIL MECÁNICO
POR: **ALEJANDRO FLAVIO SOTO ESPINOSA**
FECHA: 2021
PROF. GUÍA: ENRIQUE LÓPEZ DROGUETT

ANALYSIS OF AERONAUTICAL ENGINES BASED ON MACHINE LEARNING

La industria de la aviación depende en gran medida del correcto funcionamiento de aviones y sus componentes mecánicos, especialmente sus motores. En adición, el consumo de combustible es el mayor de los gastos operacionales de las aerolíneas

En la última década el *Machine Learning* (ML) ha tenido un gran impacto en el contexto de *Prognostics and Health Management* (PHM), transformándolo en una opción adecuada para crear modelos que entreguen la información interesante en la mantención de maquinaria.

EL objetivo principal de esta tesis es crear, mediante ML un modelo que sea capaz de identificar cuando características sensadas de un motor sean similares a las obtenidas cuando este requiere de una mantención. Los datos son analizados durante el despegue de aviones, ya que en esta ventana temporal el consumo de combustible alcanza su máximo.

Los objetivos específicos de la tesis son:

Identificar trayectos de interés en el seguimiento (datos útiles). Sobre los datos seleccionados estimar el consumo de combustible del motor. Finalmente se desarrolla un detector de anomalias que identifica diferencias entre el consumo de combustible predecido y real, para luego etiquetar como anomalias las mayores diferencias entre estos.

Los resultados muestran una clara relación entre la degradación en el tiempo y el consumo de combustible en los motores. Para la predicción del consumo de combustible se probaron 4 modelos de aprendizaje de máquinas: regresión polinomial, árboles de decisión, gradient boosting y redes neuronales long short term memory (LSTM). Los mejores resultados fueron obtenidos con gradient boosting, con un error de predicción menor al 0.7%.

Las métricas de evaluación muestran un accuracy del 92 % para los motores AIY 2 y AHD 1 y del 83 % para el motor AHD 2.

Did you know that true love asks for nothing?
Her acceptance is the way we pay.
Stevie Wonder

Acknowledgments

Esta tesis es el broche que cierra mi etapa como estudiante de pregrado. Años de múltiples cambios y distintas etapas que me dieron chance de crecer, disfrutar y aprender tanto en lo personal cómo lo académico.

Agradezco a mi comisión, especialmente a quién fue mi guía en este proceso, profesor Enrique López Droguett, usted siempre tuvo la disposición para responder mis dudas, entregar ideas y criticar múltiples presentaciones de avances. También merece un agradecimiento Ignacio, quién constantemente tuvo disposición para coordinar con SKY y ofrecer criticas constructivas a mi trabajo.

Amigos, es su turno de ser mencionados. Tengo la fortuna de decir que son muchos. Los amigos de los primeros años de la U (Marcelo, Coty, Pato, Rupa, Dani, Manu, Maxi y Diego), los geo amigos (Mendo, Pilar, Negra, Jasson, Jipi, Tania, Mati, Fran...), los de mecánica (Javi, Carpa, Sara, Pablo, Rafa, Yani, Matheus, Castro, Pancho, Chopan, Tente, señora Silvia, Carlitos, Fran, ...), los que conocí en los proyectos eolian y la fanfarria, quienes están desde la adolescencia y antes (Lucho, Tatan, Violeta, 'Chiky', Diego, Magda, Gastón y Joaquin). El Dr. Quincy Jones dice que sólo se puede ser tan buen músico como buena persona se es. Le encuentro la razón y lo extiendo cualquier profesión. Ustedes, amigos queridos, me hacen mejor persona.

Este tesis esta dedicada a mis familias. Primero la que me ha inculcado valores y me ha visto crecer. Me doy el lujo de agradecer directamente a mi abuelo Marco (en parte responsable de que yo siga este camino), mis padres (Cuanta paciencia!, y décadas de excelente crianza), mi hermano Daniel y mis tíos, siempre presentes en distintas formas.

Para finalizar le agradezco a mi familia más reciente, esa dónde soy padre y pareja. Carli, que vida tan loca nos ha tocado este último par de años, sin tu apoyo dificilmente estaria escribiendo esto, gracias por tu amor e incodicionalidad. Espero tener la fortuna de compartir más experiencias juntos. Ahora le hablo a Amaro, mi maravilloso hijo. Llegaste en un momento inusual y sin embargo termino siendo el correcto. Te amo.

Gracias.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The aviation industry depends heavily on the availability and efficient maintenance of planes and their mechanical components, specially their engines.

In the last years machine learning (ML) has had a big impact in Prognostics and Health Management (PHM), hence making it an suitable option for the development of models that give relevant outputs in the context of machinery maintenance.

1.1. Motivation

With the advance of DL techniques, its success on PHM and the continuous growth of data measured by sensors in planes, the motivation of this thesis is to predict the fuel consumption of a planes, which is related to its efficiency. To accomplish the prediction a machine learning (ML) model is used for forecasting and then used as an anomaly detector between predicted and true fuel consumption data.

1.2. Objectives

1.2.1. General objective

The main goal of this thesis is to develop an anomaly detector based on ML model which predict the fuel consumption, trained with plane's data.

1.2.2. Specific objectives

The specific objectives are listed as follows:

- To identify sections of interest from the original data set.
- To train a model capable of predicting the fuel consumption.
- To develop an anomaly detection model, using the model trained and an a proper threshold

1.3. Scope

The following objectives are within the limits of this thesis:

- The delivery of interesting characteristics from the features discovered during the data exploration process.
- The justification and selection of interesting portions of data.
- The generation of an anomaly detector model that. This model must be capable of correctly identify big differences between the predicted and true data.

,

Chapter 2

Background

This section discusses the background of this thesis and is divided in two main parts. First, the specific background is presented, including relevant components and a brief history of engines and planes. Then, a general introduction to machine learning, deep learning theory and specific architectures used in this work is given.

2.1. Aviation Background

The data supplied corresponds to 3 engines from 2 aircrafts of the Sky fleet, both Airbus 319 models. These are equipped with CFM56-5B engines. Planes and engines are described in more detail below.

2.1.1. Planes

SKY Airlines fleet consists mainly of airplanes manufactured by the french company Airbus, which is the world's leading aircraft manufacturer. In specific, data provided for this thesis comes from 2 Airbus A319 planes.

Tables 2.1 and 2.2 contain key data for model A319.

Table 2.1: Aircraft weight characteristics . Original source [1]

Valor A	**Minimum [kg]**	**Maximum [kg]**
Maximum Ramp Weight (MRW)	64 400	76 900
Maximum Take-Off Weight (MTOW)	64 000	76 500
Maximum Landing Weight (MLW)	61 000	62 500
Maximum Zero Fuel Weight (MZFW)	57 000	58 500

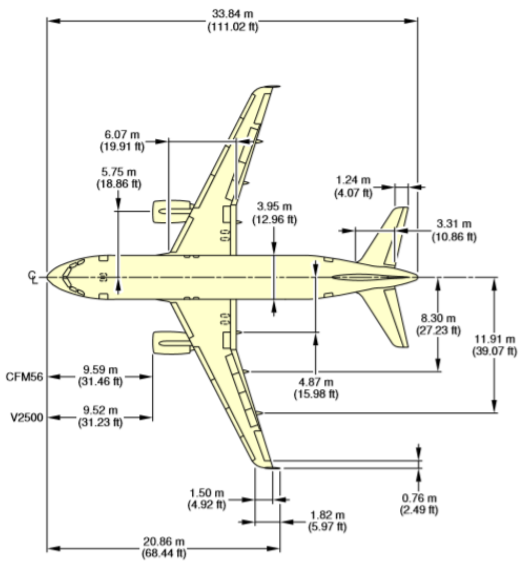
Table 2.2: Aircraft characteristics. Original source [1]

Characteristic	**Value**
Standard Seating Capacity	156 [Single-Class]
Usable fuel capacity	23 859 [L]
Pressurized Fuselage Volume	285 [m^3]
Passenger Compartment Volume	120 [m^3]
Cockpit Volume	9 [m^3]
Usable Volume, FWD CC	8.52 [m^3]
Usable Volume, AFT CC	11.92 [m^3]
Usable Volume, Bulk CC	7.22 [m^3]
Water Volume, FWD CC	10.63 [m^3]
Water Volume, AFT CC	13.91 [m^3]
Water Volume, Bulk CC	7.51 [m^3]

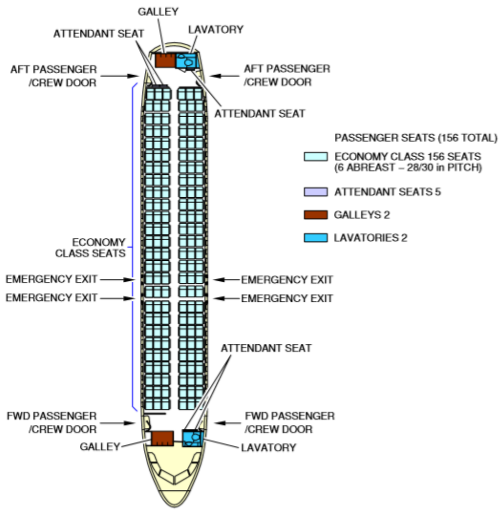
Table 2.1 has two columns (maximum and minimum) delivering a range of typical values. This is related to different variants of the plane model. Data values from table 2.2 are common to each Weight Variant.

General Aircraft Dimensions

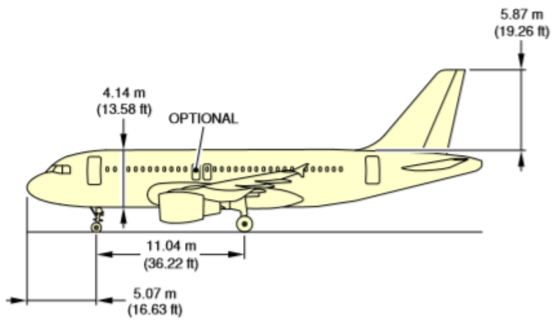
Figure 2.1 presents different views of the plane. Images (a), (c) and (d) include distances between different sections of the plane. Image (b) details seats location.



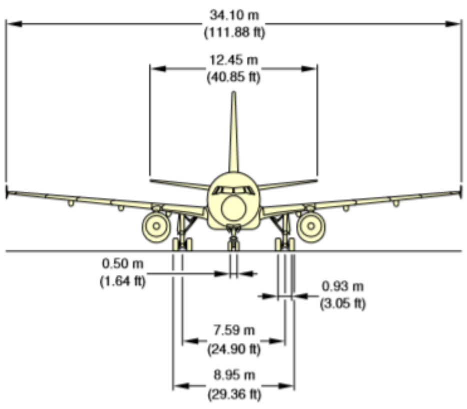
(a) Bottom view.



(b) Superior view, image specifies seats location.



(c) Lateral view.



(d) Front view..

Figure 2.1: Different views of the A319.

2.1.2. Engine

Company history

CFM International is a joint venture between Snecma (France) and GE Aviation (USA) [1]. Both companies are responsible for the production of various components, with their own assembly lines. GE is responsible for the high-pressure compressor, combustion chamber, and high-pressure turbine, while Snecma is responsible for the fan, low-pressure turbine, accessory box, and choke. The engines are assembled by GE in Evendale, Ohio, United States and by Snecma in Villaroche, France.

The CFM family has the most used engines in the commercial flights market. As of June

[1] CFM Web site Last visited October 2nd, 2020.

2016, CFM International's CFM56 fleet was the first high bypass turbofan engine family in history to achieve more than 800 million engine flight hours in service. [14]

Engine components

The specific engine used in the 319 Airbus is the CFM 56-5B. This is a high-bypass turbofan aircraft engine. The main components of this model are named as follows:

- Fan
- Low Pressure Compressor (LPC)
- High Pressure Compressor (HPC)
- Combustion Chamber (CC)
- High Pressure Turbine (HPT)
- Low Pressure Turbine (LPT)
- Exhaust Assembly (EA)

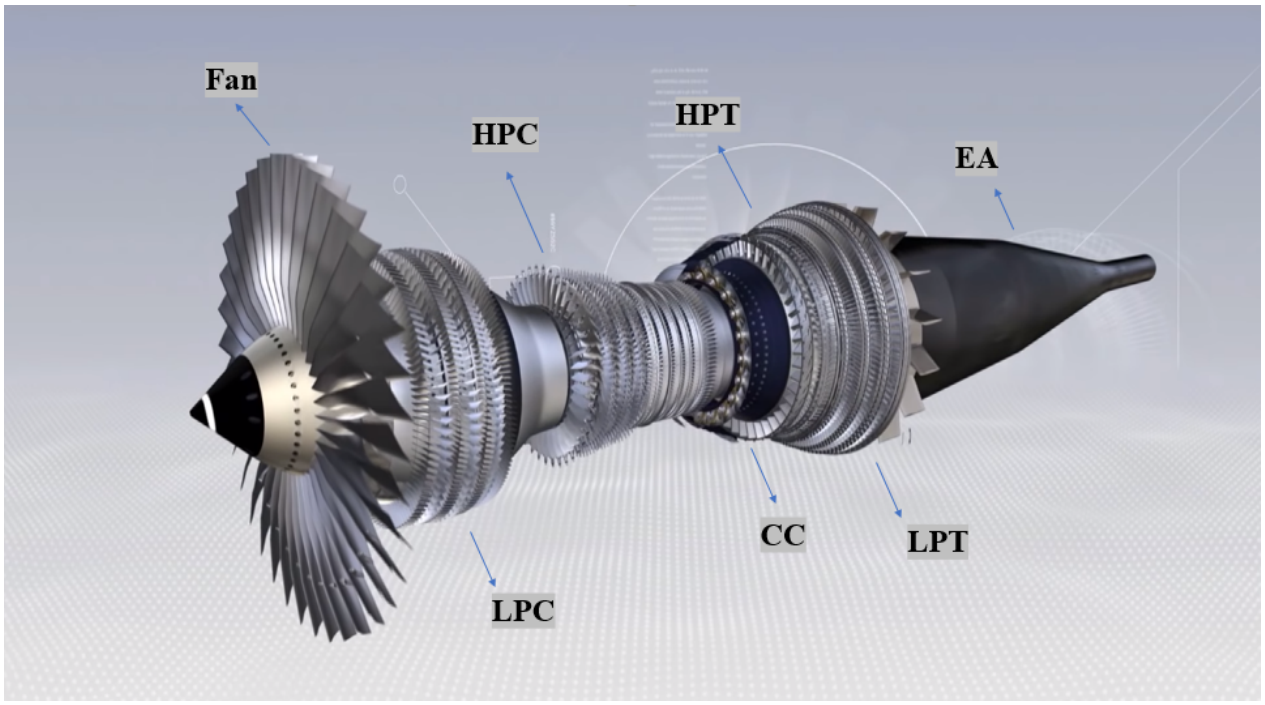


Figure 2.2: CFM 56 5B diagram with components labeled. Original image taken from Youtube video from CFM International channel.

2.1.3. Engine worflow

In a high bypass ratio engine most there are 2 flows of air. The primary flow goes through the principal components of the engine while the secondary only passes through the fan (the bypass). Proportions of air mass are 80 % to the secondary flow and 20 % to the primary.

In order to make an airplane move, a pushing force must be generated. This force is known as *thrust*, which is created by making the air accelerate between the front and the back of the engine. The detailed process of thrust generation goes as follows:

First a high diameter propeller, known as fan lets the air pass to the engine at a high rate. Then a a portion of the air goes to the low and high pressure compressors, where the pressure of the air is increased. After this jet fuel is mixed and burned with air at high pressure in the combustion chamber, creating a hot gas. This hot gas passes through high and low pressure turbines, where the pressure is reduced, making them spin. To end the process the fluid goes through the exhaust assembly.

The secondary flow is responsible for 80% of the thrust force. The temperatures reached move in the range of 450 °C in the compressors section and around 1700 °C in the combustion chamber, after the process of ignition.

Social media pages of CFM international share interesting and educational videos for a general understanding of the engine workflow[2] and are recommended as a complementary material.

2.1.4. Fuel consumption

Fuel represents 35 - 40% of Airline's operating expenses. [18]. Detail of typical expenses percentages for airlines is shown in figure 2.3.

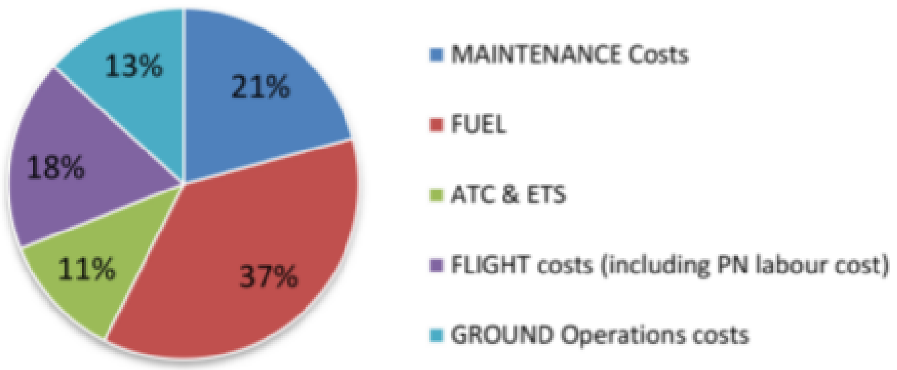


Figure 2.3: Airline's operating expenses. Original figure from [18]

Taking into account this high percentage of expenses, a main interest for airlines is to reduce fuel consumption. Since fuel consumption is related to the efficiency of the engine, maintenance must be done to keep the engine and its components operating in good mechanical conditions. This would lead to a longer life, higher efficiency and lower fuel consumption.

[2] CFM Youtube account Last visited October 2nd, 2020.

2.2. Theoretical background

2.2.1. Machine Learning

Seen as a subset of Artificial Intelligence (AI), Machine Learning (ML) is a branch of computer science that studies and develops algorithms with the capacity of improving their performance based on experience (data, training examples), without being explicitly programmed to do it. This changes the traditional programming paradigm in computation, where the rules are elaborated by hand [6].

ML algorithms can be classified according to the amount and type of supervision they get during training. There are four major categories: supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning.

Supervised learning
In supervised learning, the training data includes labels. Labels are the desired value or class to be predicted by the model.

A typical supervised learning task is to predict a target numeric value, such as the price of a house, given a set of features (location, number of rooms, year of construction, etc). This sort of task is called regression. To train the system, many labeled examples (features and labels) are needed.

Another typical task is classification. The procedure is similar, but in this case the target is normalized class. The spam filter is a typical example this: it is trained with many example emails along with their class (spam or ham), and the objective is to correctly classify new emails.

Unsupervised learning
In unsupervised learning the training data is unlabeled. Many clustering algorithms are examples of unsupervised learning, such as k-means, DBSCAN or hierarchical cluster analysis (HCA). Other applications include dimensionality reduction algorithms, such as principal component analysis (PCA).

Semi supervised learning
This type of algorithms deal with partially labeled data and are a combination of supervised and unsupervised learning. In practice, this usually means having a little amount of labelled data and larger amounts of unlabeled data.

Reinforcement learning
In this type of algorithms the learning system, called an agent in this context, can observe the environment, select and perform actions, and get rewards in return (or penalties in the form of negative rewards). It must then learn by itself what is the best strategy, called a policy, to get the most reward over time. A policy defines what action the agent should choose when it is in given situation.

In supervised learning data is splitted in *training set* and *test set*. A ML model can learn from the *training set*, automatically identifying different abstract relationships from the training set map it to a desired output. Once trained, the model is tested on new unseen data, the *test set*. The most common problems solve with ML are classification and regression tasks.

Machine learning algorithms are used in a wide variety of industries and areas. In mechanical engineering typical uses include predicting the remaining useful life (RUL) of components and classification of operation conditions.

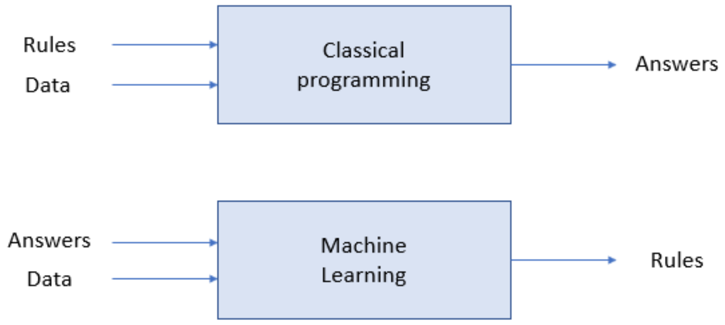


Figure 2.4: New programming paradigm.

2.2.2. Neural Networks

The central computational model used along deep learning are neural networks (NN). These receive their name from a biological inspiration, because of the way in which the neuronal connections are arranged in the cerebral cortex. Layers of junctions between neurons can be seen in figure 2.5 (a). Figure 2.5 (b) shows layers of a artificial neural network.

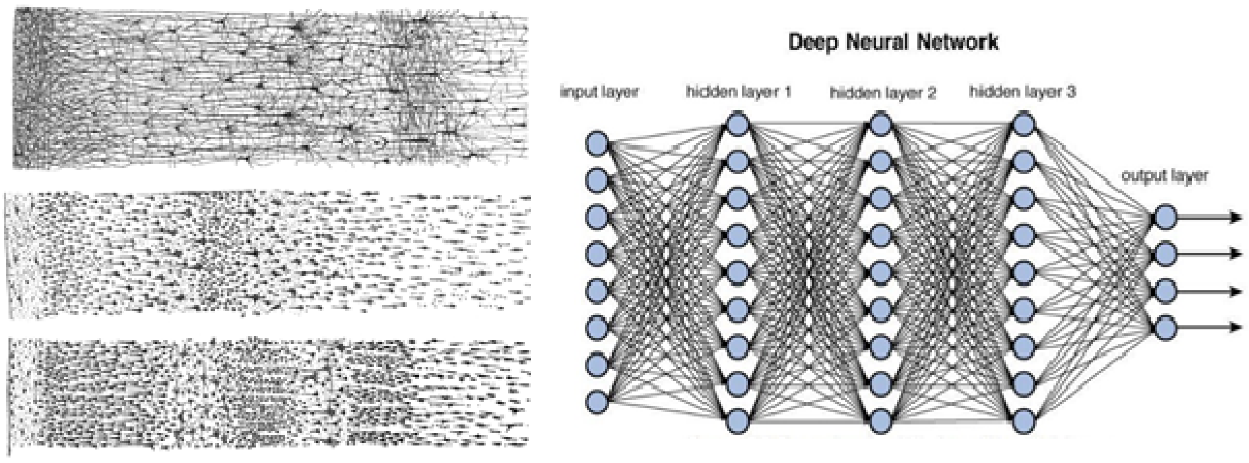


Figure 2.5: Similarities between a. Neural Cortex and b. Neural Network Architecture.

Its history dates back to the 20th century. [22] is commonly known as a seminal paper. After decades of research and the last 10 to 15 years have been characterized with great boom in the field, due to 2 fundamental aspects:

- A large volume of information available (largely thanks to the internet).
- Computational power to process information. Graphics processing units (GPU) and tensor processing units (TPU).

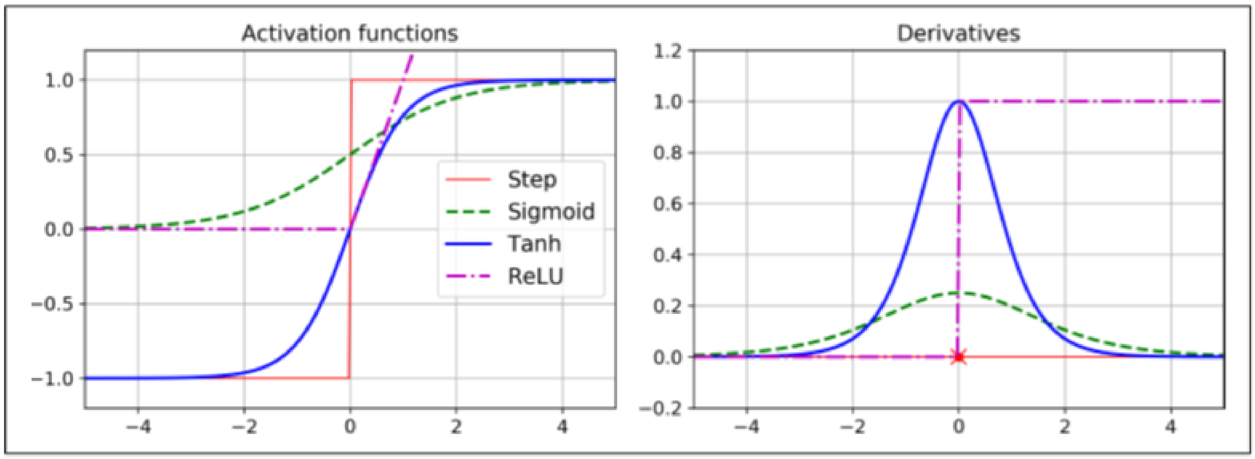


Figure 2.6: Typical activation functions and their respective derivatives. Original figure from [12]

Components of Neural Networks

The principal components are:

- **Neuron:** receives input values and produces a single output, which can be sent as an input to many other neurons. Circles in figure 2.7.
- **Weights:** values to be changed during the training of a model. By manipulating the weights which multiplied each input value the network will change its output. during training this values are changed in order to optimize en error function.
- **Connections:** establish the path of the output of each neuron. Lines of figure 2.7. Each connection carries a weights which multiplies the outputs of a previous neuron.
- **Activation function:** each neuron applies a typically non linear function to the sum of all its inputs x_i. Commonly a bias term b_i is added to the sum. Common functions are presented on figure 2.6.
- **Hyperparameters:** fixed variables given as conditions during the training of the net. Among its important to highlight:
  - **Epochs:** number of ciclyes of training. The number of times the model is trained over the train split.

- **Error function:** in order to change the values of the weights in a meaningful way an error function is need to quantify to performance of the network.
- **Learning rate:** indicates the portion of change of the weights over each iteration of the training algorithm.
- **Initialization weights:** the starting values of each weight of the network.

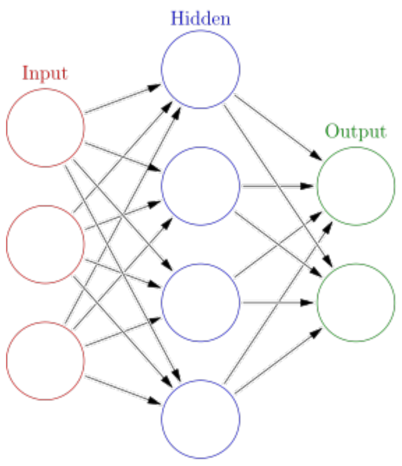


Figure 2.7: Example from an artificial neural network (ANN)

The great utility of neural networks is given by the non-linearity of the functions applied in each neuron, giving them chance to create good representations, this being, computational models with sufficient abstraction capacity to efficiently solve different tasks. [8].

Figure 2.6 shows the curves obtained with three of the most popular functions used as activation functions (Relu, sigmoid and hyperbolic tangent) and the step function.

It is worth mentioning that Rectified linear units (relu) was first proposed in [26] (year 2010) specifically in the context of AI, improving the performance of the AI models. The co author of the paper is Geoffrey Hinton, recipient of the Turing award and a pioneer in the field.[3]

Considering all the components mentioned earlier there is still a need for an algorithm which updates the weights of the network. This algorithm is the backpropagation [10]. It is used to compute the gradient of the loss function with respect to each weight of the network, using the chain rule. The calculation is performed one layer at a time, starting from the last layer (hence the name). This avoids redundant calculations of intermediate terms.

A simple explanation of the procedure is as follows:
The output prediction of a model is a non-linear function of its parameters W. $y_{pred} = f_{non-linear}(W)$ During the training the objetive is to minimize the loss function, L. The loss function quantifies the error between the output of the model (y_{pred}) and the target value (y_{true}) this is expressed in equation 2.1.

[3] Winners of the Turing award 2018 and Scholar profile of Dr. G. Hinton. He is also an author on [21], [28] and [10]

$$L_W = f_{non-linear}(W|X, (y_{true}), (y_{pred})) \tag{2.1}$$

Cost function

Backpropagation aims to modify the weights of the model W in order to minimize the loss function L. To do this first it obtains the gradient of the loss function with respect to each weight of the model, using the chain rule.

$$\boldsymbol{\nabla} L(W) = 0 \tag{2.2}$$

Gradient of loss.

A commonly used algorithm for actualization of the weights (W) is stochastic gradient descent (SGD). Equation 2.3 describes it.

$$W^{l+1} = W^l - \eta \boldsymbol{\nabla} L(W^l) \tag{2.3}$$

SGD.

Where η is the learning rate of the model (hyperparameter described on Components).

Modifications to the SGD algorithms such as Adam [15] and RMSprop have been proposed over time.

Limitations of backpropagation

- Requires activation functions to be derivable in order to update the weights. The common activation functions derivatives are shown in figure 2.6.
- Does not guarantee to find the global minimm of the loss function. However as stated in [28] "In practice, poor local minima are rarely a problem with large networks. Regardless of the initial conditions".
- As explained on [6],[5] and [12] although backpropagation does not require normalization of input vectors its recommended to apply it.

2.2.3. Deep Neural Networks

The deep neural networks (DNN) are stacked layers of artificial neural networks between the input and output layers.

2.2.4. RNN Architecture

Recurrent neural networks (RNN) are a type of neural networks architecture where connections between nodes form a directed graph along a temporal sequence. This quality makes them particularly useful for working with time series data. RNN where developed during the 1980_{ths}. On one of the seminal publications [27] ideas are a generalization of the backpropagation algorithm, proposed two years earlier on [10].

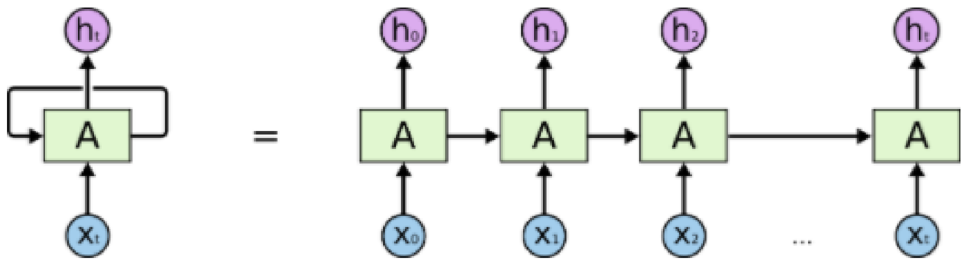


Figure 2.8: RNN diagram. Originally from Chistohper Colah’s blog

RNN have loops, allowing a previous input to be information to considering when processing a new input. Figure 2.8 shows how this loop can be seen a chain where information of input X_0 affects the information model with the next input X_1.

Several modifications of RNN have been proposed though time. Worth of mention are the Bi directional RNN, proposed in [16]. The advantage of this model resides in considering input information not only from the previous but also the following inputs from the current input. This kind of model has been used with success in translation and hand recognition.

As explained on [23] "Among the main reasons why this model is so unwieldy are the vanishing gradient and exploding gradient". Both cases are opposed extremes of the same phenom.

2.3. Models used for prediction

2.3.1. LSTM Architecture

The long short-term memory (LSTM) is a type of recurrent neural network (RNN) architecture. This type of model has the possibility to process sequences of data. It tackles the vanishing gradient problem encountered in traditional RNN architectures [29] [13] , using gates. The use of gates provides a controlled access to the memory, in contrast to traditional RNN architectures, where for every step of information added to the model, all the previous steps are also processed. This problem is also refered to as "The Problem of Long-Term Dependencies", where two cases can emerge:

- The information needed can be located in the last section of data provided to the network. An example in the context of natural language processing (NLP) is to predict the next word on the sentence: "The clouds are in the...", in this case the next word (sky) is closely related to clouds.
- The information needed can be located in an earlier section of data provided to the network. To predict to next worn on the phrase "I lived in Chile for 20 years. I speak fluent Spanish."

Experiments on RNN capacity to keep information have been done [29],[23] concluding that "The cycles in the graph of a recurrent network allow it to keep information about past inputs for an amount of time that is not fixed *a priori*, but rather depends on its weights and on the input data."

LSTM are commonly used in tasks such as translation, word on context recognition [3], handwritting recognition, speech recognition, stock pricing, among many others.

The principal components of the LSTM are the cell and gates, which interact. The first carries pieces information and interacts with new data, feed via the gates.

- The cell state (superior horizontal line i figure 2.10) carries information through the network and interacts with the gates via minor interactions (sum and multiplication, linear), this in order to be selective (diminish or augment) with the proportions of importance given to new data.
- A gate (figure 2.9) is composed by a sigmoid neural net layer, which has the non-linear component and a point wise operation. The sigmoid function outputs a number between 0 and 1 and is in charge of restricting how much of the information is given to the cell. Three gates interact with the cell.

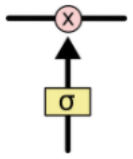


Figure 2.9: Gate of a LSTM. Originally from Chistohper Colah's blog

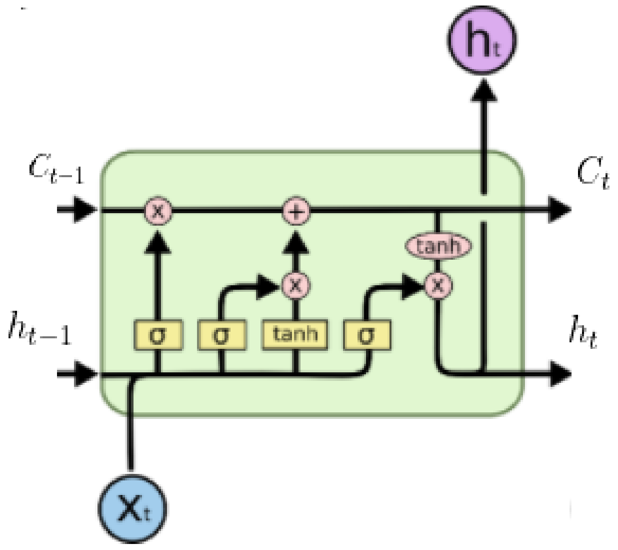


Figure 2.10: LSTM cell. Originally from Chistohper Colah's blog

LSTM equations

The LSTM can be described mainly in three sections, each one involving the interaction of the cell state with one of three gates, commonly named forget, input and output gates. Each one and its corresponding equations are described:

Forget gate: The objective is to select which part of the cell state should be considered. The sigmoid function is applied to a linear transformation considering looks at h_{t1} and x_t, and outputs a number between 0 and 1 for each number in the cell state C_{t1}. The closer the output is to 1, translates in keeping that piece of information and vice versa, a number close to 0, represents forgetting. Equation 2.4 describes the process.

$$f_t = \sigma(W_f * [h_{t-1}, x_t] + b_f) \tag{2.4}$$

Forget gate equation.

Input gate: The second step consist of selecting what new information is to be added to the cell state. To achieve this a two step process is done, first, through the same mathematical process as in equation 2.4, a the information to be added is picked, then via the application of a tanh layer a vector of new candidate values $(\widetilde{C_t})$is created. This to components are multiplicated and their result is added to the cell state. This has two parts. First, a sigmoid layer called the “input gate layer” decides which values we'll update. Next, a tanh layer creates a vector of new candidate values, C t, that could be added to the state. In the next step, we'll combine these two to create an update to the state. Equations 2.5 and 2.6 describe the process.

$$i_t = \sigma(W_i * [h_{t-1}, x_t] + b_i) \tag{2.5}$$

Input gate equation, part 1.

$$\widetilde{C_t} = tanh(W_i * [h_{t-1}, x_t] + b_c) \tag{2.6}$$

Input gate equation, part 2.

With the forget and input gate updated, C_t state cell is created via linear operations described in equation 2.4.

$$C_t = f_t * C_{t-1+i_t*\widetilde{C_t}} \tag{2.7}$$

Cell state actualization.

Output gate: The third step decides what will the output be. Filtered versions of the cell state and components $[h_{t-1}$,$x_t]$ are involved. A sigmoid layer decides which parts of the cell state are going to be outputed. Then, the cell state goes through a tanh and is multiplied it by the output of the sigmoid gate, this way the output only considers the parts selected. Equations 2.8 and 2.9 describe the process.

$$\sigma_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{2.8}$$

Output Gate. Part 1.

$$\sigma_o = \sigma_t * tanh(C_t) \tag{2.9}$$

Output Gate. Part 2.

Since the data provided is a time series, this type of model is selected.

Complementary and highly recommended material for a deeper understanding of LSTM can be found at Class on RNN and LSTM from the NLP course, University of Chile and Colah's blog, post 'Understanding LSTM Networks' .

2.3.2. Polynomial regression

Polynomial regression is a form of regression analysis. In this type of model the relationship between the independent variable x and the dependent variable y is modelled as an nth degree polynomial in x.

Equation 2.10 shows an example of this.

$$y = \alpha_0 + \alpha_1 x_i + \alpha_2 x_i^2 + \alpha_3 x_i^3 + ... + \alpha_n x_i^n \tag{2.10}$$

Polynomial regression.

Where:

- i represents the number of features used for the polynomial regression.
- n represents the number of degrees for the polynomial regression.
- α represents the parameters optimized during fitting.

2.3.3. Decision trees

Decision trees are a popular model because they are easy to interpret. They are constructed by means of the recursive division of the data set into subsets through if-then questions. During adjustment, the training set is divided into groups according to the attributes of the instances, with the objective of achieving the greatest class uniformity within each one. For this, the algorithm chooses an attribute and a threshold of that attribute from which the division of the instances is made.

There are three types of nodes, explained below:

- Root node: initial group that includes all the training instances. From this node are generated divisions.
- Decision node: it corresponds to an intermediate node, that is to say, that receives a subdivision of instances generated previously and that in turn divides this group, delivering two subdivisions.
- Leaf node: corresponds to a terminal node, which only receives a previous subdivision. Ideally a sheet should group instances of a single class, which does not necessarily occur. For being a terminal node, in classification a sheet assigns the class to the instance by virtue of the most abundant class.

The data groups present in a decision tree are represented graphically as a node.

The model in a decision tree is built using an induction algorithm which builds from the top (root node) to the bottom (leaf node). During training, an impurity criterion is used to choose the nodes. When instances of more than one class are grouped together in a node, the node is said to be impure. Two commonly used measures of impurity are the Gini index and information theory entropy. Equations 2.11 and 2.12 describe these indexes.

$$Gini(S) = 1 - \sum_{i=1}^{n} (\frac{|S_i|}{|S|})^2 \tag{2.11}$$

Gini equation.

$$Entropy(S) = -\sum_{i=1}^{n} \frac{|S_i|}{|S|} log_2(\frac{|S_i|}{|S|}) \tag{2.12}$$

Information theory entropy index equation.

Where:

- S: Set of training instances.
- S_i: Set of training instances belonging to the class c_i.

The goal is to find an attribute that divides the data set into subsets of the highest possible purity. This means that when dividing a node the attribute selected must minimize the average impurity of the separation. Equation 2.13 details calculation of average impurity.

$$Impurity(S, A) = -\sum_{t} \frac{|S_i|}{|S|} Impurity(S_t) \tag{2.13}$$

Mean impurity.

Where:

- S_t: Subsets of S obtained when applying attribute A as a division criterion.

2.3.4. Gradient boosting

Gradient boosting is a machine learning technique which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees.

Hyper parameters include:

- Depth of trees: maximum number of levels in each tree. Higher values are more likely to overfit.
- Number of estimators: number of decision trees used in the model.
- Learning rate: determines the step size at each iteration while moving toward a minimum of a loss function.

2.4. Regularization techniques

Neural networks provide non linear relationships between input and output data. During the training of the network weights associated to each neuron are changed thought an optimization algorithm aiming to improve its performance on training data. However, after a certain number of epochs of training the network could learn patterns exclusive to the training data, losing the generalization capacity over new data. This phenomenon is known as *overfitting* and several techniques have been developed to fight it.

A common technique for regularization, which aims to avoid overfitting of the NN during training is dropout. First proposed in [21], this technique does a random drop of connection between units. The input of dropout is an hyperparameter, p which is a value between 0 and 1. This value represents the probability of a node in a layer being removed. Autors recommend p = 0.5 after developing experiments.

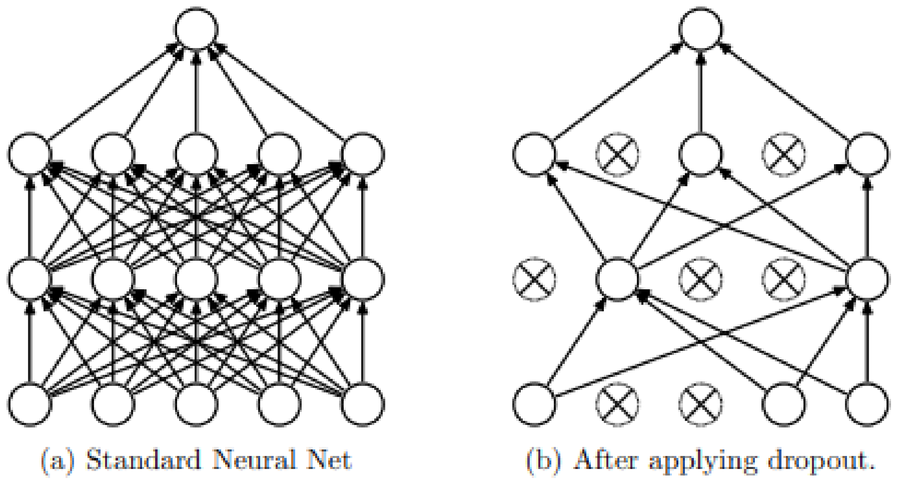


Figure 2.11: Dropout application diagram. Originally from [21]

2.5. Metrics for evaluation

The first problem solved in this thesis corresponds to a supervised regression task. The objective is to predict a numerical real value (engine's fuel consumption) from other related features. To evaluate the performance of a regression model, the following metrics are commonly used:

Root Mean Square Error (RMSE):

Equation 2.14 shows the formula used to calculate the RMSE.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{y}_i)^2} \tag{2.14}$$

RMSE formula.

Where:

- n is the total of observations.
- y_i is the target value.
- $\widehat{y}_i$ is the predicted value.

Mean Absolute Error (MAE):

The expression in equation 2.15 shows the related formula.

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |y_i - \widehat{y}_i| \tag{2.15}$$

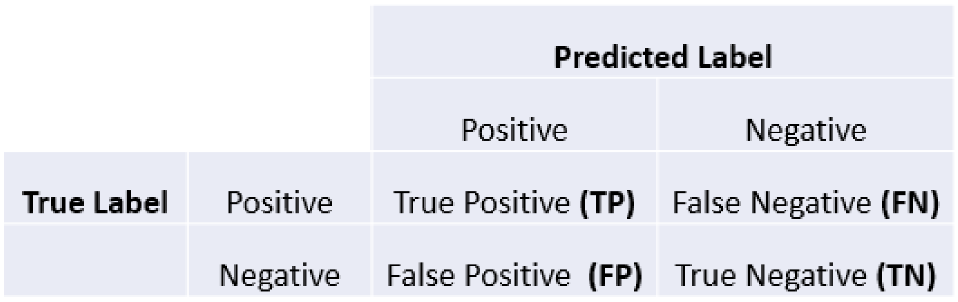
MAE.

The notation is the same as in equation 2.14.

Although similar results are obtained using MAE and RMSE, its worth mentioning that the second provides a higher penalization for greater differences between y_{true} and y_{pred}.

The same losses presented above (RMSE and MAE) are commonly used as loss functions in regression tasks. For this thesis the loss function used during the training of the LSTM models was MAE.

The second problem solved solved is a classification task. For the evaluation of a classification model, the most common metrics used are described below.



		Predicted Label	
		Positive	Negative
True Label	Positive	True Positive **(TP)**	False Negative **(FN)**
	Negative	False Positive **(FP)**	True Negative **(TN)**

Figure 2.12: Confusion matrix in the general case. Own elaboration.

Figure 2.12 describes the four possible cases of predictions for a binary problem. Based on the names of these cases, the metrics are described as follows.

Accuracy:

The accuracy corresponds to the fraction of correct classifications made by the model with respect to the total number of predictions. Equation 2.16 describes the calculation calculation of accuracy.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.16}$$

Accuracy.

Precision:

Precision indicates the percentage of true positives within all instances classified as positive by the model. This metrics is more robust against unbalanced data. Equation 2.17 describes the calculation of this metric.

$$Precision = \frac{TP}{TP + FP} \tag{2.17}$$

Precision.

Recall:

Recall, also kown as sensitivity indicates represents the percentage of positive instances that the model classifies as such. This evaluation method is highly relevant when the cost of one class is much higher than the other, as is the case in the aviation industry. Equation 2.18 describes the calculation of recall.

$$Recall = \frac{TP}{TP + FN} \tag{2.18}$$

Recall.

Receiver operating characteristic curve (ROC curve):

A receiver operating characteristic curve (ROC curve) is a graphical plot that illustrates the performance of a binary classifier system as its discrimination threshold is varied. Figure 2.13 presents an example of this curve.

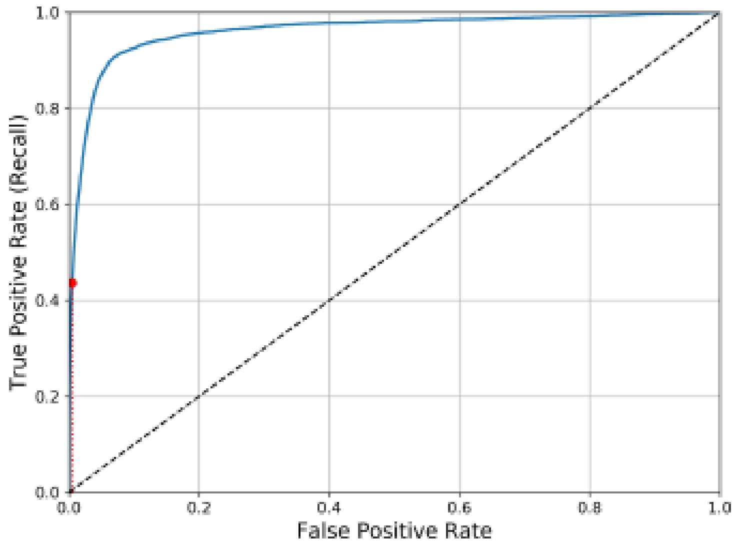


Figure 2.13: ROC curve originally from [12].

Recall/prercision curve:

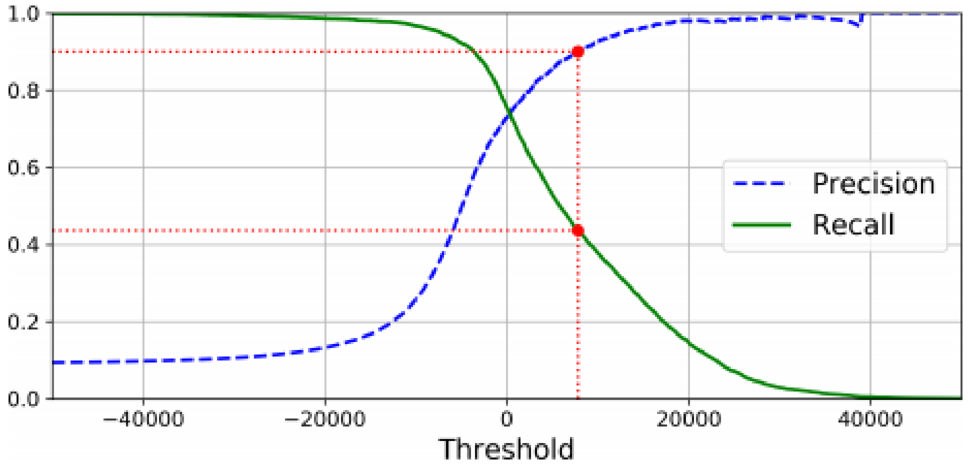


Figure 2.14: Recall/precision curve curve originally from [12].

The recall/precision curve illustrates the value of the metrics precision and recall at different thresholds. Figure 2.14 presents an example of this curve.

ROC and recall/prercision curves can be used to select a proper threshold.

Chapter 3

Data description

There are 3 data sets containing information from 2 months and 16 days of continuous measures. Each one is a time series dataset. Information comes from 3 engines, where two of them are from engines 1 and 2 of the plane AHD. The remaining DS comes from the second engine of the plane AIY.

Five sensors provide the data used in this thesis. These are:

- Altitude (feet over the sea).
- Fuel mass flow rate (Kg/hour).
- Exit temperature (°C).
- High rotor speed (%).
- Low rotor speed (%).

Image 3.1 shows a diagram indicating the location of sensors.

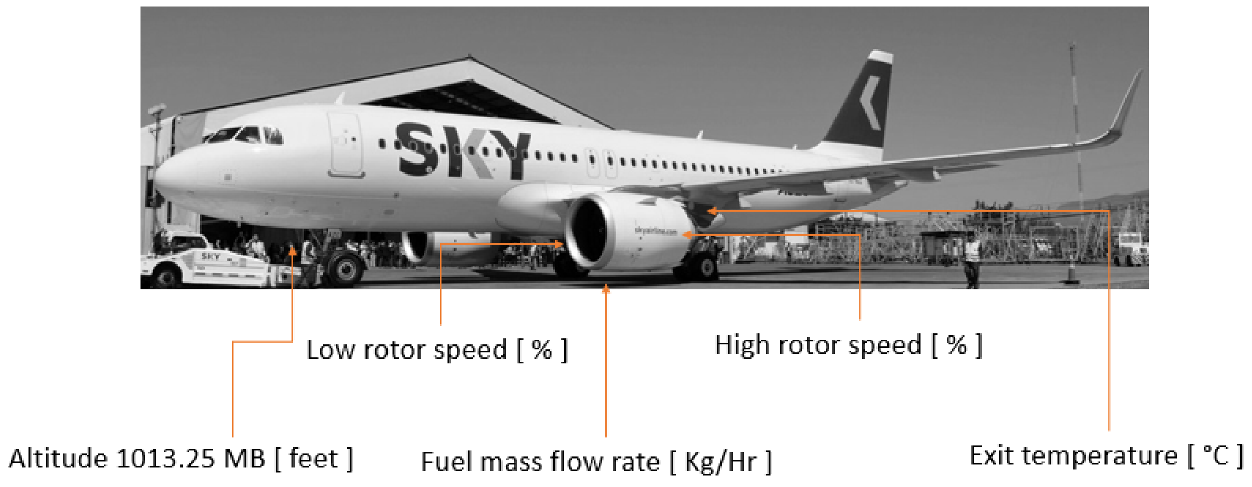


Figure 3.1: Diagram with the general location of sensors.

The altitude measurement is the same for both engines of the plane AHD. The data provided for the rotors speed (N1 and N2) is given in percentages according to the maximum suggested by design.

3.1. Data Samples

Examples of measurements collected during a flight interval are shown in Figure 3.2 from figure (a) to (e).

3.2. Ranges

The frequency acquisition is 1 hertz for all the data. The time lapse is of 2 months and 16 days for the three engines, from August 13^{th} to October 29^{th}, 2019. As the data is obtained from continuous measures and the planes fly around several airports, there are different starting points for different take offs, each one involving different external conditions (weather, altitude, weight load).

Specifications for ranges in each feature:

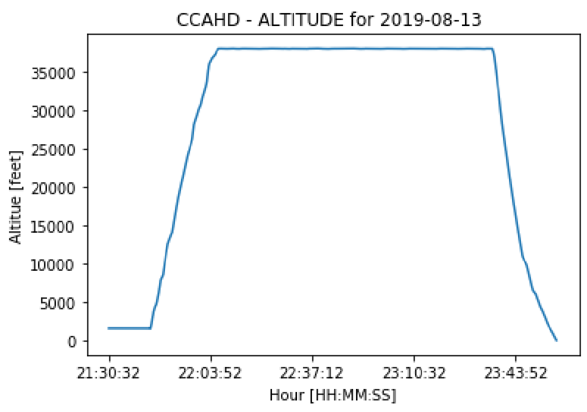
- Altitude: 0 to 39.000 [ft].
- Fuel mass flow rate: 0 to 3.900 [Kg/hour].
- Exit temperature: 0 to 850 [°C].
- High rotor speed: 0 to 100 [%].
- Low rotor speed: 0 to 100 [%].

The planes are mainly used between 7 AM and 12 AM.

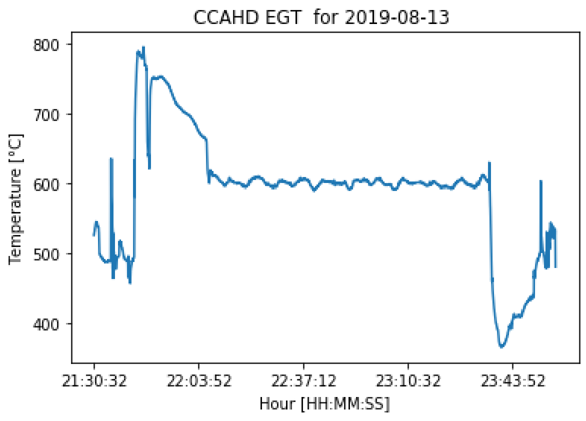
3.3. Measurement Failures

During most of the nights sensors are disabled and tend to show abnormal data. However, since this periods of time are not used, this doesn't become a problem. Chapter 4 gives details. about the data selection.

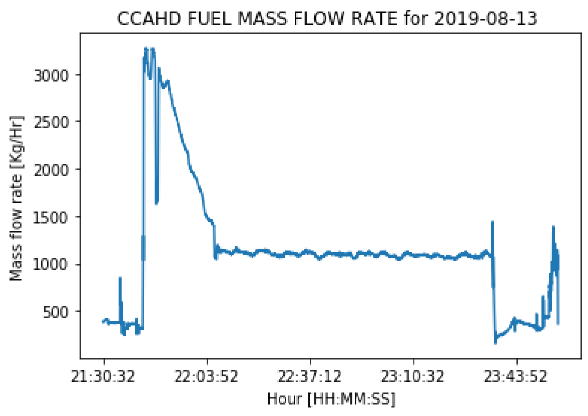
In between flight intervals, sometimes measures have sudden changes only in one of features, as seen in the plot f of figure 3.2. This behaviour is attributed to failures in the sensor during the measurement or transmission of the data.



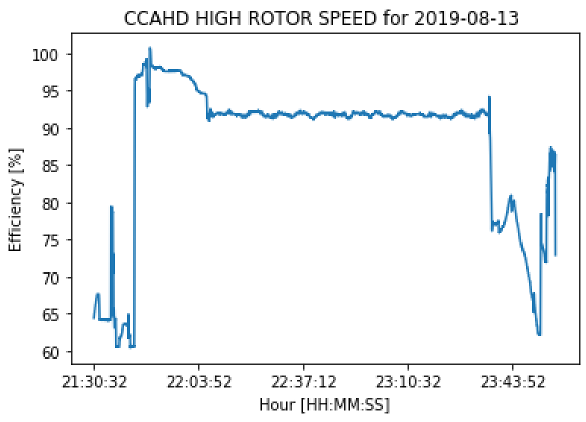
(a) Altitude.



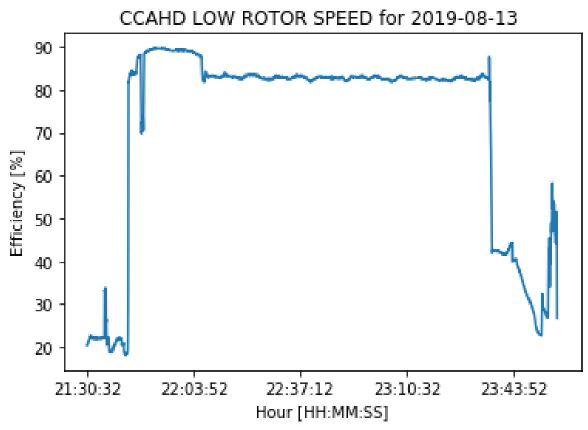
(b) EGT.



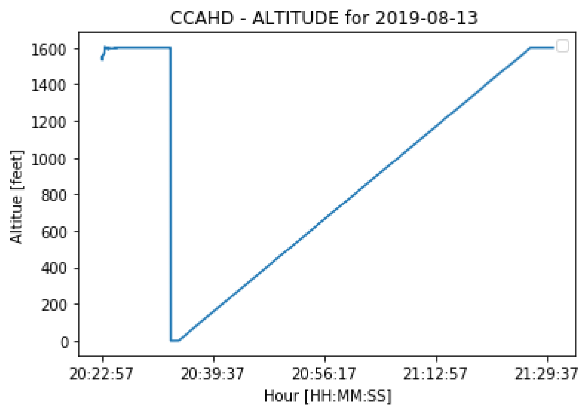
(c) Fuel mass flow rate.



(d) High rotor speed efficiency.



(e) Low rotor speed efficiency.



(f) Measurement failure on the altitude sensor.

Figure 3.2: Sensors samples during a flight.

Chapter 4

Methodology

This thesis is developed as following:

- First, a data exploration phase, searching for interesting patterns in the global whole DS.
- Secondly, the data has to be divided, keeping sections of interest for further processing. For this purpose 2 algorithms are developed and tested: The first one aims to identify the beginning and end of a flight. The latter identifies a subsection of interest in the sections obtained with the first algorithm. For the remainder of this thesis this subsection of interest will be called take off.
- Once a portion of the data is selected and obtained, the third step is to process it. A feature of interest is selected as target (fuel consumption). Following the standard approach independent train and test splits are selected and normalized (using constants obtained from train data). Temporal windows are applied to data and then it is feed it to a LSTM neural network architecture. The model aims o predict the fuel mass flow rate having as an input the remaining features (EGT, Altitude, high and low rotor's speed). This problem is defined as a supervised regression task, i.e., the prediction of a real numeric variable.
- Finally the model is used as an anomaly detection model, considering a threshold that identifies abnormal differences between true and predicted data.

4.1. Literature review

During the development of this work the main topics studied where engine health monitoring, deep learning (DL) for anomaly detection, and aviation software simulation.

Additional topics included backpropagation, dropout, RNN and LSTM to include the mathematical formulations and historical line in the development of algorithms used in this work.

4.2. Neural network architecture

This type of NN architecture is selected because of its ability to work with time series and advantages over traditional RNN architectures.

The model implemented follows uses the following architecture:

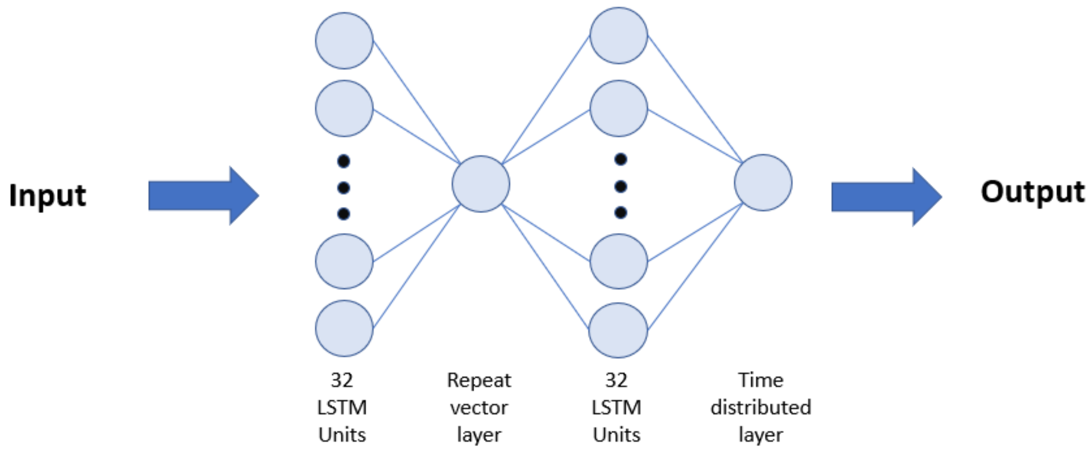


Figure 4.1: Diagram with the general location of sensors.

Details of the architecture include:

- Total number of parameters: 13.600
- Regularization techniques: dropout with a factor of 0.2 between after layers 1 and 4.
- Windowing: the network is trained with batches of window sizes equal to 20 seconds. The temporal distance between batches is of 16 seconds.

4.3. Training of the model

4.3.1. Data separation

The standard protocol working with time series of machinery is to separate train and test data with no overlap between them and a clear separation between both clusters.

Figure 4.2 shows a plot with data labeled according to 3 classes, data not selected, a train portion and test split.

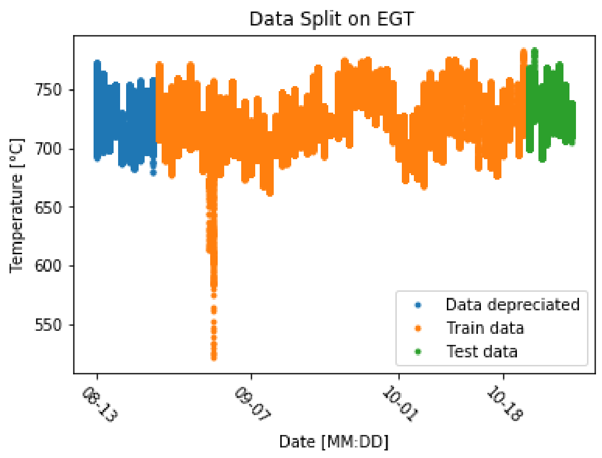


Figure 4.2: Data split.

Data was divided respecting the temporal sequence. To achieve this the train and test set was separated considering the maintenance date. Details of the separation are shown in table 4.1.

Table 4.1: Portions of data selected for each split. .

Engine	**Train set [days]**	**Test set[days]**	**Maintenance [day]**
AHD 1	10 to 66	66 to 78	73
AHD 2	10 to 66	66 to 78	73
AIY 2	22 to 75	10 to 22	13

For the engine AIY 2 test data comes from days previous train data, this separation is done considering that the engine had a maintenance on the day 13, so test data considers information from few days pre and post maintenance and train data considers a time spectrum where the fuel consumption was low.

4.3.2. Normalization and windowing

In addition to the train test split, during the data pre processing stage, data is Normalized. As explained on [6], consider the example where for a regression model the objective is to predict the price of sale for a house considering as input its location, the year it was constructed, the price of rent from other years and so on. Without the normalization of data, different features will have very different ranges and without a normalization process the effect of features with higher ranges of variance would mitigate the contribution of other features with lower variance in the network parameters.

After the normalizing the data, it is splitted in temporal windows, considering an overlap between continuous windows. Each window is composed by 20 continuous measures, which correspond to 20 seconds. Between continuous windows there's a separation of 16 seconds.

The selection of this number of continuous measures is justified by an improvement in the prediction curve (compared to the prediction with no temporal windows) and both a lower training time and similar results than the ones obtained with larger temporal windows.

It is worth mentioning that during data normalization the standard approach was followed, considering only the train data to obtain the normalization constants.

$$\begin{pmatrix} X_{1,1} & X_{1,2} & X_{1,2} & X_{1,3} & y_{1,1} & \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ X_{20,1} & X_{20,2} & X_{20,2} & X_{20,3} & y_{20,1} & \end{pmatrix} \tag{4.1}$$

Model's input. This shows the information contained on one window.

Where:

- $X_{:,1}$ to $X_{:,4}$ correspond to features Altitude, EGT, N1 and N2 speed.
- $y_:$ represents the fuel consumption.

4.3.3. Training

During the training process, one of the most significant hyperparameters to be set is the number of epochs over the training process is going to be repeated. Typically the train set is going to have a validation split. This data is not used to modify the weights of the model but to test the model performance after each epoch of training, checking the capacity of generalization during training. Figure 4.3 shows the classical result obtained after training for a long number of epochs.

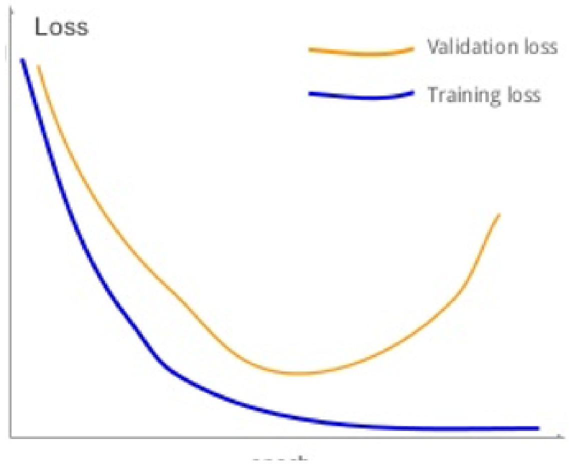


Figure 4.3: Validation and train loss over epochs.

The plot in Figure 4.3 is composed by three main sections, described bellow.

- The first is known as *underfitting*, a stage characterized by a decrease of both validation an training loss.
- The second phase is an appropriate fitting of the model, where both validation and training loss are low, hence the network has a good capacity of generalization.
- The final stage is the *overfitting*. In this section the training loss continues to decrease over epochs but validation loss starts to increase, showing that the model losses its capacity of generalization and starts to identify patterns only present on the train split.

Both under and overfitting are unwanted. Simple examples of the prediction boundary for a classification problem on 2D considering each type of fitting are shown in figure 4.4

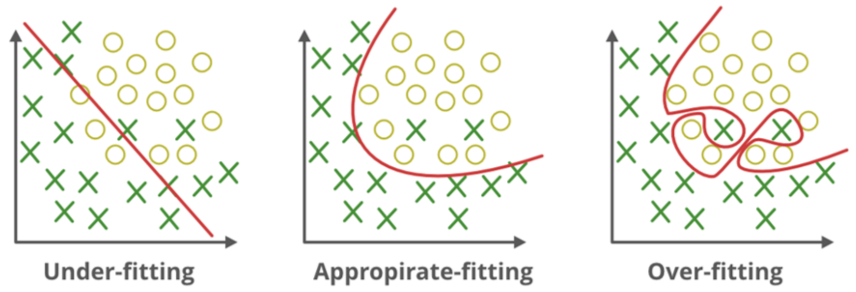


Figure 4.4: Types of fittings.

4.4. Resources

To develop the thesis the following resources are used:

- **Measured data:** The databases used were provided by SKY Airlines.
- **Software:** for programming Python 3.7 is used, including Pandas (1.1.2), Sklearn (0.21.2) and Numpy (1.18.1) as the mainly used libraries. The machine learning framework includes Keras 2.2.4 and TensorFlow 1.13 [11].

Chapter 5

Results and Discussion

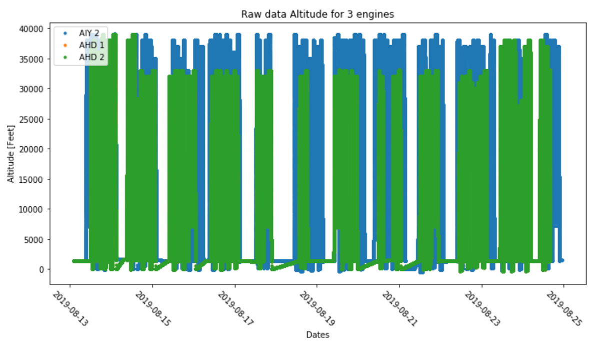
5.1. Data Exploration

5.1.1. Raw data plots

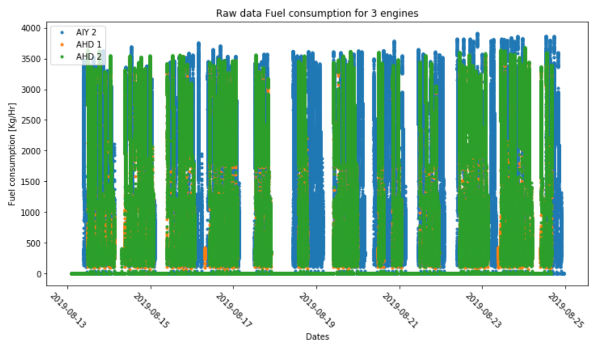
The plots (a) to (e) in Figure 5.1 show a percentage of the full data (this portion is selected to both show a big picture of the data and keep it roughly distinguishable). With this big spectrum of information some conclusions emerge:

- None of the sensors has a low variance.
- The altitude measurements are equal for AHD 1 and AHD 2.
- There are no evident outliers.
- The three DS have similar ranges, which is to be expected considering similar engines and planes.
- High density zones are similar for the three engines.

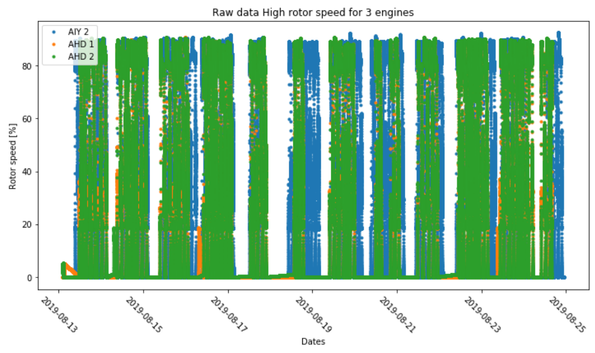
With the information pointed above, data doesn't show evident problems at first sight.



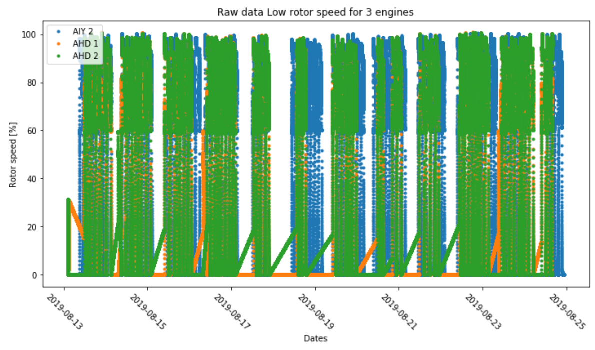
(a) Altitude.



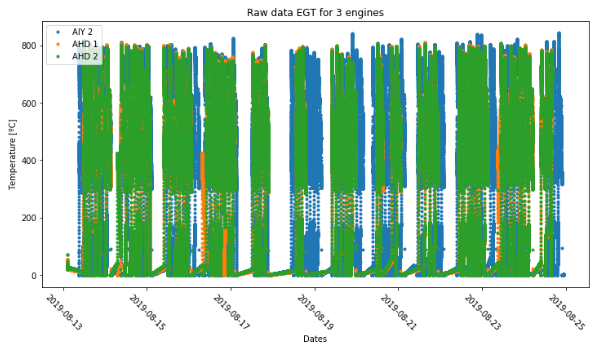
(b) Fuel mass flow rate.



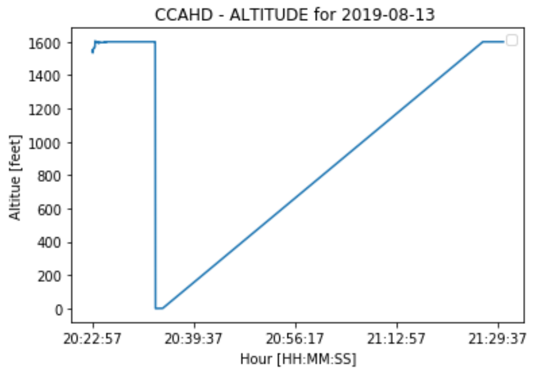
(c) High rotor speed efficiency.



(d) Low rotor speed efficiency.



(e) EGT.



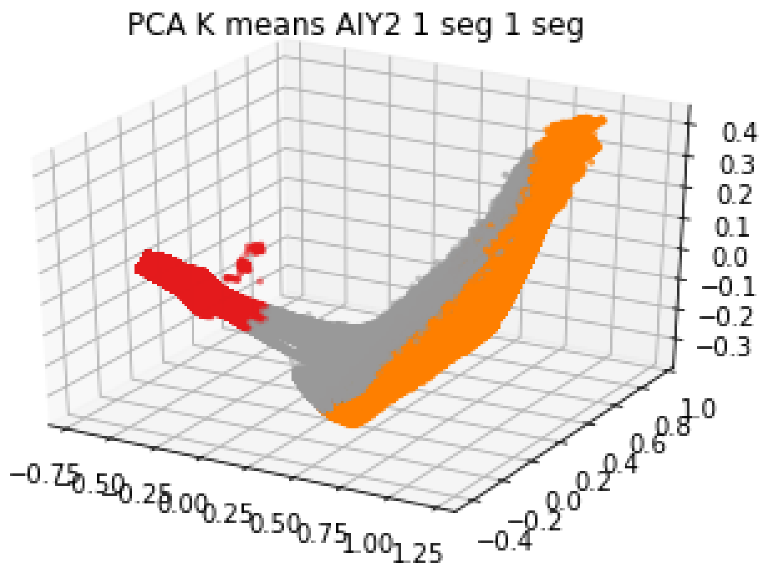
(f) Measurement failure on the altitude sensor.

Figure 5.1: Portions from raw data (a-e) and a measurement failure (f).

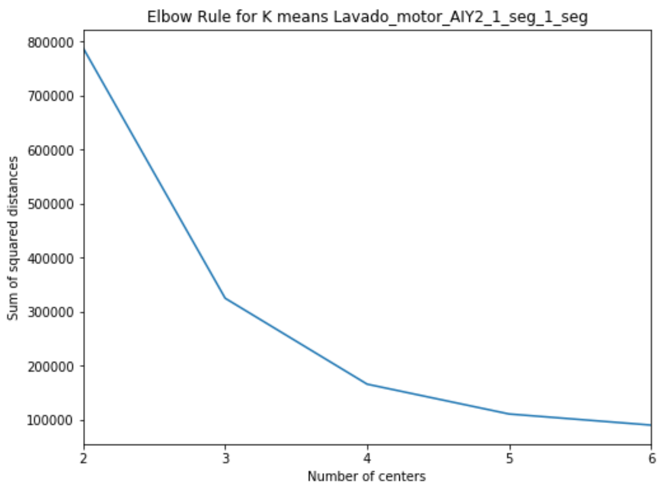
5.1.2. PCA and clustering techniques

The motivation behind the use of PCA and clustering algorithms is to obtain a low-dimensional representation of the data while maintaining as much information as possible and to identify the sections of interest arising from the clustering algorithms.

After appling a PCA with 3 dimensions and using the K means clustering algorithm, plot a from Figure 5.2 figure (a) is obtained. Figure 5.2 figure (b) shows the plot of SSE for various k values, justifying k = 3 as a good choice, according to the elbow rule.



(a) PCA plot with 3 dimensions and clustering colors obtained using K Means with k = 3.



(b) Elbow rule for selecting k = 3.

Figure 5.2: PCA and elbow rule.

About the visualization of the full DS on 3D, the results show a spatial location without large gaps or marked islands that translate into obvious gaps., which is expected because of the continuous movement measurement on the time series.

The results of the classification are as expected, showing a marked border between classes, this because of k centers used in the algorithm. Several other clustering algorithms were used during the data exploration phase (DBSCAN, Agglomerative clustering, HDBSCAN) but none of them obtained a separation useful for the automatic identification of flight stages. Then this stages identification must be done manually or via an algorithm. The next section is dedicated to those explain the function of the developed algorithms.

5.2. Flight intervals results

Two algorithms were developed in order to automate the identification of flight intervals. This first algorithm identifies the full flight interval.

To obtain the correct result, first a definition of a flight is needed. A flight is defined as such if it meets a set of conditions. These conditions are named as follows:

- Derivative of the altitude sensor must be greater than threshold 1.
- Airport altitude less than a threshold 2.
- Maximum differences in altitude values between the positions located with the derivatives must be greater than a threshold 3.
- Skip continuous points in a defined time interval (threshold 4).

The results obtained with this algorithm are presented in table 5.1.

Table 5.1: Flight intervals results for the three engines.

Engine	**Sampling rate (Hertz)**	**Number of flights**
AHD 1	1	612
AHD 2	1	612
AIY 2	1	563

Regarding the results obtained with this algorithm, they mean an average of approximately 6 flights per day, a number very likely to be true during a normal day of operation, validating the results. A verification of the results was done and gave positive results, validating the proposed algorithm. As expected, the identification for engines AHD 1 and AHD 2 produce the same number of flights.

Several optimizations can be done to make this algorithm more efficient, but are out of the scope of this project, which does not require a continuous application of the algorithm with new data.

5.3. Take off intervals results

A second algorithm, much simpler than the first one, takes a subsection from the data provided by the first. Taking in consideration the procedure applied in [19] a subsection of the data is selected, only considering the interval from 10.000 to 30.000 feet over the sea. This selection of the data helps to avoid eventual differences stemming from the beginning of the flight and reduces the amount of intervention of external conditions, such as temperature and other weather conditions that can affect the combustion occurring inside the engine.

This simple algorithm only considers the data if the altitude sensed is between 10.000 and 30.000 feet.

The pseudo algorithm is defined as:

Code 5.1: Python example of the take off interval algorithm.

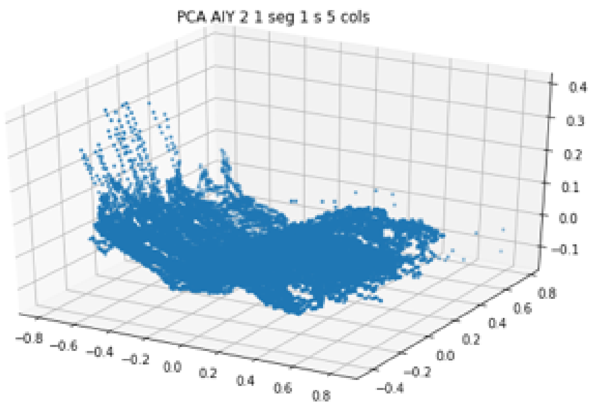
```python
import pandas as pd

data = pd.Dataframe(data)
alti = data['Altitude'] # define the altitude feature as a series
first_fly_section = [] # list defined to include the positions
for j in range (len( first_fly )):
    section =[]
    for i in range( first_fly [j ][0], first_fly [j ][1]) :
        if alti [i] < 10000 or alti [i] > 30000:
            continue
        else :
            section .append(i)
    first_fly_section .append([section [0], section [-1]])
```

In the example code 5.1 *first_fly_section* is a list that includes the first start and end of the first flight of each day.

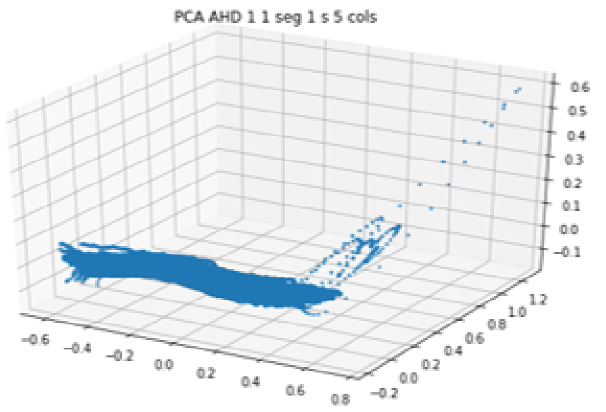
There is room for improvement in the example above and is declared as out of the scope, considering that the objective of this work is not to be used in real time.

5.4. PCA on the final data

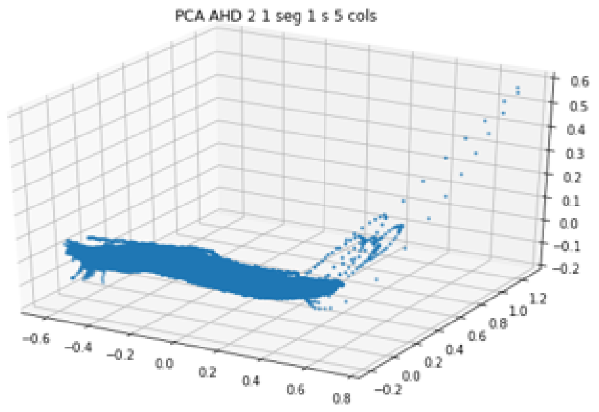
From original dataset, which included continuous measures from 2 months and 16 days, 70 to 76 flights are selected, considering that this flights have similar conditions (similar hour of departure, same airport and similiar weather conditions). With this new selected data the PCA plots are presented in Figure 5.3 figures (a) to (c), considering the 3 main components, with one plot for each engine.



(a) AIY 2



(b) AHD 1



(c) AHD 2

Figure 5.3: PCA on selected data. 3 Engines.

From this plots, the following observations emerge:

- Very similar space locations are obtained for engines AHD 1 and 2. This is expected since both engines are exposed to the same external factors (weather, weight load, maintenance) and work together.
- The plot on 5.3 a, related to engine AIY 2 has some common characteristics with b and c, but doesn't have the same degree of similarity. This is also expected, since it works on a different plane which deals with different external factors.

An important question is, having this similarities, do they mean anything regarding the original data? To partially answer this, the percentage of variance kept from the original DS is presented on table 5.2.

Table 5.2: Percentages of variance on the three PC.

Engine	**1st PC** [%]	**2nd PC** [%]	**3rd PC** [%]	**Total Sum** [%]
AIY 2	78.56	19.47	1.27	99.31
AHD 1	94.26	4.99	0.41	99. 67
AHD 2	93.79	5.39	0.45	99.64

This results indicate that considering the three main components for each engine imply keeping more than 99% of the variance of the original data, augmenting the probability of this similar space locations being relevant on the original data.

5.5. Fuel consumption

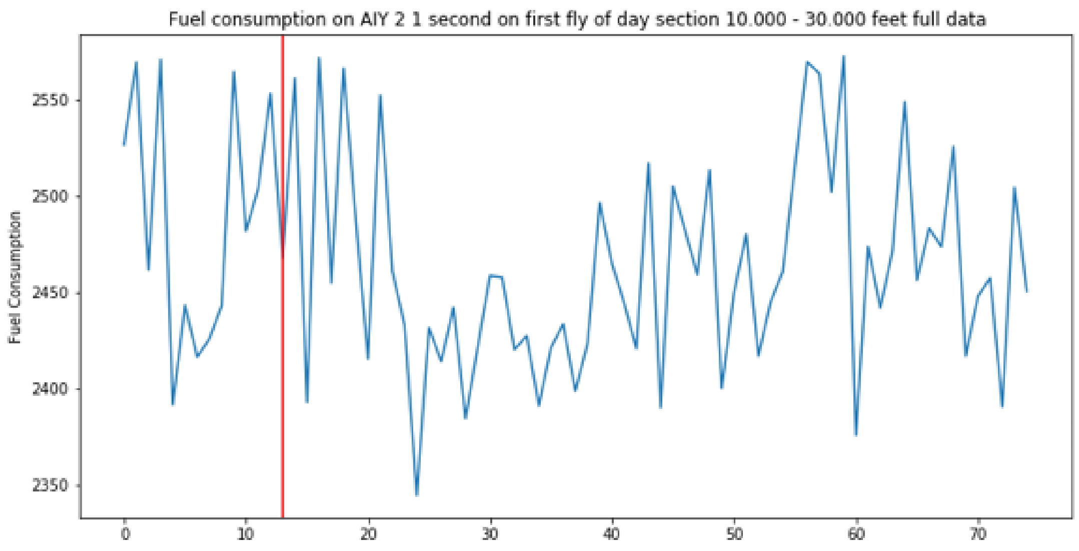
As an engine degrades, its fuel consumption decreases in efficiency and requires more fuel to run. This produces a relation between efficiency and fuel consumption. In order to check this consumption the plots in Figure 5.4 figures (a) and (b) show the mean fuel consumption during the 10.000 to 30.000 feet interval.

Using the mean fuel consumption of all the take off intervals (500 to 600 flights) obtained in section 5.3 does not provide a clear result, this is attributed to a variance in external conditions that affect the flight, such as weight charge, weather conditions, airport altitude, among others. In order to reduce the effect of these conditions, the analysis only considers the first flight of each day. These flights always start from the same airport, have similar hours of departure and reduce the variance of external conditions.

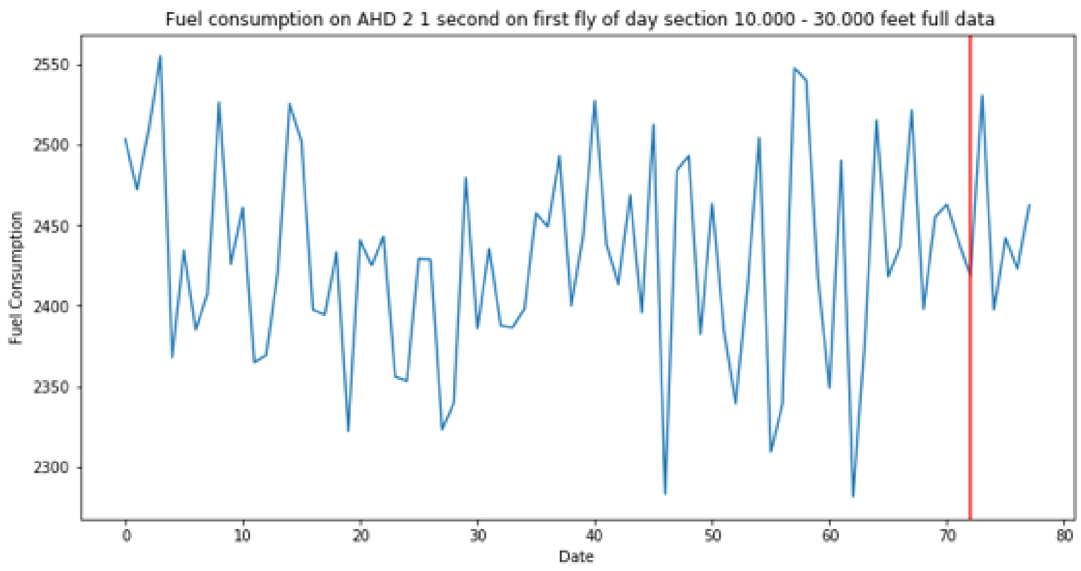
With this new subsection of data interesting information appears:

- The blue curve, which represents the fuel consumption, still conserves some variance, probably because of weight charge, wind or other non measured external factors. That being said, there is a clear tendency where previous to the maintenance of an engine the mean fuel consumption is higher than 10 days after it.
- The fuel consumption does not diminish immediately after the maintenance, but takes approximately 5 to 10 days to show a clear decrease. This is phenom is common in the maintenance of mechanical equipment.
- Following the decrease there is a clear trend to increase the fuel consumption over time.

For engines of the plane AHD maintenance was done on the first day of measures, day 0 on figure 5.4 (b).



(a) Fuel consumption in AIY2. Red line represents day of engine maintenance.



(b) Fuel consumption in AIY2. Red line represents day of engine maintenance.

Figure 5.4: Mean fuel consumption on engines AHD 2 and AIY 2 for the first fly of the day.

5.6. Experiments on PCA results considering fuel consumption

On this new space, created by the three principal components of the original data, several experiments were done, looking for a relationship between the distance of different flights data and the degradation of the engine in time.

The experiments considered:

1. Measuring distance between the mean value of different flights.
2. Check if a particular direction on this new 3D space indicated more degradation.

Since data has variance, zones were identified, a low consumption zone, located approximately from 10 days after the maintenance to 40 days after the maintenance, and 2 high consumption zones, one 10 days pre and post maintenance and the second from 40 days after the maintenance to the end of measurements. Considering this clusters, experiments 1 and 2 were repeated.

Since different measures with increasing degradation (defining degradation as the combination of more days since the maintenance date and a higher mean in the consumption of fuel, this to fight the variance of data) didn't follow a particular direction no conclusive results were identified. as no relationship was found, no further results are presented.

These experiments were developed considering 2 and 3 dimensional spaces obtained with PCA, and similar results were obtained.

5.7. Fuel consumption prediction

For the prediction of fuel consumption 4 machine learning models are used: linear regression (LR), decision trees (DT), gradient boosting (GB) and long short term memory neural networks (LSTM). Details on the selection of hyper parameters for each model are presented in appendix.

Tables 5.3 to 5.5 show mean average error (MAE) for predictions obtained with the different models on each engine.

Table 5.3: MAE for different epochs of the model. Engine AHD 1.

Model	Train set	Test set
LSTM	0.026	0.022
LR	0.006	0.006
DT	0.001	0.008
GB	0.004	0.006

Table 5.4: MAE for different epochs of the model. Engine AHD 2.

Model	Train set	Test set
LSTM	0.022	0.018
LR	0.006	0.005
DT	0.007	0.007
GB	0.003	0.005

Table 5.5: MAE for different epochs of the model. Engine AIY 2.

Model	Train set	Test set
LSTM	0.023	0.026
LR	0.009	0.009
DT	0.010	0.0010
GB	0.003	0.007

Results show a better performance for GB on the three cases. This motivates its selection for the development of an anomaly detector.

5.8. Anomaly detector

5.8.1. Motivation and justification of the model

Very few publications share operational data from both planes and engines. As stated in [9] 'Although a turbine manufacturer usually provides data about the turbine interface, the data required to estimate the thermodynamic cycle of a particular gas turbine remains hidden' and [2] 'Those fortunate enough to be able to collect long-term data for fleets of systems tend to – understandably – hold the data from public release for proprietary or competitive reasons'.

One of the few exceptions was the public competition of the PHM society of 2008 [2], where a filtered version of planes and engines data was publicly released with the objective of a competition, predicting the remaining useful (RUL) life of components.

With this lack of public information most of the academic literature focus on the use of specialized software such as GasTurb[1] [7],[20] GSP[2] or C-MAPSS [3] [24], all of this software requires licenses.

Without access to this kind of specific software a good choice is to create and anomaly detector. The model predicts the fuel consumption using as input the other features (N1 and N2 speed, EGT and Altitude). This anomaly detector is based on gradient boosting algorithm.

Overall the results obtained show great performance of the prediction model, considering normalized data the mean absolute error (MAE) is around 0.5-0.8 %. An example of a prediction and true value of the fuel consumption is shown in Figure 5.5.

[1] GasTurb web page. Last check October 6th.
[2] GSP web page. Last check October 6th.
[3] C-MAPSS web page. Last check October 6th.

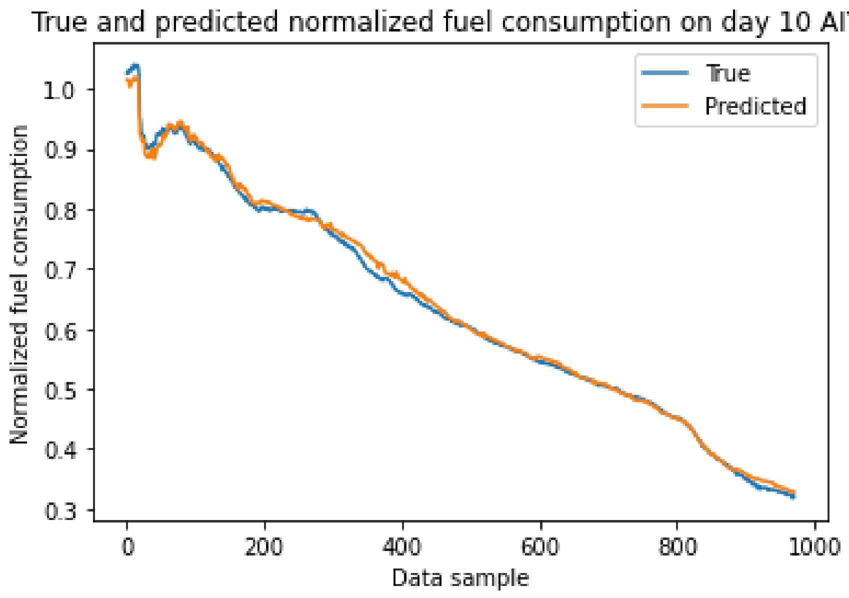


Figure 5.5: AIY 2 True and Predicted normalized fuel consumption exmple.

5.8.2. Anomaly detector results

After obtaining a model that predicts the engine's fuel consumption, its error in the prediction per day is analyzed. The objective is to develop an anomaly detector that labels an engine as healthy or degraded during a flight.

Engine AIY 2

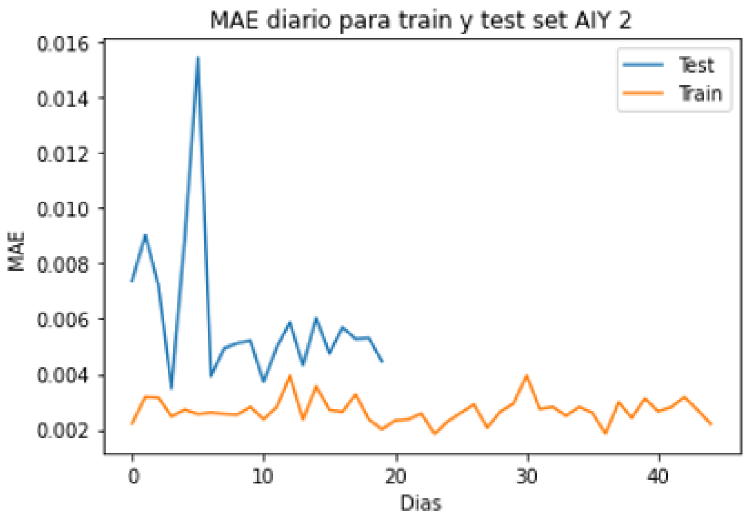


Figure 5.6: AIY 2 Daily MAE for train and test data.

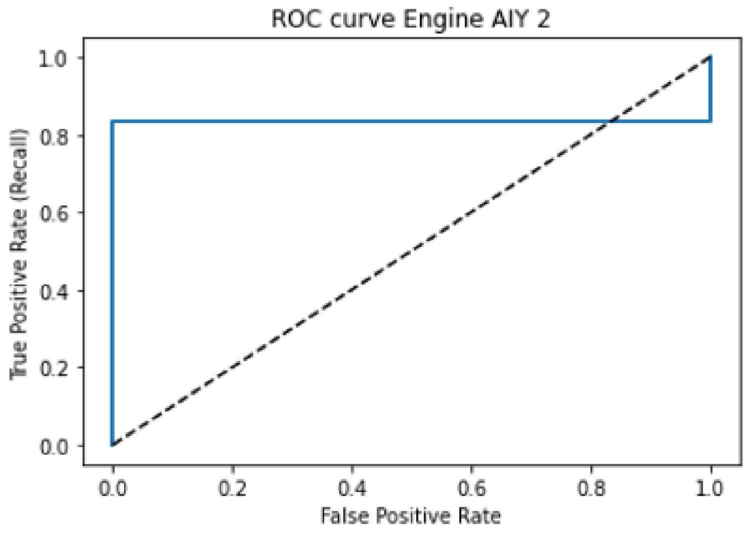


Figure 5.7: AIY 2 ROC curve.

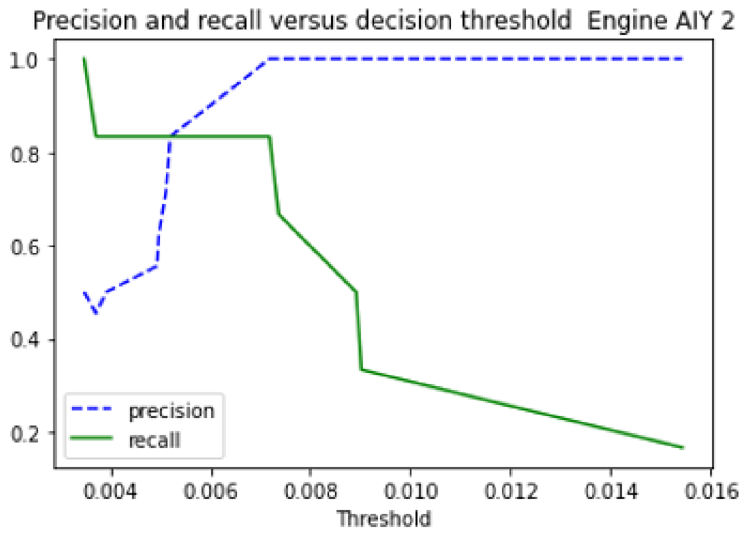


Figure 5.8: AIY 2 ROC curve.

Figures 5.6 to 5.8 show daily MAE obtained for train and test data, receiver operating characteristic curve and recall/precision curve, respectively.

Engine AHD 1

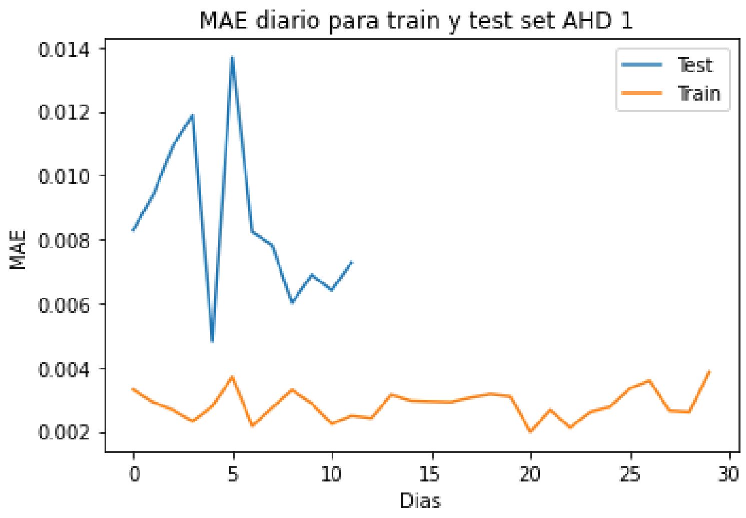


Figure 5.9: AHD 1 Daily MAE for train and test data.

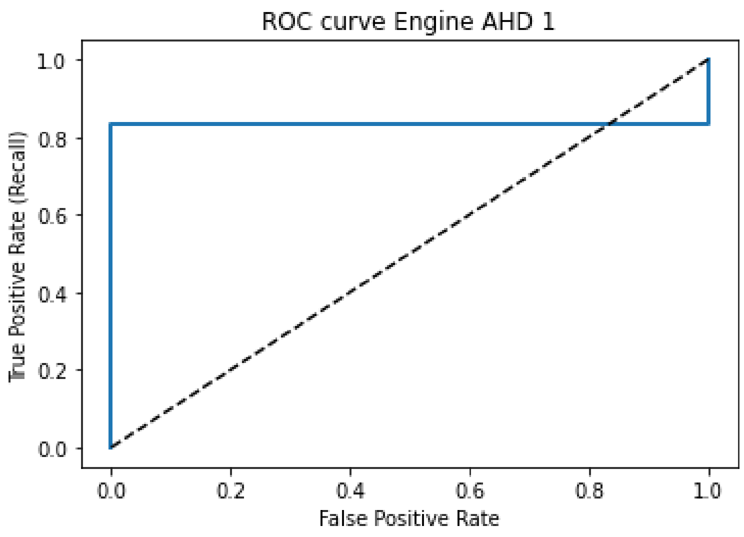


Figure 5.10: AHD 1 ROC curve.

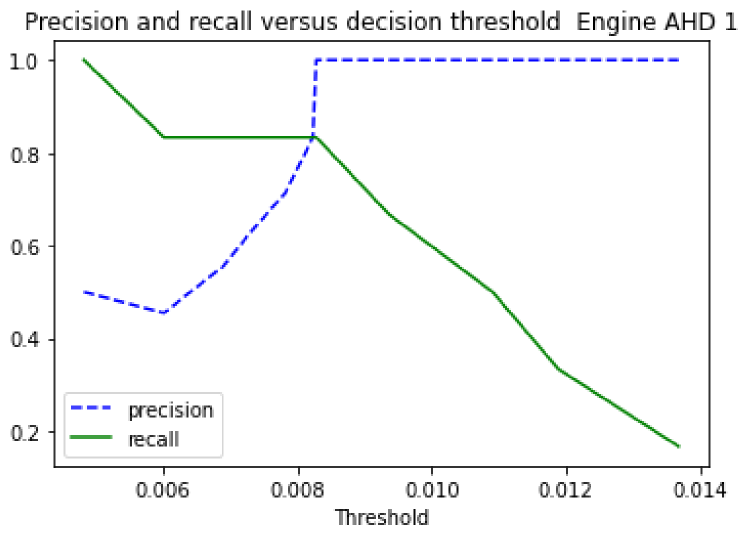


Figure 5.11: AHD 1 ROC curve.

Figures 5.9 to 5.11 show daily MAE obtained for train and test data, receiver operating characteristic curve and recall/precision curve, respectively.

Engine AHD 2

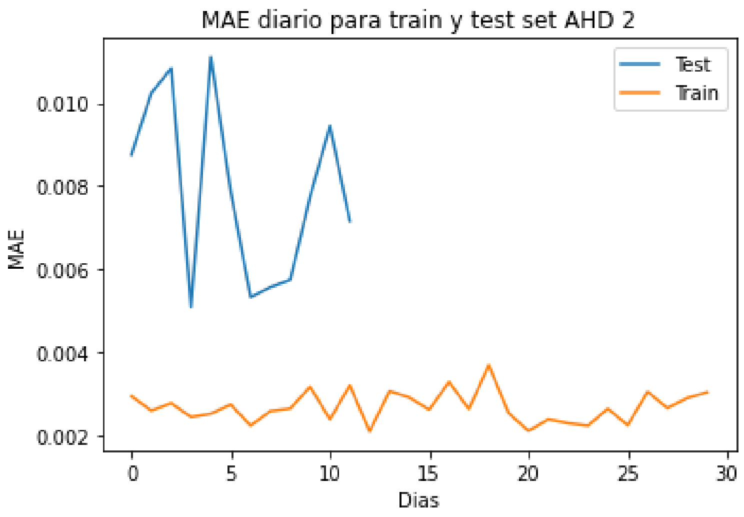


Figure 5.12: AHD 2 Daily MAE for train and test data.

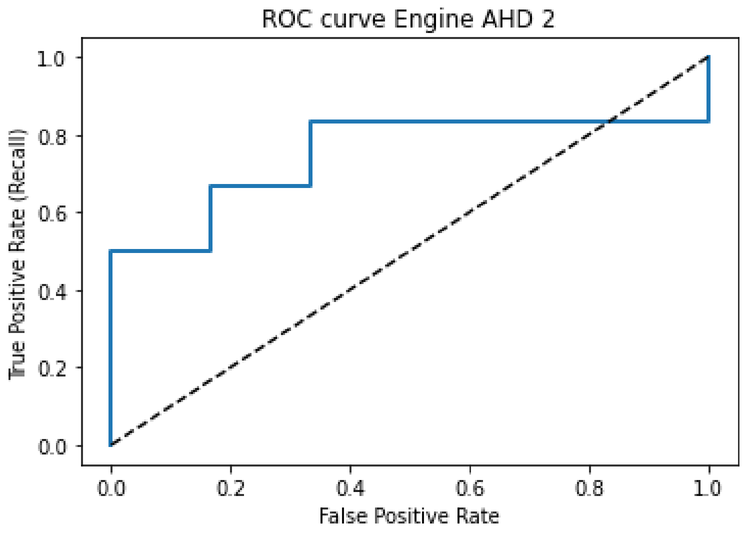


Figure 5.13: AHD 2 ROC curve.

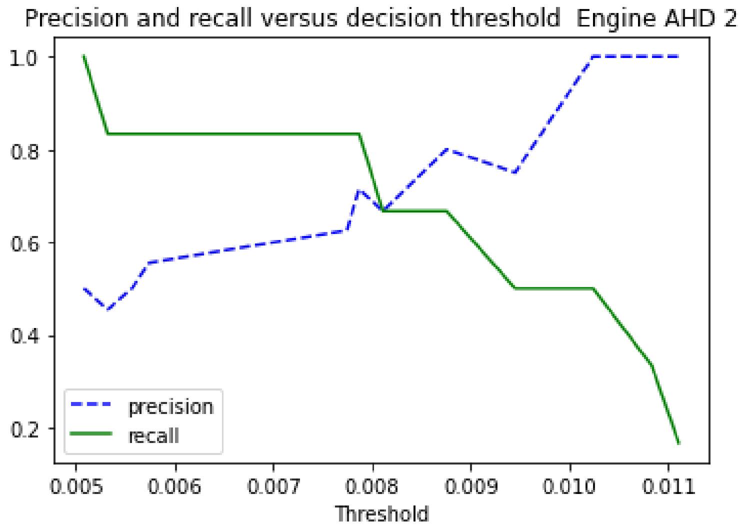


Figure 5.14: AHD 2 ROC curve.

Figures 5.12 to 5.14 show daily MAE obtained for train and test data, receiver operating characteristic curve and recall/precision curve, respectively.

Table 5.6 shows details on the threshold selected for each anomaly detector.

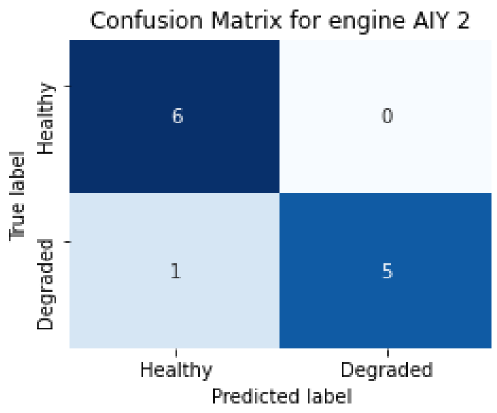
Table 5.6: Thresholds selected.

Engine	Threshold
AHD 1	0,0082
AHD 2	0,0078
AIY 2	0,0052

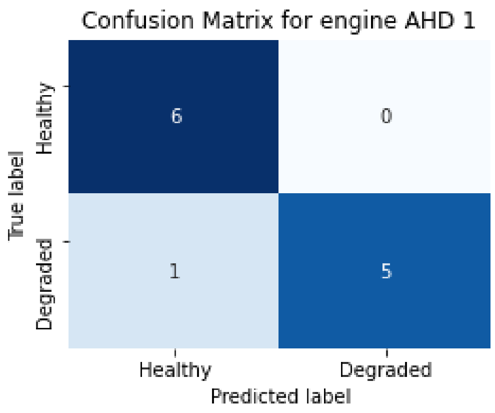
The selected thresholds are in the same order of magnitude for the three DS.

The data selected for the test corresponds to 12 days, 6 before and 6 after the maintenance of each engine. The data collected before and after maintenance are labeled as unhealthy and healthy, respectively.

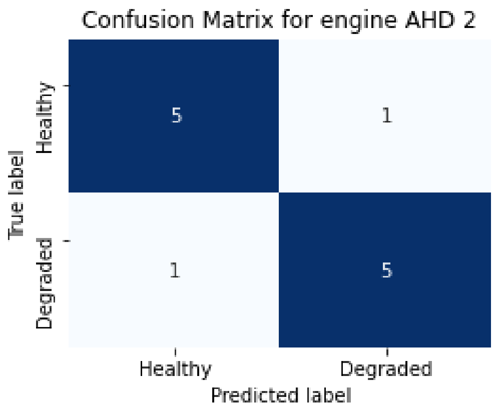
After labelling data and selecting a proper threshold for each model the following metrics of evaluation are obtained.



(a) AIY 2



(b) AHD 1



(c) AHD 2

Figure 5.15: Confusion matrix for each engine.

Table 5.7: Accuracy, recall and precision for each engine.

Engine	**Accuracy (%)**	**Recall (%)**	**Precision (%)**
AHD 1	92	83	100
AHD 2	83	83	83
AIY 2	92	83	100

Evaluation metrics for engines AIY 2 and AHD 1 are the same, having only one incorrect prediction. Specific results for evaluation metrics are an accuracy of 92%, recall of 83% and precision of 100% for both cases.

Results from the confusion matrix of the engine AHD 2 (Figure 5.15, (c)) show a higher prediction error for engine AHD 2. classification problem. In this case 2 flights are incorrectly classified. Accuracy, recall and precision reach an 83 %. This indicates that the classification problem is harder for engine AHD 2.

For the three engines the majority of the days used as train data have a lower MAE compared to the days used for test data (Figures 5.6, 5.9 and 5.12). This is attributed to a better fit of the model's predictions when using data from a healthy engine compared to more inaccurate predictions when dealing with a degraded engine (Figure 5.15, (a) to (c)).

Overall, evaluation metrics tend to show few incorrect predictions, 1 for engines AIY 2 and AHD 1 and 2 for engine AHD 2.

Chapter 6

Conclusions and Remarks

Many industries have taken advantage of the data revolution [25]. New techniques have emerged and succeeded to process vast amounts of data in the PHM field. Each flight generates a large amount of information and this trend is only going to increase over the next few years. This data could be used to feed machine learning models which can predict the remaining useful life of an engine or detect anomalies during the flight.

As suggested by the literature [2] a mayor drawback for the development of the field is the privacy of data for both engines and planes. This - although understandable - limits the academic field until synthetic data could achieve the quality needed to develop and test algorithms around it.

A literature review shows that one of the main sensibilities of the aviation industry, both for an economical and environmental perspectives is the fuel consumption of the engines.

This work has the advantage of using real data sensed from planes, but the limitations of ignoring specific behaviour of internal parts of the engine. This tendency is common on specialized literature. Considering this scenario an interesting path (the main objective of this work) is to develop an anomaly detector. This anomaly detector first predicts the fuel consumption and then labels the difference between true and predicted data as normal or abnormal.

During the data exploration phase, interesting patterns appeared, such as a notable peak in the fuel consumption in the seconds previous and the first minutes of take off. Considering the importance of fuel consumption and its relation to engine efficiency this section of high fuel consumption is selected to further analysis.

After the delimitation of data to only take offs the mean fuel consumption of fuel was obtained and no clear tendencies were identified. This is attributed to the high variance still present in this section of data. This variance can come from several external factors such as the weight at departure, weather conditions, altitude of the airport among others. Aiming to reduce the mentioned variance two modifications were made; first a subsection of the original section was selected, this time considering only flights from the same airport. After this intervals from 10.000 to 30.000 feet of altitude where selected.

A second exploratory phase was conducted, this time with the final data and similar patterns were presented for the 3 engines. Mean fuel consumption maintains variance but shows much clearer tendencies. There is a clear relation between maintenance being done to engines and its effect within a few days to reach a lower mean fuel consumption. This is to be expected.

The last section of this work trains a gradient boosting (GB) model and delivers an anomaly detector. The GB model is based on 16.000 decision trees (DT) as estimators. The objective of the model is to predict the fuel consumption based on the other features (altitude, EGT, N1 and N2 rotor's speed), a regression task.

To create the anomaly detector a threshold must be selected to identify predictions as normal or anomalies. The error metric used in prediction was MAE. The threshold selected for each engines took in consideration receiver operating characteristic (ROC) and recall-precision curve. Three values are in the same order of magnitude (Table 5.6).

After the selection of an appropriate threshold the objective is to correctly classify the state of health of the engine during a flight. Evaluation metrics show accuracies of 92 % for engines AIY 2 and AHD 1 and 83 % for engine AHD 2 (Figure 5.15 and Table 5.7).

Overall, no extremely abnormal behaviours were identified and a clear effect of maintenance in the fuel consumption is present (Figure 5.4.). This tendency is more clear when analyzing data from flights exposed to similar external factors.

The method used to detect anomalies considers as a main factor the fuel consumption according to the other parameters sensed. To generate a more robust method it would be interesting to include typical analyses performed prior to the maintenance of an engine, such as analysis of lubricants.

Chapter 7

Future Work

An interesting approach to test the methodology developed in this work would be to test the models on data collected from a larger time spectrum, which would include more maintenance dates. This would lead to validate the fuel consumption pattern occurring pre and post maintenance.

The inclusion of other useful features on the model would provide more consistency to the predictions. It's important to mention that some features were excluded because their use was not beneficial to the prediction.

Other ideas for future future work include use of different architectures for the model selected, such as gated recurring units (GRU) [17] or attention models [4], which have had great success in other areas of ML.

With respect to external factors affecting the data, one approach could be the normalization of data considering these factors (weight of the plane at departure, weather conditions, etc..).

Other options not considering specialized software include the use of a neural network (NN) to extract information from raw data and then train a second NN with the extracted features.

To generate a more robust method it would be interesting to include typical analyses performed prior to the maintenance of an engine, such as analysis of lubricants.

Bibliography

[1] Aircraft characteristics airport and maintenance planning., 2020.

[2] Don Simon Abhinav Saxena, Kai Goebel and Neil Eklund. Damage propagation modeling for aircraft engine run-to-failure simulation. *International Conference on PHM*, 2008.

[3] Felipe Bravo-Marquez Alan Ansell and Bernhard Pfahringer. An elmo-inspired approach to semdeep-5's word-in-context task. 2019.

[4] Niki Parmar Jakob Uszkoreit Llion Jones Aidan N. Gomez Lukasz Kaiser Illia Polosukhin Ashish Vaswani, Noam Shazeer. Attention is all you need. *NIPS*, 2017.

[5] Christopher M Bishop. *Pattern recognition and machine learning*. Springer, 2006.

[6] Francois Chollet. *Deep Learning with Python*. Manning Publications, 2017.

[7] Mehmet Emin Cilgin and Onder Turan. Entropy generation calculation of a turbofan engine: A case of cfm56-7b. *International Journal of Turbo Jet-Engines*, vol. 5:pp 217–227, 2017.

[8] G. Cybenko. Approximation by superpositions of a sigmoidal funtcion. *Mathematics of control, signals and systems*, vol. 2:pp. 303–314, 1989.

[9] Francisco da Costa Baptista. A 0-d off-design performance prediction model of the cfm56-5b turbofan engine. 2017.

[10] Geoffrey E. Hinton Ronald J. Williams David E. Rumelhart. Learning representations by back-propagating errors. *Nature*, vol. 323:pp. 533–536, 1986.

[11] M. Abadi et al. Tensorflow: A system for large-scale machine learning. *Proceedings of the TensorFlow: A system for large-scale machine*, 2015.

[12] Aurélien Géron. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*. O'Reilly, 2018.

[13] Josef Hochreiter. Untersuchungen zu dynamischen neuronalen netzen. 1991.

[14] Perry Bradley Jamie Jewell, Charles Soret and Talal Ahmed Almahmood. Cfm56 fleet surpasses 800 million flight hours.

[15] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *ICLR*, 2015.

[16] B. Kosko. Bidirectional associative memories. *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 18:pp. 49–60, 1988.

[17] Dzmitry Bahdanau Kyunghyun Cho, Bart van Merrienboer and Yoshua Bengio. On the

properties of neural machine translation: Encoder-decoder approaches. *Proceedings of SSST-8*, 2014.

[18] Luca Di Vito Lorenzo Fedele and Fulvio Enzo Ramundo. Increasing efficiency in an aeronautical engine through maintenance evaluation and upgrades: Analysis of the reliability and performance improvements under financial issues. *Energies*, 13, 2020.

[19] Maria Navas Loro and Jérôme Lacaille. Datamining turbofan engine performance to improve fuel efficiency. *IEEE Aerospace Conference*, 2017.

[20] Daniel Alexandre Rodrigues Martins. Off-design performance prediction of the cfm56-3 aircraft engine. 2015.

[21] Alex Krizhevsky Ilya Sutskever Nitish Srivastava, Geoffrey Hinton and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, pages pp. 1929–1958, 2014.

[22] Warren S. McCulloch Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, vol. 5:pp. 115–133, 1943.

[23] Tomas Mikolov Razvan Pascanu and Yoshua Bengio. On the difficulty of training recurrent neural networks. *ICML*, 2013.

[24] Thomas Lavelle Jonathan Litt Ryan May, Jeffrey Csank and Ten-Huei Guo. A high-fidelity simulation of a generic commercial aircraft engine and controller. *ARC*, 2010.

[25] D.J. Patil Thomas H. Davenport. Data scientist: The sexiest job of the 21st century. *Harvard Business Review*, 2018.

[26] Geoffrey E. Hinton Vinod Nair. Rectified linear units improve restricted boltzmann machines. *ICML*, page pp. 807–814, 2010.

[27] Paul J. Werbos. Generalization of backpropagation with application to a recurrent gas market model. *Neural Networks*, vol. 1:pp. 339–356, 1988.

[28] Yoshua Bengio Geoffrey Hinton Yann LeCunn. Deep learning. *Nature*, vol. 521:pp. 436–444, 2015.

[29] Patrice Simard Yoshua Bengio and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEE Transactions on Neural Networks*, vol. 5:pp 157 – 166, 1994.

Chapter 8

Appendix

8.1. A. Training and evaluation of LSTM NN model

The LSTM model was trained to predict the engine's fuel consumption. Train data was feed to the NN through temporal windows. The loss function used was MAE.

Several combinations of layers and neurons per layer were tested, obtaining better results with 32 units per layer and the use of dropout after each LSTM layer.

Details of results are presented on tables 8.1 to 8.3. A visual representation of results is presented on Figure 8.1 figures (a), (b) and (c).

Table 8.1: MAE for different epochs of the model. Engine AHD 1.

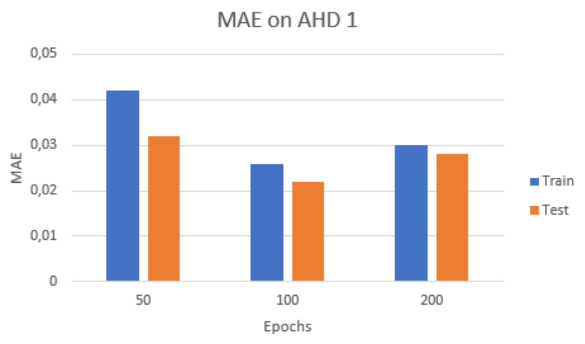
Epochs	**Train set**	**Test set**
50	0.042	0.032
100	0.026	0.022
200	0.030	0.028

Table 8.2: MAE for different epochs of the model. Engine AHD 2.

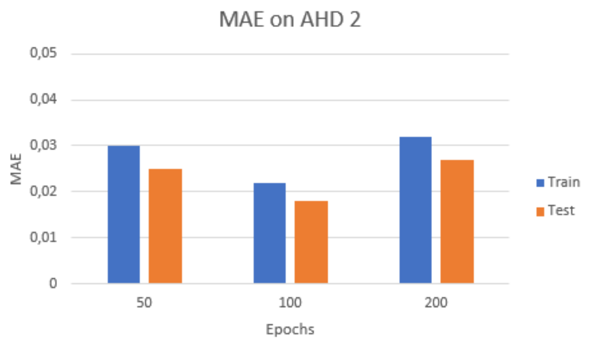
Epochs	**Train set**	**Test set**
50	0.030	0.025
100	0.022	0.018
200	0.032	0.027

Table 8.3: MAE for different epochs of the model. Engine AIY 2.

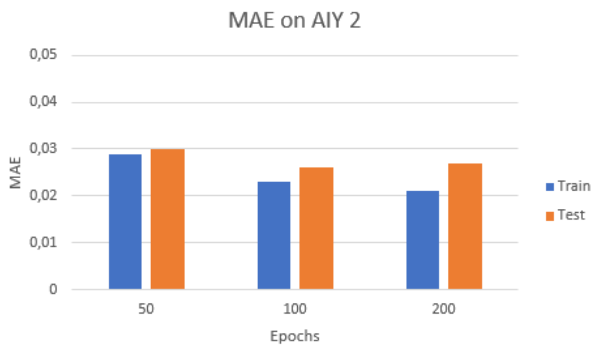
Epochs	**Train set**	**Test set**
50	0.029	0.030
100	0.023	0.026
200	0.021	0.027



(a) AHD 1 data.



(b) AHD 2 data.



(c) AIY 2 data.

Figure 8.1: MAE for the three engines on test and train data.

The best result considering MAE as the metric to evaluate each model were obtained with 100 epochs, after which signs of overfitting started to appear.

For the results using the engines AHD 1 and AHD 2 similar results are obtained. For both train and test data the best results are obtained using 100 epochs for the training of the model. After testing the model with 50 and 100 epochs train and test set predictions obtain a higher error. This results are attributed to under and overfitting, respectively.

On the model trained with data from engine AIY 2 (table 5.5) there is a continuous improvement of the prediction on the training data. On the side of test data, the best result are obtained with 100 epochs, after which there is a slight worsening in the prediction, attributed to overfitting.

8.2. B. Training and evaluation of linear regression model

During training 4 different degrees where used. Tables 8.4 to 8.6 detail results obtained for each engine.

Table 8.4: MAE for different degrees of the model. Engine AHD 1.

Degree	**Train set**	**Test set**
2	0.010	0.008
3	0.008	0.007
5	0.006	0.006
10	0.004	0.008

Table 8.5: MAE for different degrees of the model. Engine AHD 2.

Degree	**Train set**	**Test set**
2	0.010	0.008
3	0.009	0.007
5	0.006	0.005
10	0.004	0.007

Table 8.6: MAE for different degrees of the model. Engine AIY 2.

Degree	**Train set**	**Test set**
2	0.010	0.011
3	0.008	0.010
5	0.006	0.009
10	0.004	0.021

In the three cases a smaller MAE was obtained using a degree of 5, after which signs of overfitting appear.

8.3. C. Training and evaluation of decision trees model

Experiments considered different numbers of maximum depth allowed on decision trees. Tables 8.7 to 8.9 show details of results.

Table 8.7: MAE for different maximum depth of the model. Engine AHD 1.

Maximum depth	Train set	Test set
8	0.009	0.010
16	0.001	0.008
32	0.000	0.008
64	0.000	0.008

Table 8.8: MAE for different maximum depth of the model. Engine AHD 2.

Maximum depth	Train set	Test set
8	0.009	0.009
16	0.001	0.008
32	0.000	0.008
64	0.000	0.008

Table 8.9: MAE for different maximum depth of the model. Engine AIY 2.

Maximum depth	Train set	Test set
8	0.009	0.013
16	0.001	0.009
32	0.000	0.010
64	0.000	0.010

For the three DS a better generalization is obtained using a maximum depth of trees equal to 16. After this better results are obtained in train data but test set doesn't improve.

8.4. D. Training and evaluation of gradient boosting model

During experiments the maximum depth of trees and number of estimator were tuned. Since better predictions were obtained varying the number of estimators details only consider the change of this hyperparameter. Tables 8.10 to 8.12 show details.

Table 8.10: MAE for different numbers of estimators of the model. Engine AHD 1.

Number of estimators	**Train set**	**Test set**
2000	0.005	0.006
4000	0.004	0.006
8000	0.004	0.006

Table 8.11: MAE for different numbers of estimators of the model. Engine AHD 2.

Number of estimators	**Train set**	**Test set**
8000	0.003	0.005
16000	0.003	0.005
32000	0.002	0.005

Table 8.12: MAE for different numbers of estimators of the model. Engine AIY 2.

Number of estimators	**Train set**	**Test set**
8000	0.003	0.007
16000	0.003	0.007
32000	0.004	0.007

In the case of engines AIY 2 and AHD 2 the best fit was obtained using 16.000 estimators. For AHD 1 the number of estimator selected was 4.000. The other hyperparameters selected were a learning rate of 0.01, a maximum depth of 4 and the minimum number of samples for split of 5.