UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA AERONÁUTICA Y DEL ESPACIO

MÁSTER UNIVERSITARIO EN INGENIERÍA AERONÁUTICA

TRABAJO FIN DE MÁSTER

Machine Learning for Global Mission Assessment

AUTOR: David CALVO FERREIRA

INTENSIFICACIÓN: Aeronaves

TUTOR PROFESIONAL: Joaquín TORRES MÁRQUEZ

TUTOR DEL TRABAJO: José Félix ALONSO ALARCÓN

Febrero de 2020

ABSTRACT

Mission Assessment is a very important part in aviation, specially in the military field, since it allows us to discover what has gone well or bad in the development of the mission. All the necessary information to carry out this assessment is obtained through Debriefing systems, which download the data recorded in the flight recorders of the aircraft.

Currently, the amount of downloaded signals is huge and it is increasing as more missions are executed. This makes the traditional methods of performing assessments to be more tedious and complex.

Therefore, this project proposes to carry out an analysis to the determine if it is feasible to use machine learning techniques to apply automatic assessments for new missions that are performed, using all the information from previously executed missions.

RESUMEN

La evaluación de una misión es una parte muy importante dentro de la aviación, sobretodo en el ámbito militar, porque permite descubrir que es lo que ha ido bien y mal en el desarrollo de la misión. Toda la información necesaria para realizar esta evaluación se consigue a través de los sistemas de Debriefing, los cuales descargan toda la información grabada en los grabadores de vuelo de la aeronave.

Actualmente, la cantidad de señales que se descargan es enorme y va en aumento según se van realizando más misiones. Esto hace que los métodos tradicionales para realizar la evaluación se vuelvan cada vez mas tediosos y complejos.

Por ello, este proyecto propone realizar un análisis para determinar si es factible la utilización de técnicas de aprendizaje automático para aplicar evaluaciones automáticas de nuevas misiones que se realicen, utilizando toda la información ya conocida de misiones realizadas con anterioridad.

ACKNOWLEDGEMENTS

Contents

List of Figures

List of Tables

1 Introduction

In recent years, the application of Mission Planning and Debriefing has become essential in both civil and military aviation, as it provides solutions in order to improve the performance of the aircraft and to increase missions' success.

For this project, the Debriefing process provides all the signal from the aircraft, however, the amount of signals recorded during a flight is huge, so, working manually with this data is becoming more complex and tedious, since the amount of information is continuously increasing.

That is the reason why it is necessary to develop an automatic tool to analyze all the information faster and correctly, and to reach good results when performing Mission Assessment. Then, a mathematical model is created to follow the process in an analytical way.

Using this model, a hierarchical assessment of all the components is performed until achieving Global Mission Assessment. Finally, using Machine Learning algorithms, a predictive tool is developed to provide good results for Global Mission Assessment.

The prototype is implemented using the software Python, which is a programming language that lets you work more quickly and integrate your systems more effectively than doing it by hand. As an advantage against other programs, Python is developed under an OSI-approved open source license, making it freely usable and distributable, even for commercial use. Python's license is administered by the Python Software Foundation [1].

1.1 Scope

Nowadays, the amount of signals recorded on-board during a Mission is huge. In the short-term this situatuion will be worse and mission data should be managed in a massive way, specifically if we consider Complex Missions in Coorporative Scenarios.

Current Mission Debriefing Systems (platform-oriented) use just a portion of this recorded data. Post-flight analysis tools are very system-oriented or domain-oriented: Engineering, Safety, Maintenance, Electronic Warfare, etc.

Air Forces want to handle, as much as possible, recorded data to understand all the circumstances of missions (root causes, hidden factors). Pilots and Squadron Leads want to perform more agile Mission Debriefing sessions, considering all available data in an efficient way.

The concept Global Mission Assessment aims to integrate all the available information and automatically exploit it the right way to offer smart and customized responses to each end-user, according to their roles/needs.

Under Global Mission Assessment umbrella, this project is focused on a Smart Mission Debriefing System with the following valuable benefits: Time, Cost and Complexity reduction of Debriefing sessions.

1.2 Background of Mission Assessment and Flight Recording Systems

Before starting the explanation of the development of the project, it is necessary to define what is Mission Assessment and why it is employed inside the aeronautical field. Also, a brief description of the Flight Recording Systems is provided in order to illustrate the history of these devices, how they work and why they are so important in the improvement of signal analysis.

1.2.1 Mission Assessment

Mission Assessment has always been fundamental within aeronautical field, since it allows to establish if a mission was performed correctly and, otherwise, it enables to perform the necessary improvements to enhance future missions.

One of its main objectives is to find out why a mission was not successful. Then, it is necessary to detect any failure and inconsistency that appeared during the flight of the aircraft and these issues must be analyzed to discover the reasons why they arise.

Additionally, it is crucial to determine the source of the failure, being possible to have a malfunction in only one of the aircraft systems or to have a problem due to the combination of different systems which carry the issue form the first to the last one. For example, there is a fatigue structural failure affecting the oxygen tanks of the aircraft, this problem makes the pilot to have insufficient oxygen inside his body. Then, he might not take an appropriate decision and he does not execute the tasks properly, thus, the mission is not performed successfully. So, the first issue provoked the emergence of successive problems leading to fail the mission. Therefore, it is fundamental to detect the origin of the problem in order to develop a solution for future similar missions.

Once the failure and its causes have been detected and analyzed, the second step is to transfer all the collected and developed data to the responsible for that section of the plane, so that, they are able to fix what has produced this problem and, thus, avoid this

failure to be repeated in the future for the same aircraft or different ones.

Keeping this in mind, it is concluded that the process to evaluate the mission is fundamental, since, it is possible to keep clear of errors that led the mission to fail. In the case of this project, all the signals coming from aircraft systems are collected to develop a tool, which allow to determine, automatically, if a mission is successful or not.

1.2.2 Flight Recording Systems

Flight recorders are devices that register some data from the aircraft performance, this information is collected through the sensors located in the airplane. Governmental regulatory demand to have these instruments on commercial planes in order to be able to analyse the events that occurred during the flight. In fact, flight recorders are composed by two devices: the flight data recorder (FDR) and the cockpit voice recorder (CVR). Sometimes both are combined into a single unit called Combination recorders (ICAO Definition) [2].

The FDR must be placed in the aircraft's most protected part, which usually is the tail section. The decision to locate it in that place is made in order to make possible the survival of the device to be able to collect all the important information that was recorded during flight. This stored data is analysed by investigators to find out why the accident happened, determining if it was caused by a pilot error, by an external event or by a system problem. The reports developed by the analysts are helpful to improve aircraft design procedures and to predict potential failures that might occur in similar aircraft [3]. Also, an example of an application, in which the data recorded is implemented, is the generation of a computer animated video reconstruction of the flight [4]. Despite of their popularised name black box, they are painted using a vermilion colour called "international orange". In Figure 1 it is observed how a standard FDR can be.

Figure 1: Flight Data Recorder (FDR). Taken from [4].

Flight data recorders are classified into two groups:

- First-generation FDRs: Introduced in the 1950s, they recorded using metal foil that was located inside a box at the rear end of the plane. In the following years, they started to be painted to make them easier to find in case of an accident.
- Second-generation FDRs: Introduced in the 1970s due to the necessity to store bigger amounts of data.To do so, the flight data acquisition unit (FDAU) was created, which is a system that maps all the parameters coming from the sensors and avionic systems to a flight data recorder, this information is sent via data frames that are specified by the manufacturer. As an evolution, the second-generation digital FDR (DFDR) was developed and it uses a likely audio recording tape.

Nowadays, most recent FDRs have no moving parts, decreasing the probability of breaking and the maintenance becomes simpler. The data coming from the FDR and the cockpit voice recorder (CVR) is saved on stacked memory planks inside the crash-survivable memory unit (CSMU). Also, an Emergency Locator Transmitter (ELT) and an Underwater Locator Beacon (ULB) were incorporated in order to locate the FDR in case of an accident above water [3].

Regarding the operation of these devices, the FDRs installed in the newly manufactured aircraft must record eighty-eight main parameters. Additionally, there are FDRs that can record more than one thousand variables. Annually, it is mandatory to apply a verification

check to FDRs to test if all the important parameters are stored [3].

Additionally, there is a new type of recorders: Automatic Deployable Flight Recorders. This new concept states that it is more possible the survival of the data recorded if the flight recorder is ejected from the aircraft in a specific time. The recovery percentage of these devices is very high and all the information recorded can be used. So, they are a good alternative instead of using traditional flight recorders [5].

1.3 Objectives

The main objective of this project is to perform automatic Mission Assessment. In particular, to develop a model which contains all the information and to offer a tool, which has valuable benefits, such as, cost and time reduction when performing Mission Assessment.

As a way to achieve that purpose, the following steps must be fulfilled:

- Analysis of signals/data available through different flight recorders of an aerial platform.
- Selection of the data/signals to create different Flight Facets to group all the signal according to their characteristics.
- Define the formal model which describes for each mission the behavior/status of each Flight Facet in terms of procedure correctness and airworthiness constraints, to determine improvable procedure execution and/or potentially safety critical situations.
- Explain the mathematical model created to describe the assessment of each one of the components of Mission Assessment.
- Apply Machine Learning algorithms to the formal model to improve it in terms of prediction during mission planning phase.
- Design a simple prototype in which the formal model is implemented.

1.4 Document structure

This project is structured in six different chapters including an introduction and a conclusion:

- *Chapter 1 Introduction*: This section serves as an introduction where the basis on the different aspects that make up the project are explained.

- *Chapter 2 Analysis of source signal/data*: In this chapter, different aircraft systems are analyzed to obtain useful information from them. From this information a selection of the fundamental data is performed.
- *Chapter 3 Proposal Description*: This section includes a description of the different proposed models that are going to be applied.
- *Chapter 4 System Design*: In this chapter, the final system with all the component is developed, explaining the reasons for choosing this final scheme and illustrating the final model.
- *Chapter 5 Prototype*: This chapter describes the prototype implemented and how it works using the signals and the system model defined in previous chapters.
- *Chapter 6 Conclusions*: This chapter includes the final comments about the project, explaining the difficulties found and stating hat is the future of this work.

2 Analysis of source signal/data

An aircraft has multiple sensors that gather all the necessary data produced during the whole flight. This information is recorded as a set of signals that have to be decoded in order to be able to manage and interpret them.

The information extracted as signals will be grouped together according to some common characteristics that all of them share. This collection is called Flight Facet and it is explained in Section 2.1. Also, it is important to have knowledge about available and known signals and about the ones that are unknown and/or unobtainable as explained in Section 2.2. In Section 2.3, the criteria followed to classify the signals is described.

2.1 System/subsystem vs. Flight facet

The development of the systems inside an aircraft has evolved greatly during these years making possible to elaborate advanced technology that allowed to create more sophisticated and safer planes in civil and military field. Also, the emergence of the Remoted Piloted Air Systems (RPAS) has produced a progress in this technology and especially in the way of recording, receiving and storing all the signals coming from those systems.

The most important elements related to the interchange of data are the digital data buses that connect all the systems with the recording devices. These buses transmit all the information coming from the different systems to a central or distributed computing centre, this evolution has enabled to record health and prognosis together with failures. Also, this devices are a fundamental part for the integration of all the systems together allowing the communication and interaction between them [6]. Figure 2 shows, with a simple design, an example of three interconnected systems.

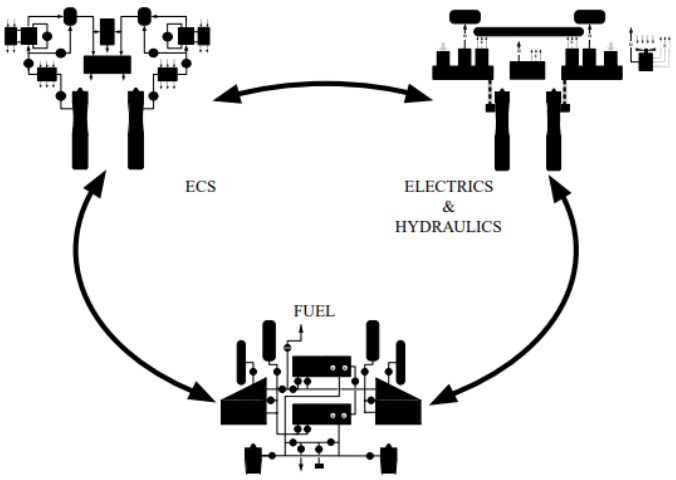


Figure 2: Typical military aircraft interaction. Taken from [6] page 29.

An aircraft is composed by the following systems (using the same classification as in [6]):

- Flight Control Systems (FCS): It enables the pilot to control the aircraft during the whole flight, this system is connected to control surfaces in order to command them to maneuver the airplane using translation and rotational motion.
- Engine Control System: Its main purpose is to manage the flow of fuel and air to the engine in order to achieve its optimum performance and efficiency for a range of speeds, altitudes and temperatures, allowing the pilot to control the engines avoiding any failure. This control depends on the type of aircraft and engine.
- Fuel System: The main purpose of this system is to supply fuel to the engines, it is an essential element to guarantee the safest possible flight, since its power is fundamental to maintain the whole flight.
- Hydraulic System: It is a vital source of power for flying controls, braking, undercarriage and anti-skid systems. The performed functions must work correctly and when commanded avoiding any failure condition. Nowadays, these systems are being substituted by electrical ones.
- Electrical System: These systems are the ones that have had a notable growth during the recent years. First, it is necessary to generate and to distribute the power, once it is done, this system covers several functions classified into: Motors and actuation,

lighting services, heating services, subsystem controllers and avionics systems.

- Pneumatic System: Its main purpose is to provide bleed air to other systems. The high pressure bleed air is used to operate the air extraction valves, medium-pressure air is applied to start the engine using an Auxiliary Power Unit (APU). Also, bleed air is utilized for anti-ice protection and to pressurize the cabin.
- Environmental Control System (ECS): ECS must provide good climatic conditions inside the cabin supplying the necessary oxygen's concentration avoiding moisture and optimizing humidity.
- Emergency Systems: They must be the final step to assure the survival of the people and aircraft. These systems are separated from the others to prevent the propagation if a failure is produced in the previous systems. Also, they usually have alternative sources of power and, when an emergency is going to happen, some warning indications are displayed.

So, taking into account the previous classification for aircraft systems, the Flight Facets are created. Each one of them is a group of decoded signals that share some mutual characteristics. The data obtained after the decoding is matched with an envelope that specifies the limits which should not be exceeded. Knowing these boundaries, it is possible to predict the behavior of the data and if some improvements must be done to fulfill the requirements. The main purpose of these data structures is to provide fundamental information about the recorded signals. After making the match with the envelopes it will be possible to know which data is inside the limits and which is outside or close to them. Making feasible to classify the signals depending on its behaviour.

Depending on the mission and its procedures, each flight facet will have a specific weight that determines the impact produced in the action. Considering this influence together with the results obtained after matching the data with the envelopes, it will be achievable to evaluate a particular action accomplished in a precise time interval. Performing this process for each of the actions, the assessing of the procedures can be completed and, consequently, the final evaluation of the whole mission is determined.

2.2 Analysis of available signals/data

The amount of signals recorded and stored in the flight recorders is huge. But it is not possible to have all of them accessible, so, it is necessary to group them in three different groups: Known and available signals, known but unavailable signals, and unknown and unavailable signals.

Inside the known and available ones, it can be found data that represents different signals coming from diverse aircraft systems, and the information has different characteristics:

- Signals related to the environmental control system like indication of the cabin pressure, air generation, temperature inside the cabin and more.
- Information related to the level of breathing air inside the cabin, the amount of auxiliary oxygen and its concentration.
- All the data associated to the hydraulic system of the aircraft like its pressure, the reservoir level and if any failure has been generated.
- Information about the different generators of power, including secondary power systems, inside the aircraft indicating any malfunction in any of them.
- Flight Control System data that indicates the aircraft's attitude like the angle of attack, pitch angle, yaw angle and more.
- Signals that give information about the flight condition of the aircraft like altitude and Mach number.
- Data representing the behavior of the electrical system.
- Information related to the engines and the propulsion of the aircraft.
- Signals containing data about the pilot's behavior.
- All the data which contains structural status of the aircraft and its systems.
- Information about what the different monitors were showing in a specific moment.
- Weapon's data: Weapon's quantity, status, and type.
- Any detail about extra cargo inside the aircraft and its status.
- Signals which indicate pilot's workload and all the steps performed by him.
- The condition of the pilot at a specific time.

The most important signals that are known but unavailable are the ones related to the health of the pilot, these signals are fundamental to be aware of any situation inside the cabin that can cause any failure that affect the overall performance of a mission.

2.3 Selection of the most suitable signals/data for each flight facet

Taking into account the available signals and their characteristics, it is possible to group them in different flight facets depending on the properties of the stored information and the common attributes shared by the signals. The proposed classification is the following:

- Cockpit System Facet: It contains all the signals that indicate the situation of the cabin during the flight.
- Hydraulic System Facet: It includes all the data that points out the behavior of all the systems related to the hydraulic field.
- Electric System Facet: It is composed of all the information that defines the functioning of all the electrical devices that compound the systems of the aircraft.
- Propulsion System Facet: It groups the data related to the engines and the propulsion of the aircraft.
- Flight Control & Guidance System Facet: It contains the information about the status of the Flight Control System.
- Flight Condition Facet: It shows the performance of the aircraft giving information about the attitude during the whole flight.
- Pilot Workload Facet: All the information about pilot's behavior and the actions performed by him is included in this facet.
- Pilot Condition Facet: It contains all the signals that specify how is the pilot at a specific instant.

2.4 Analysis Conclusions

In conclusion, after performing Debriefing process all the information about the signals is obtained. Since, the amount of signals is huge, and each one of them has its own characteristics, it becomes necessary to analyze and study them considering the information that they provide.

In order to reduce the amount of scattered information, Flight Facets are created to group all the signals that share common characteristics, and whose data is interrelated. It is important to recall that the Flight Facets explained in Section 2.3 are not the only ones, since it is possible to create additional ones depending on the signals selected and, also, to modify any of them to have another data inside. The proposed Flight Facets are created according to the available data of this specific project.

3 Proposal Description

The concept of Global Mission Assessment proposed in this project is composed of different components, which are defined and described in Section 3.1, and the assessment process is simply explained in Section 3.2.

Finally, the whole model is approached using Machine Learning process, whose steps are applied considering the data utilized in this project. All this process is explained in Section 3.3.

3.1 Components of Mission Assessment

The proposed model consists of a series of elements, which analyzed as a whole and taking into account the relationships between them, will describe how the mission is performing. Each of these elements is in turn composed of different related sub elements, from which the necessary information to be used to evaluate the mission is obtained.

The components that are part of the set are:

- **Signals:** As explained in the previous chapter, it is the information received from the different engraving systems of the aircraft that allow us to take out their values during a defined time interval.
- **Envelopes:** It is the area in which the values coming from the signals must be in order for the action to be classified as correct. The envelopes will be characterized by establishing limits that should not be exceeded if one wishes to ensure that the actions, and consequently the mission, are carried out correctly.
- **Flight Facets:** Subset of data that provides necessary information to describe a specific aspect or feature of the aircraft or the mission.
- **Actions:** Steps that must be followed in order to complete a task or a procedure.
- **Procedures:** Series of actions that must be performed when required to correctly fulfill a specific demand, they are usually collected in the flight manual and are executed in a time interval.
- **Tasks:** Set of actions that have to be executed during a period of time in order to achieve particular objectives of the mission.
- **Mission:** It is the main objective that must be fulfilled. The operations performed during the whole mission are classified in tasks and procedures.

Throughout the whole project these elements are going to be used repeatedly, then from this point forward, the following nomenclature is used to refer to each of them:

- $\boldsymbol{S}$: Signal
- $\boldsymbol{E}$: Envelope
- $\boldsymbol{FF}$: Flight Facet
  - $\boldsymbol{FF_1}$: Cockpit Flight Facet
  - $\boldsymbol{FF_2}$: Hydraulic Flight Facet
  - $\boldsymbol{FF_3}$: Electric Flight Facet
  - $\boldsymbol{FF_4}$: Propulsion Flight Facet
  - $\boldsymbol{FF_5}$: Flight Control & Guidance Flight Facet
  - $\boldsymbol{FF_6}$: Flight Condition Flight Facet
  - $\boldsymbol{FF_7}$: Pilot Workload Flight Facet
  - $\boldsymbol{FF_8}$: Pilot Condition Flight Facet
- $\boldsymbol{A}$: Action
- $\boldsymbol{P}$: Procedure
- $\boldsymbol{T}$: Task
- $\boldsymbol{M}$: Mission

3.2 Simple design for assessing a Mission

As it was explained in Section 1.2.1, the evaluation of a mission has been fundamental in the aeronautical field in order to improve the performance gradually and to allow the achievement of more complicated objectives. Due to this, it is necessary to establish an evaluation criteria to be followed during all the process.

The assessing of the missions is performed using the Task Based Management (TBM) criteria that consists in evaluating a whole process analyzing its sub-processes individually.

For this case, as we have different components inside a mission, the evaluation is done for each procedure and, within them, for each flight facet and action respectively. The assessing will be done selecting a time interval when an action is executed and the signals within that time are selected and analyzed. At that specific time, each of the signals has a precise state or value that must be examined to know if the data fulfill the requirements or it has a wrong value making the results to be inadequate. Figure 3 represents in a simple

way how an action's evaluation is related to the combination of each flight facet's assessing.

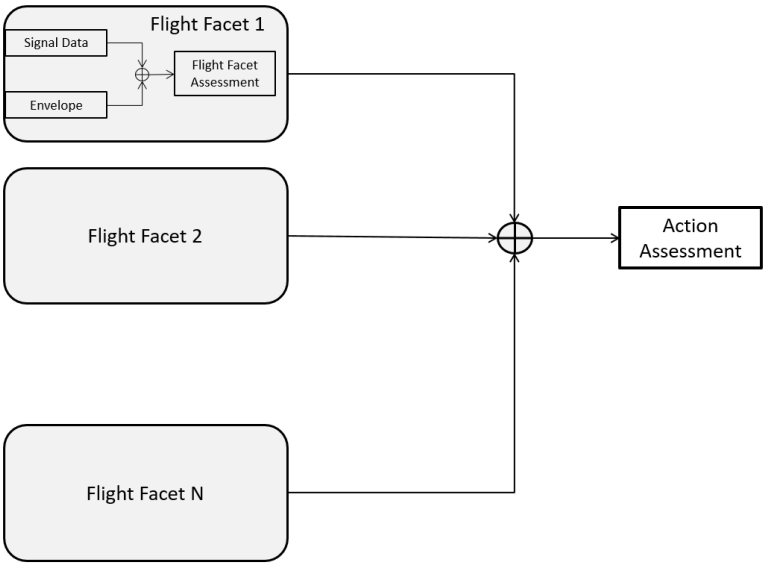


Figure 3: Evaluation of an action in a specific time interval.

Once the evaluation of all the actions has been done, it is possible to determine the result of the procedures and tasks that must be executed to complete the mission. Each one of the procedures and tasks has a particular weight that determines how each one of them influences on the mission. Figure 4 illustrates a diagram of the relation between a mission and its components.

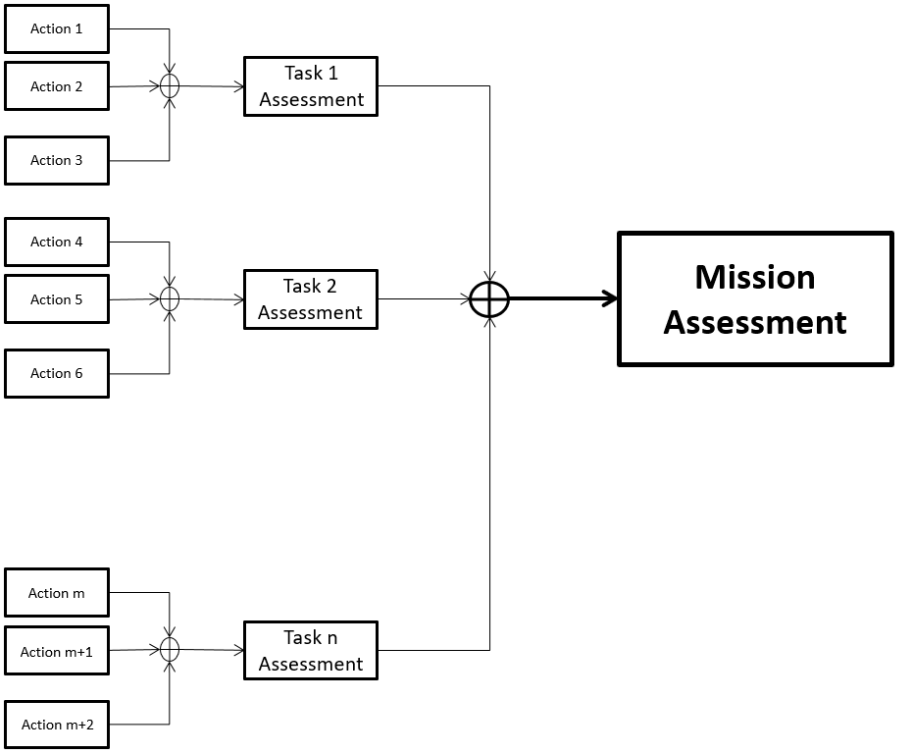


Figure 4: Scheme for the evaluation of a mission.

Mission assessments are collected and utilized as an input for Mission Planning. This system is capable of predicting the probability of success of a mission depending on the variables treated as inputs. One of the in puts is the final evaluation of all the missions carried out by the model, explained in Section 4.1. A simple block diagram, showing an example of some of the variables treated as inputs for the Mission Planning System, is represented in Figure 5.

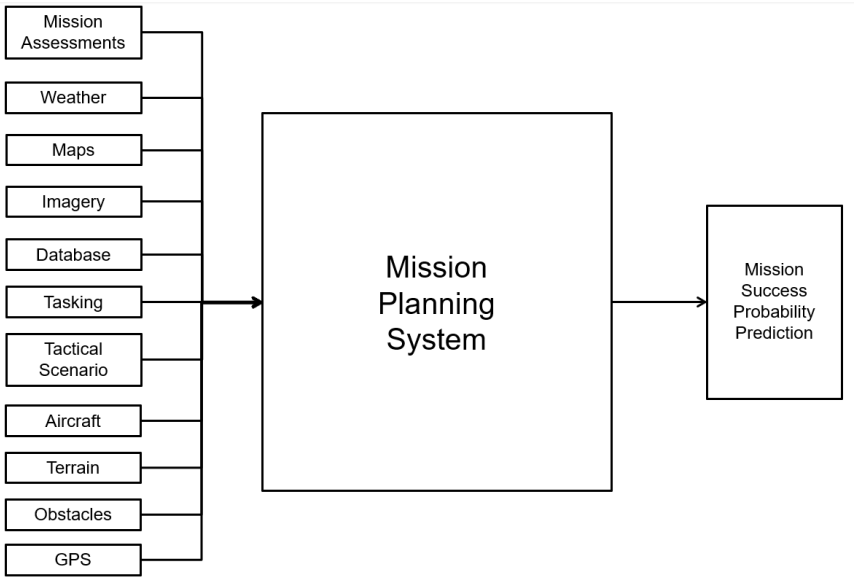


Figure 5: Block diagram for Mission Planning.

3.3 Approach to a refined model by means of Machine Learning algorithms

3.3.1 Machine Learning definition

Machine Learning is an application of Artificial Intelligence (AI) that makes a system to be able to learn from the data provided and to make use of that knowledge to achieve better solutions and/or decisions using prediction [7].

Machine learning enables analysis of massive quantities of data. While it generally delivers faster, more accurate results in order to identify profitable opportunities or dangerous risks, it may also require additional time and resources to train it properly. Combining machine learning with AI and cognitive technologies can make it even more effective in processing large volumes of information.

The procedure followed to learn is carried out by algorithms that work with the data and information supplied, in order to identify patterns and to make a decision based on the experienced acquired from the examples provided before. The main goal is to make computers able to learn with no human intervention at all [8].

In order to do the learning process effectively, there exists three categories inside Ma-

chine Learning algorithms [7]:

- **Supervised Machine Learning:** Its main objective is to learn a pattern from a set of data used as training and it will make predictions taking data sets which were not previously observed. This process consists in determining a mapping function that relates the input data with the output data, which are known. Once the mapping function is obtained using training data, the algorithm is applied to new input data to determine the value of the output. There are two types of Supervised Machine Learning problems depending on the type of the expected output variable: Classification if it is categorical and Regression if it is a real value. The main algorithms are: Linear Regression, Support Vector Machines, Logistic Regression, Naive Bayes, Linear Discriminant Analysis, Decision Trees and K-nearest Neighbor.

- **Unsupervised Machine Learning:** The data is not labeled and there is no previous information about identifying patterns or knowing the class they belong to. This learning type tries to create subgroups based on the data set exploration taking into account, i.e., similar characteristics within the data. Unlike the previous type, this one does not have information about the output, so, its objective is to show the best structure that best relates the input data. Also, there are two types of problems: Clustering is related to the group of the input data and Association related to detection of the input data rules. The main algorithms are: Hierarchical Clustering K-means Clustering, Mixture Models, DBSCAN, Local Outlier Factor, Neural Networks, Expectation-maximization algorithm, Principal Component Analysis and Non-negative matrix factorization.

- **Reinforcement Learning:** In this case, the goal is to obtain a system that improves its own learning while the surroundings is changing, that is, to deduce which path must be chosen among the different states in order to maximize the outcome. Along each step a path towards total reward is taken by the algorithm sequentially, the reward can be positive or negative. The final reward is the sum of all the positives and negatives ones throughout the path. The main algorithms are: Q-Learning, Policy Iteration State-Action-Reward-State-Action (SARSA), Deep Q Network and Deep Deterministic Policy Gradient.

Additionally, within AI applications there exists Deep Learning that is the next generation of Machine Learning. Models created using Deep Learning can perform predictions without the dependence of humans. These models use neural networks whose design is inspired by the human's brain neural network, thus, the data is analyzed with a logical procedure similar to the used by humans. The potential of Deep Learning comes from the ability to interpret data from the lowest level improving its behavior during time.

Also, it is able to take supervised and unsupervised algorithms. The main algorithms are: Convolutional Neural Networks, Artificial Neural Networks, Feedforward Neural Networks, Multiple Linear Regression, Gradient Descent and Logistic Regression.

These techniques are best appropriate to solve specific issues over others. These problems are: Repetitive, Data Intensive and Super Human Tasks.

A fundamental step in Machine Learning is the selection of the appropriate algorithm to be applied to the data set. This election depends on the data and its characteristics. So, It is important to start explaining how the data is mathematically represented to perform Machine Learning techniques.

The set of data is divided into a MxN matrix and a vector of size $Mx1$. The matrix contains all the collected information divided in columns called features and rows known as training instances. Every single training instance is a sample, whose information is defined by the value of each feature. The vector is called label or target label, it stores the result obtained from the features combination of each sample. An example of the matrix and the vector is represented in Tables 1 and 2, where:

- $\boldsymbol{F_N}$ is the feature number N.
- $\boldsymbol{TI_M}$ is the feature number M.
- a_{MN} is the value that feature M has in the sample N.
- r_1 is the result that the sample N has after combining all M features.

Table 1: Example of a dataset matrix.

	$\boldsymbol{F_1}$	$\boldsymbol{F_2}$	$\boldsymbol{F_3}$	$\boldsymbol{F_4}$	$\cdots$	$\boldsymbol{F_N}$
$\boldsymbol{TI_1}$	a_{11}	a_{12}	a_{13}	a_{14}	$\cdots$	a_{1N}
$\boldsymbol{TI_2}$	a_{21}	a_{22}	a_{23}	a_{24}	$\cdots$	a_{2N}
$\boldsymbol{TI_3}$	a_{31}	a_{32}	a_{33}	a_{34}	$\cdots$	a_{3N}
$\boldsymbol{TI_4}$	a_{41}	a_{42}	a_{43}	a_{44}	$\cdots$	a_{4N}
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ldots$	$\vdots$
$\boldsymbol{TI_M}$	a_{M1}	a_{M2}	a_{M3}	a_{M4}	$\cdots$	a_{MN}

Table 2: Example of a dataset vector.

$\boldsymbol{L}$
r_1
r_2
r_3
r_4
$\vdots$
r_N

Once the dataset is known and ready, the next step is to choose the algorithm that is going to be implemented. In order to select the appropriate one, it is important to consider some factors:

- Data characteristics, i.e. the amount of features and training instances, the type of the information and the rules that control the computation of the output. Additionally, the time taken by the algorithm to design the model and to obtain the result is an important factor which must be taken into account.

- Hyper-parameters. Each algorithm has its own parameters that must be tuned in order to determine the values that produce best scores in the prediction. If an algorithm has a lot of hyper-parameters, it requires more time to obtain the best combination, however it also provides greater versatility.

- Accuracy. Maybe the most important factor. Depending on the type of problem and on the algorithm applied, the value of the accuracy may change. It is fundamental to find a balance between accuracy and the rest of characteristics of the problem to establish which minimum value is the proper one.

After choosing the most suitable algorithms for the problem considered, the next action is to start with the training process. The dataset is divided into training set and test set. The training set is the one used by the algorithm to learn the behavior and rules that domain the computation of the final results. On the other hand, the test set will be employed to apply a final evaluation to the algorithm.

The fundamental problems that appear when training a Machine Learning algorithm with data similar to the one addressed for this problem are underfitting or overfitting [9]:

- Underfitting: This issue appears when the model implemented is very simple and the values obtained after training are not acceptable. The algorithm is losing the tendency of the data, thus is, it does not generalize the behavior appropriately. This kind of problem often occurs when a linear algorithm is implemented to work with a dataset that does not follow a linear distribution.

- Overfitting: This methodological error is defined as an over-training of the algorithm with data whose result is known, causing an extreme fitting to the specific dataset, preventing the algorithm from obtaining good predictions when the data introduced has slightly differences with the training data.

In order to prevent any of this issues, it is necessary to find the best way to evaluate the model and, then, to obtain its best performance. To be sure that the algorithm will compute good outputs cross-validation techniques are implemented, which will indicate how the behavior of the algorithm is when applying new non-observable data.

Cross-validation is a technique used to evaluate analysis results to guarantee that they are independent from the training set. It consists in splitting the dataset in two parts:

training subset and validation set. This division is equally distributed among both sets, in turn, the training set is split into smaller groups called folds. Some of these folds are employed to train the model and the result obtained is validated with the validation set, which will be composed of the remaining folds. This process is performed n times, depending on the number of splits required. The final score obtained using this method is the average value of each score obtained in every split. The most important cross-validation algorithms are: KFold, Stratified KFold, Shuffle Split and Stratified Shuffle Split [10].

In Figure 6 it is appreciated the way the split is performed. Each blue section is an independent fold used to train the model while the yellow one, as its name indicates, is utilized to validate the algorithm. In every iteration the validation block changes, so, all the blocks are trained and validated. This process allows the data to be tested completely.

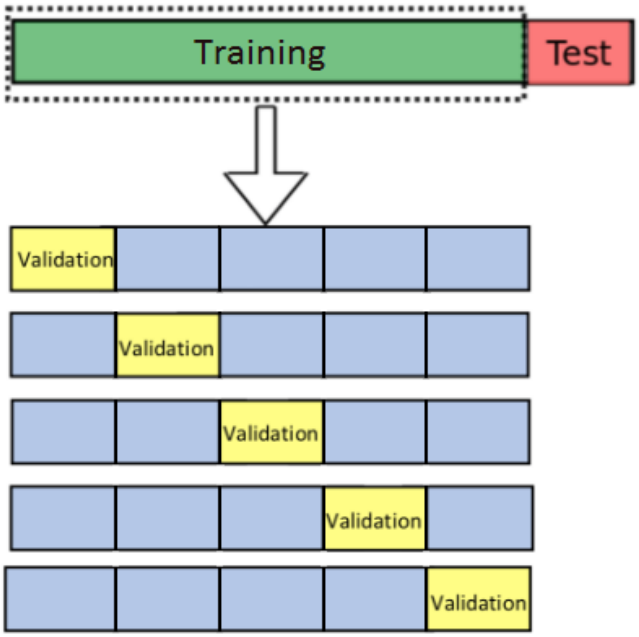


Figure 6: Cross-Validation process. Edited from [11].

After explaining how the validation works, the next step is to find and select the best hyper-parameters for the algorithms chosen. These parameters are different depending on the algorithm, accepting values within a specific range and they can affect highly in the performance of the model. The selection of the hyper-parameters can be tedious and can take a lot of time depending on the algorithm analyzed. Additionally, it is recommended

to apply cross-validation to obtain the best efficiency when getting the best value for each parameter. Then, the search of the optimal hyper-parameters consists in:

- The selection of the algorithm whose parameters are wanted to be found.
- To choose the appropriate parameter to be optimized.
- A method to perform the search.
- To apply the selected cross-validation technique.
- To establish the scoring criteria.

All these previous points depend on the type of problem addressed, so, they are explained deeply in Section 5.2 where the Machine Learning prototype for this project is defined and described.

3.3.2 Machine Learning Lifecycle

In order to apply Machine Learning techniques it is necessary to follow some steps within the Data Science [12]. The Lifecycle that must be followed has a lot of interpretations but its core is always the same: to achieve a specific objective. The aforesaid steps are:

1. **Business Understanding:** Before starting with a project, it is fundamental to understand the problem that is wanted to be solved. In addition, the main goals of the project must be identified together with the principal variables that are expected to be predicted. It is crucial to comprehend the strength of data and the way you can employ it to obtain successful results for your project.
2. **Data Mining:** Once the objectives were defined, the next step is to collect the data from different sources. This process can take a lot of time since finding the right data and information requires to seek how to obtain it and, also, to establish if the gathered data is useful for developing the project.
3. **Data Cleaning:** After collecting the required data, it is necessary to clean and prepare it. This process consists in detecting inconsistencies inside the information such as having different types of variables, or mixing integers with text instances. Also, it is important to check if there exists missing data and to determine if it is possible to fill or to ignore it. Finally, all the data must be uniform with the type of variables expected for the project. Due to these reasons, this step is the most time consuming one, sometimes it can take 50 to 80 percent of the total time of the whole lifecycle.

4. **Feature Engineering:** A feature is a specific property or characteristic of an observed event that can be measured. Feature engineering consists in modifying the raw data into meaningful features that exemplify the problem wanted to be solved, this phase is fundamental as it has a critical influence in the future predictions. This step is composed of two specific tasks:

    - Feature selection: Technique that consists in separating features that produce noise from the ones that add relevant information. This is applied to avoid difficulties when working with huge sets of features where a lot of them are not fundamental to determine the desired output. Some of the procedures used are: Filter methods, wrapper methods and embedded methods.
    - Feature construction: Involves the creation of new features from the ones that already exist. It also lies in merging multiple features to make this combination more informative.

5. **Training Modeling:** This stage has crucial importance, as it is going to determine how good the accuracy of the prediction will be. This process consists in dividing the collected data into train and test data. First, machine learning algorithms are applied to the training set to learn all the fundamental properties that define the data, then, the testing set is utilized to perform the prediction and, comparing the result obtained with the known one, it is possible to verify if the algorithm worked well or not.

6. **Predictive Modeling:** After making the validation of the solution obtained when training the model, all the characteristics (hyper-parameters) that compound the algorithm are known and it can be applied to a new set of inputs whose output is wanted to be known.

7. **Data Visualization:** Finally, the results obtained must be analyzed to determine if they make sense or if they are what was expected. Also, it is important to define how the results are shown to the audience, trying to use the best tools to make the visualization simple and straightforward.

Once the whole cycle is completed, the success of the developed model must be evaluated identifying problems or improvements that can be applied in successive iterations until reach the best possible model. Figure 7 shows the lifecycle explained before.

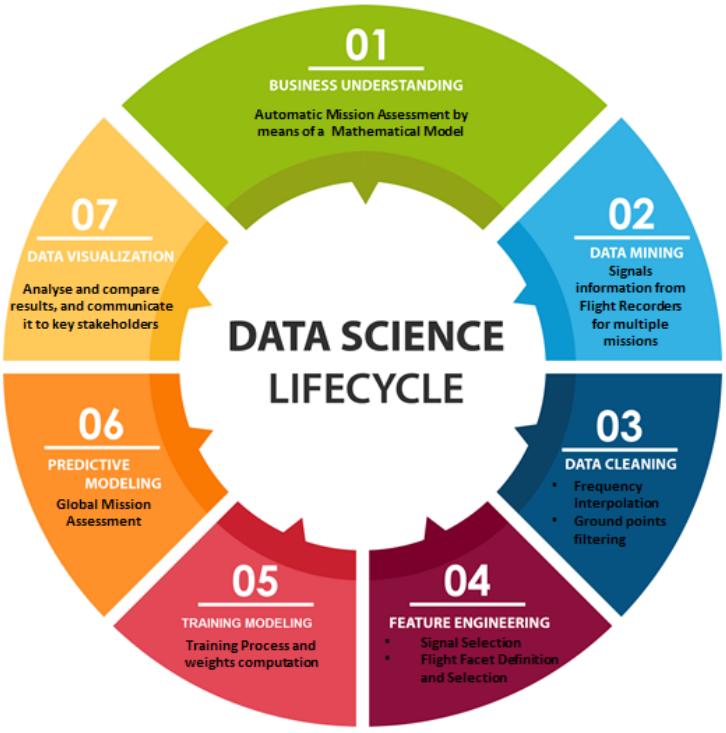


Figure 7: Data Science Lifecycle diagram.

As it can be seen in Figure 7, the phases of the lifecycle, previously defined in a general way, are now described taking into account this project:

1. **Business Understanding:** Automatic Mission Assessment by means of a Mathematical Model.
2. **Data Mining:** Signals information from Flight Recorders for multiple missions.
3. **Data Cleaning:** Frequency interpolation and ground points filtering.
4. **Feature Engineering:** Signal selection and Flight Facet definition and selection.
5. **Training Modeling:** Training process and weights computation.
6. **Predictive Modeling:** Global Mission Assessment.
7. **Data Visualization:** Analyse and compare results, and communicate it to the stakeholders.

Each one of these stages are explained in depth in Section 4.2.

4 System Design

As described in the Section 3.1 the proposed system is composed by different elements, each of them have a different weight inside the mission.

The system implemented is defined by means of an established criteria explained in Section 4.1, also the final representation of the model is shown and described in Section 4.2.

4.1 Description

As it was explained in Section 3.1, the system is composed of five principal components: Mission (M), Tasks (T), Procedures (P), Actions (A) and Flight Facets (FF). All these components are interconnected and interrelated. So, it is possible to define Mission Assessment as an hierarchical, successive and aggregate assessment of all the components, from Flight Facets to Mission. Figure 8 shows a simple diagram of the whole process.

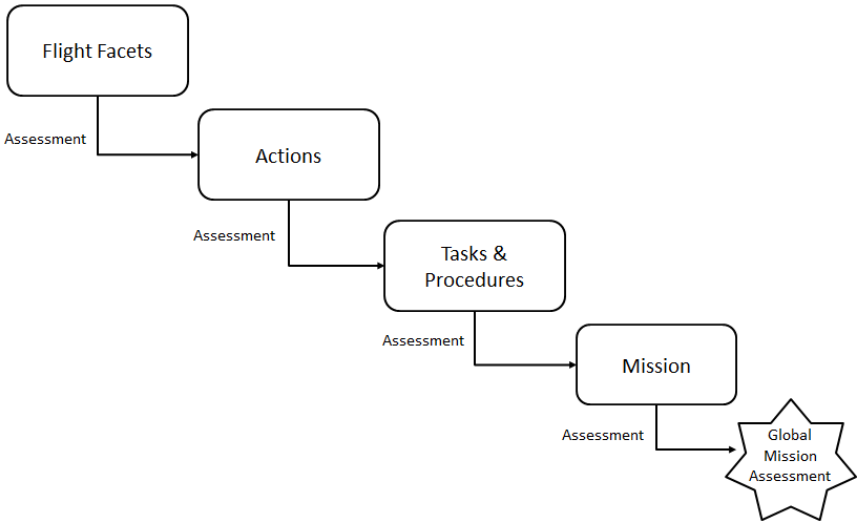


Figure 8: Mission Assessment process.

It is observed how the assessment of each element is dependent of the previous one, starting with Flight Facets, until achieving Global Mission Assessment. The components have different assessment methods, which are explained in the following sections.

Since mathematical formulation is applied to describe the whole model, it is important to highlight that the symbol $\Phi(x)$ represents the assessment of each component, where x symbolizes M, T, P, A or FF, depending on the element assessed. To conclude, the result obtained after performing the assessment of every part is represented as **1** if it is successful or as **0** if it is a wrong solution.

4.1.1 Flight Facet Assessment

In order to perform the assessment of the Flight Facets it is necessary to collect all the data. Each Flight Facet has a different assessment method depending on the information gathered.

The data provided are the signals that come from the different recorders placed in the aircraft. These signals, since they are from different aircraft systems, have diverse properties and values. For this reason, once the appropriate input has been extracted after cleaning the data, it is necessary to apply an adequate assessment process.

The method that must be followed to perform the assessment is included in the technical documentation. In these official reports, there are several ways to evaluate the recorded information according to the signals and their properties:

- Envelopes: They represent a limited area or region where the points obtained from the data have to be.
- Numerical limits: They represent a specific value that must not be exceeded.
- Numerical ranges: They specify a range inside which the points must lie.

Figure 9 is a simple diagram that illustrates how the combination of the recorded signals together with the evaluation methods provides the Flight Facet assessment.

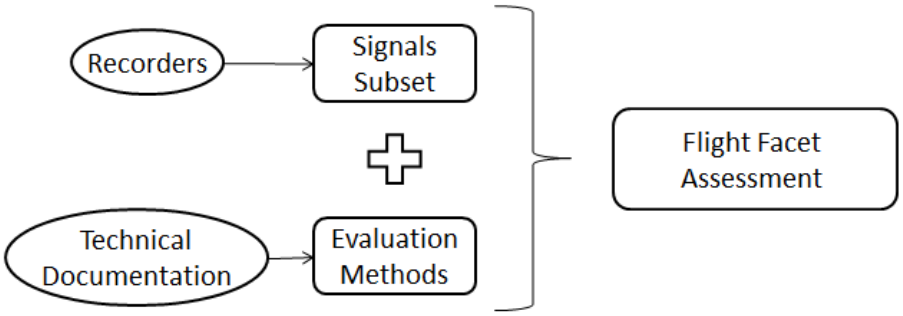


Figure 9: Flight Facet Assessment process.

The final result given by this assessment specifies if the signals, collected during a defined period of time, have the desired values and behavior or if they are outside the imposed limits. Additionally, this results will be the Action assessment inputs.

In order to explain how this assessment is performed, an example is presented. Table 3 shows how the data is stored after applying filtering methods to clean the data.

Table 3: Flight Condition data example.

Instance Number	**Altitude**	**Mach Number**
1	H_1	M_1
2	H_2	M_2
3	H_3	M_3
4	H_4	M_4
5	H_5	M_5
6	H_6	M_6
7	H_7	M_7
8	H_8	M_8
9	H_9	M_9
10	H_{10}	M_{10}
$\vdots$	$\vdots$	$\vdots$
N	H_N	M_N

As it is previously explained, it is necessary to have different evaluation methods. In Figure 10 an example of an envelope is illustrated. The black line represents the limits

of the original flight envelope while the blue line indicates the limits of another envelope created after knowing which are the best points where the best performance is achieved. The values for altitudes and mach number are normalized to be in percentage considering the highest and lowest values for each of the magnitudes.

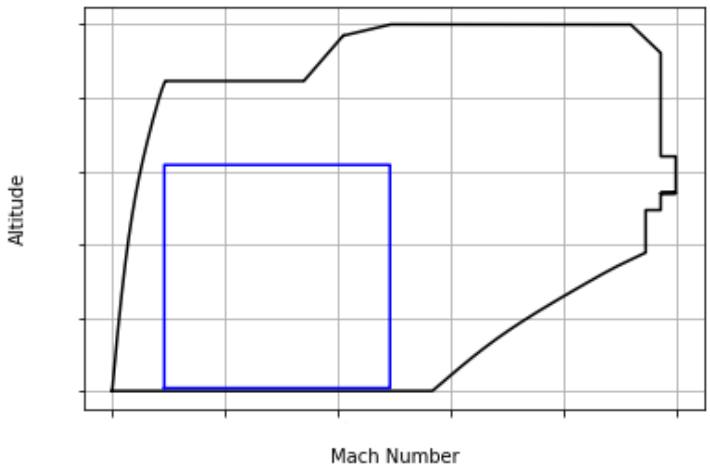


Figure 10: Example of an envelope.

Figure 11 shows an example of a dataset of the Flight Condition Flight Facet. It is observed how the majority of the points are inside the inner envelope, however, there are some that are outside. The objective is, given an action performed during an interval of time, to determine the status of the Flight Facet at a specific moment.

Figure 11: Example of Flight Facet data representation.

The criteria established to assess this type of Flight Facets is, given the time interval when the action is performed, to determine if the data points are inside or outside the inner envelope. For example, lets assume an Action is performed between t_1 and t_2, which are the initial and final times respectively, the data points of the Flight Facet are represented in Figure 12. It can be seen that all the points fall inside the inner envelope, then, the assessment of this Flight Facet for this Action is good.

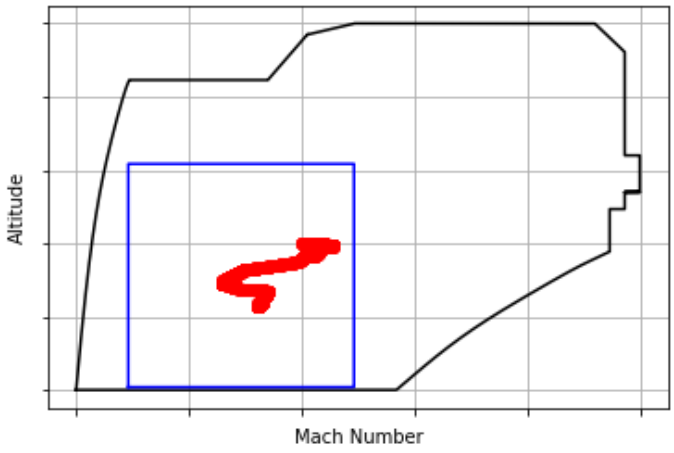


Figure 12: Data representation for successful assessment.

However, it can happen in the opposite way, an Action is performed between the times t_3 and t_4, and the majority of data points fall outside the inner envelope, as it can be observed in Figure 13. So, in this case, the assessment of this Flight Facet for that Action is unsuccessful.

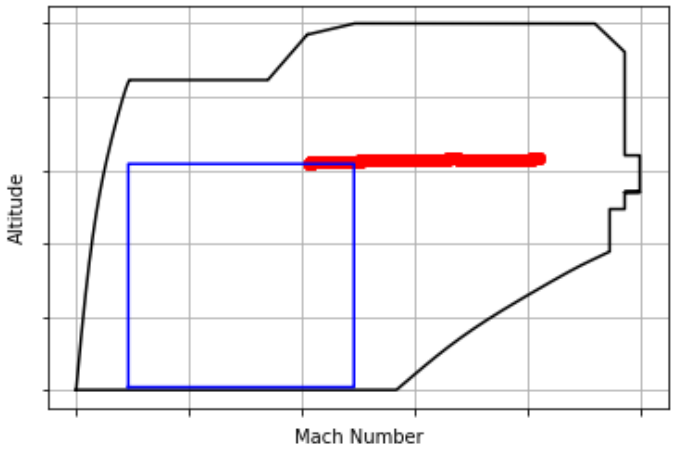


Figure 13: Data representation for unsuccessful assessment.

Then, this is one example of how Flight Facets are assessed, there are different ways depending on the data analyzed. However, the main idea is to establish an appropriate criteria to get the most adequate information and to avoid potential mistakes when assessing.

4.1.2 Action Assessment

The next step is to make the assessment of the actions. Each Action is performed during a period of time. Then, the assessments of every Flight Facet is analyzed for that specific time interval. Action assessment is formally defined as:

$$\Phi(A_j) = \frac{1}{N_{FF}+1}\left[\alpha_{pilot_j}\Phi(FF_{pilot}) + \sum_{K=1}^{N_{FF}} \alpha_{jk}\Phi(FF_k)\right] \tag{1}$$

where:

- α_{jk} is the weight (influence) that the Flight Facet k has in the assessment of Action j.
- N_{FF} is the total number of available Flight Facets.
- FF_{pilot} is the Flight Facet that describes the situation of the pilot.

- $\alpha_{j_{pilot}}$ is the weight (influence) that the pilot Flight Facet has in the assessment of Action j.
- $\frac{1}{N_{FF}+1}$ is applied to normalize the result obtained, getting a value between 0 and 1.

In Figure 14 another way to describe Action assessment is represented in order to display it in a more visual way.

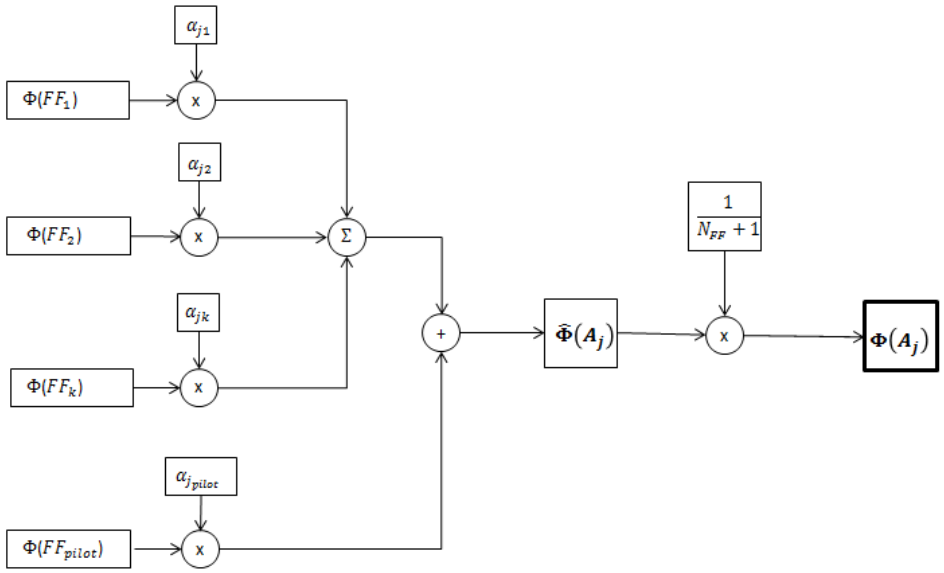


Figure 14: Action Assessment Flowchart.

It is important to recall that, from this point forward, FF_{pilot} is not considered in the development of the project, then, Equation 1 can be simplified, obtaining the following expression:

$$\Phi(A_j) = \frac{1}{N_{FF}} \left[\sum_{K=1}^{N_{FF}} \alpha_{jk} \Phi(FF_k) \right] \tag{2}$$

4.1.3 Procedure Assessment

Procedure Assessment is obtained taken into account the assessment of the set of actions that must be fulfilled to complete a specific demand. Then, following the same methodology as for action assessment, the assessment is mathematically expressed as:

$$\Phi(P_i) = \left[\sum_{i=1}^{N_{A_{P_i}}} \gamma_{ij} \Phi(A_j) \right] \frac{1}{N_{A_{P_i}}} \tag{3}$$

where:

- γ_{ij} is the weight (influence) that a specific Action j has in the Procedure i.
- $N_{A_{P_i}}$ is the total number of required actions that must be executed to complete the Procedure i.
- $\frac{1}{N_{A_{P_i}}}$ is applied to normalize the result obtained, getting a value between 0 and 1.

Figure 15 is a more visual representation of Procedure Assessment.

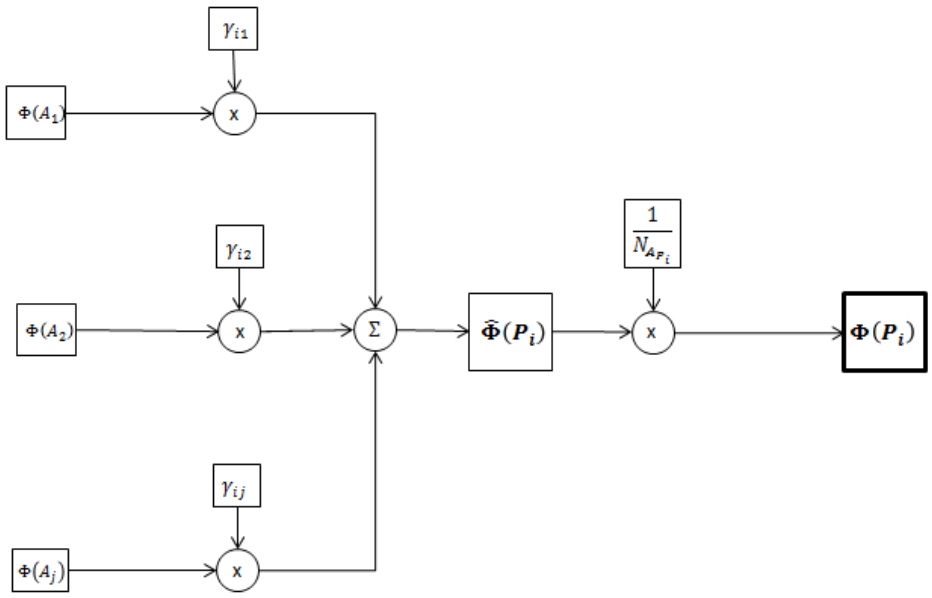


Figure 15: Procedure Assessment Flowchart.

4.1.4 Task Assessment

Task Assessment is computed following the same process than for Procedure Assessment, as it is a combination of Actions that must be completed. Then, the assessment of a Task is defined in Equation 4:

$$\Phi(T_i) = \left[\sum_{i=1}^{N_{A_{T_i}}} \beta_{ij} \Phi(A_j) \right] \frac{1}{N_{A_{T_i}}} \tag{4}$$

where:

- β_{ij} is the weight (influence) that a specific Action j has in the Task i.
- $N_{A_{T_i}}$ is the total number of required actions that must be executed to complete the Task i.
- $\frac{1}{N_{A_{T_i}}}$ is applied to normalize the result obtained, getting a value between 0 and 1.

Figure 16 is the visual representation of Task Assessment.

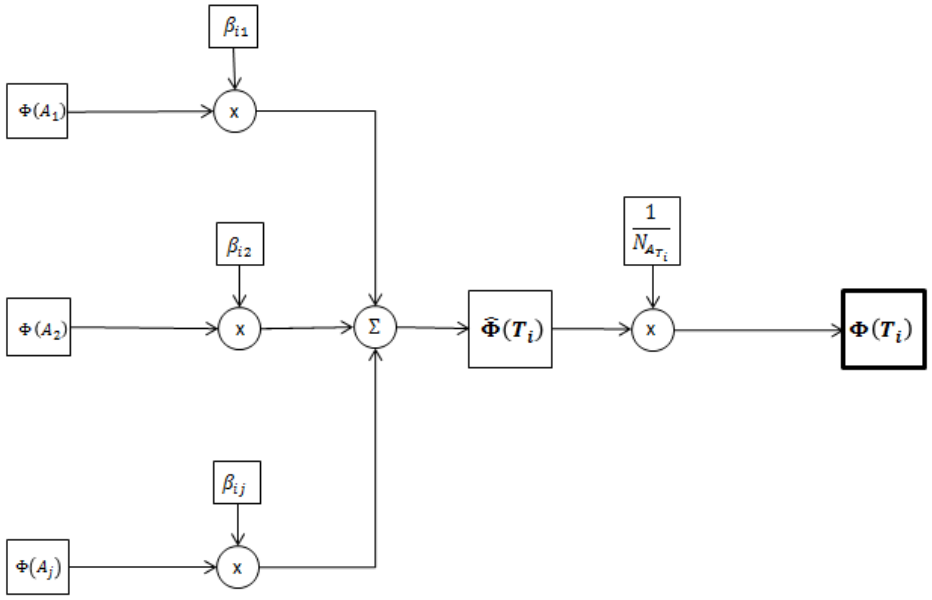


Figure 16: Task Assessment Flowchart.

4.1.5 Mission Assessment

To conclude, knowing the assessment of Procedures and Tasks, Mission Assessment is obtained using the Equation 5, considering that it is a combination of Procedure and Task Assessment and that the same process is followed:

$$\Phi(M) = \frac{1}{N_T} \sum_{i=1}^{N_T} \delta_i \Phi(T_i) + \frac{1}{N_P} \sum_{i=1}^{N_P} \epsilon_i \Phi(P_i) \tag{5}$$

where

- δ_i and ϵ_i are the weights (influence) that Task i and Procedure i respectively have on the performance of the whole mission.
- N_T and N_P are the total number of Tasks and Procedures respectively.
- $\frac{1}{N_T}$ and $\frac{1}{N_P}$ are applied to normalize the result obtained, getting a value between 0 and 1.

Finally, substituting Equations 3 and 4 into Equation 5, a final and more complex definition is achieved in Equation 6:

$$\begin{aligned}\Phi(M) = &\left(\frac{1}{N_T + N_P}\right)\left(\frac{1}{N_{FF}}\right)\left\{\sum_{i=1}^{N_T} \frac{\delta_i}{N_{A_{T_i}}} \left[\sum_{i=1}^{N_{A_{T_i}}} \beta_{ij} \left[\sum_{k=1}^{N_{FF}} \alpha_{jk} \Phi(FF_k)\right]\right]\right\} + \\ &+ \left(\frac{1}{N_T + N_P}\right)\left(\frac{1}{N_{FF}}\right)\left\{\sum_{i=1}^{N_P} \frac{\epsilon_i}{N_{A_{P_i}}} \left[\sum_{i=1}^{N_{A_{P_i}}} \gamma_{ij} \left[\sum_{k=1}^{N_{FF}} \alpha_{jk} \Phi(FF_k)\right]\right]\right\}\end{aligned} \tag{6}$$

A visual representation of Mission Assessment is depicted in Figure 17.

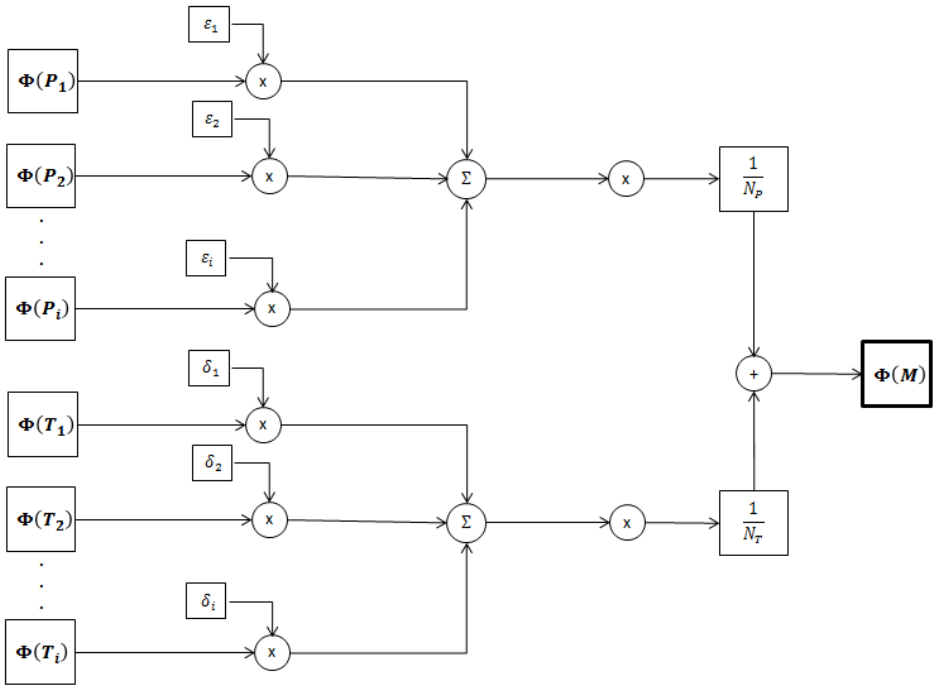


Figure 17: Mission Assessment Flowchart.

4.2 System Model

Once the Mission Assessment procedure has been defined, the next step is to develop a model which is able to determine the Global Mission Assessment using the process previously explained.

In order to perform the assessment, Machine Learning techniques are applied. Then, the general model explained in Section 3.3 is going to be implemented considering the type of problem addressed, as it was displayed in Figure 7.

So as to explain the process, an example mission is proposed to describe all the required steps that must be performed to follow the general model. The mission suggested is represented in Figure 18.

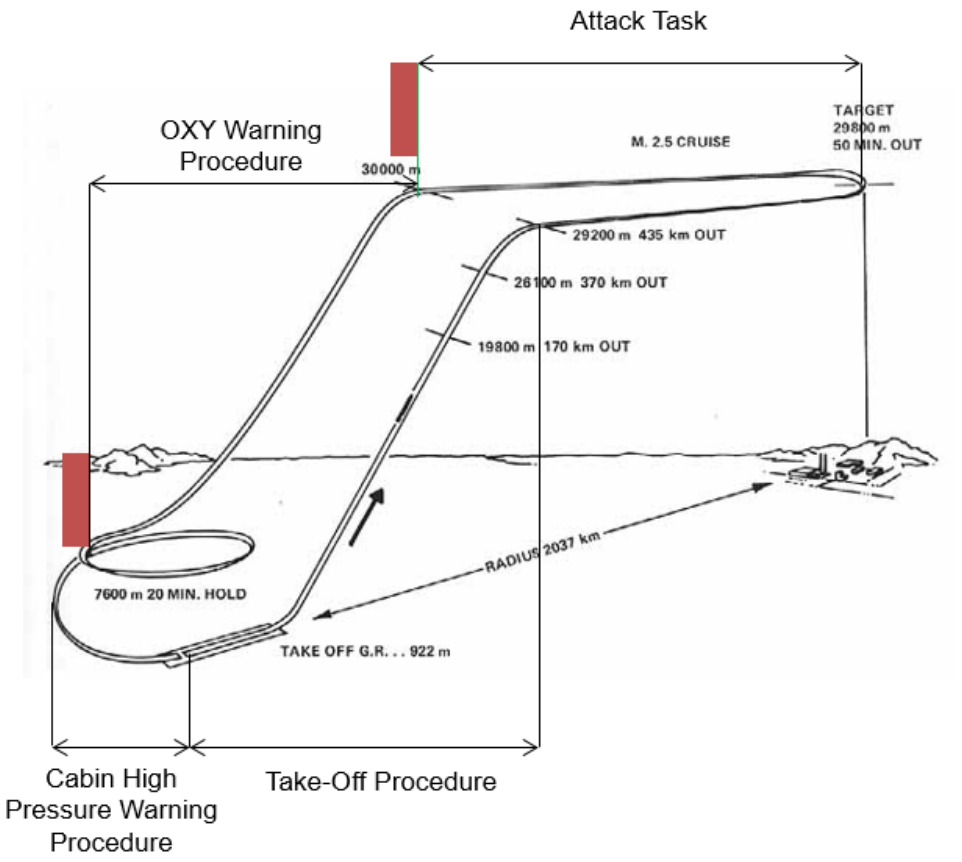


Figure 18: Proposed mission example.

As it can be observed, the mission's main goal is to carry out an attack to a specific objective. This duty must be performed following concrete tasks and procedures. In this case, the mission begins executing a Take-Off Procedure, when completed, an Attack Task is effected, suddenly, an OXY warning is detected and, then, the OXY Warning Procedure must be applied, while it is being realized, another warning shows up, the Cabin High Pressure Warning, and the Cabin High Pressure Warning Procedure is carried out to solve this issue. Once all the steps were performed properly, the aircraft lands and the mission can be defined as accomplished.

In order to make a better representation of the procedures and tasks executed during the mission, the mission's profile is depicted in Figure 19, in which, the time taken by each procedure and task together with their specific actions are also represented in the graph.

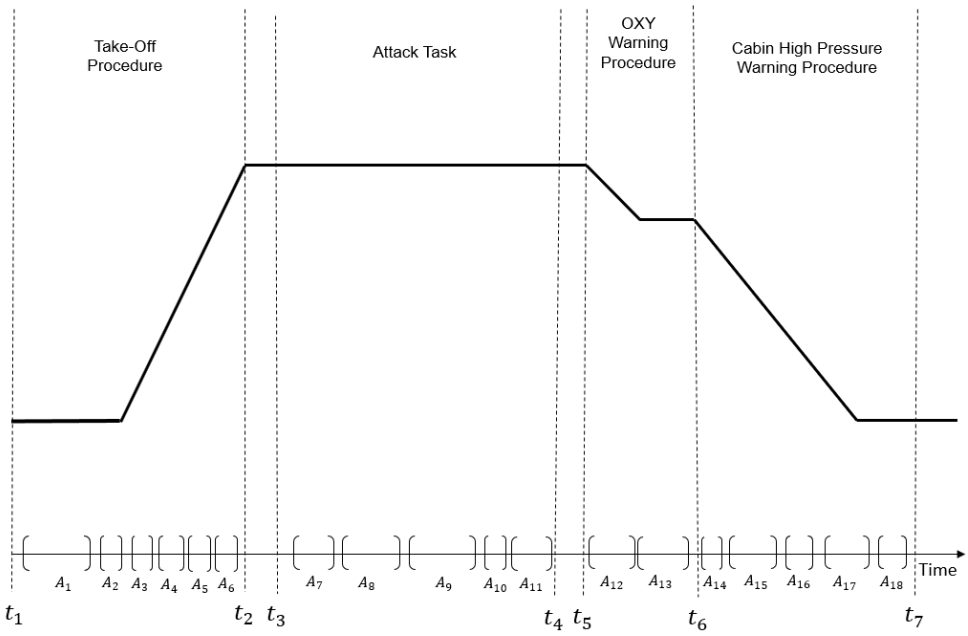


Figure 19: Proposed mission's profile.

The Task and Procedures, together with their Actions, of the proposed mission are explained and detailed below:

- Take-Off Procedure (From t_1 to t_2):
  1. Reach 240 KDAS in the runway (A_1).
  2. Pull-up nose to 25^o (A_2).
  3. Hide landing gear (A_3).
  4. Fighter take-off maneuver (A_4).
  5. Maintain a climb angles of 5^o (A_5).
  6. Reach an altitude of 25130 ft (A_6).
- Attack Task (From t_3 to t_4):
  1. Ingress attack zone (A_7).
  2. Attack maneuver (A_8).
  3. Attack positioning and weapon's engagement (A_9).

4. Release Bomb (A_{10}).
5. Egress Attack Zone (A_{11}).

- OXY Warning Procedure (From t_5 to t_6):
  1. Select Auxiliary Oxygen Bottle (A_{12}).
  2. Perform a descent until reach a specific altitude or below (A_{13}).
- Cabin High Pressure Warning Procedure (From t_6 to t_7):
  1. Turn off or reset Environmental Control System (A_{14}).
  2. Perform a descent until reach a specific altitude or below (A_{15}).
  3. Check if the RAM air fulfills the limits imposed by the Environmental Control System (A_{16}).
  4. To follow the process for Environmental Control System failure (A_{17}).
  5. Try to land as soon as possible (A_{18}).

Once the problem is described and explained, the general model defined in Section 3.3 is implemented considering the specific characteristics of the proposed mission. Then, each one of the steps described in Figure 7 is explained in the following sections.

4.2.1 Business Understanding

Once the problem to be solved is defined, the next step is to identify the objective of the project, the variables required, and the output expected.

In this particular case, the goal is to develop an Automatic Mission Assessment by means of a Mathematical Model. The model developed is the one explained in Section 4.1. As a remainder, Global Mission Assessment final expression is:

$$\Phi(M) = \left(\frac{1}{N_T + N_P}\right)\left(\frac{1}{N_{FF}}\right)\left\{\sum_{i=1}^{N_T}\frac{\delta_i}{N_{A_{T_i}}}\left[\sum_{i=1}^{N_{A_{T_i}}}\beta_{ij}\left[\sum_{k=1}^{N_{FF}}\alpha_{jk}\Phi(FF_k)\right]\right]\right\} + \\ + \left(\frac{1}{N_T + N_P}\right)\left(\frac{1}{N_{FF}}\right)\left\{\sum_{i=1}^{N_P}\frac{\epsilon_i}{N_{A_{P_i}}}\left[\sum_{i=1}^{N_{A_{P_i}}}\gamma_{ij}\left[\sum_{k=1}^{N_{FF}}\alpha_{jk}\Phi(FF_k)\right]\right]\right\} \tag{7}$$

From Equation 6, the main task is to determine the value of the weights: α_{jk}, β_{ij}, γ_{ij}, δ_i and ϵ_i, since they define the influence that each component has to determine the

assessment of the next one, from Flight Facet to Mission.

The data required to develop a solution to this problem are the signals stored in the flight recorders located along the aircraft, specifically, what they represent and their behavior. Since, these signals will give information about how the aircraft and its systems were at a specific time.

The output expected is the assessment of each one of the components until obtaining the Global Mission Assessment. As it was explained before, the final output given by the mathematical model developed will be **1** to point out that the mission (or any component) was executed correctly or **0** to indicate that it was completed unsuccessfully.

4.2.2 Data Mining

As it was mentioned, the data used to develop this model is the information obtained from the signals. The source of the signals are all the aircraft systems and they are transmitted through the buses inside the aircraft to the flight recorders, where all the information is stored.

4.2.3 Data Cleaning

Once all the data has been collected, it is necessary to prepare it to be used as the inputs of the model. The signals stored in the flight recorders

Once the first cleaning process has been performed, the data is gathered in a spreadsheet in order to have it more accessible and ready to be used in any moment. However, depending on the type of signal and its behavior, it will be necessary to apply another cleaning method.

In the case of this project, since the most important signals are the ones that give information about the status of the aircraft during the flight, the data wanted to be used is the one representing this behavior, hence, it is essential to ignore or to eliminate the information related to the aircraft when it is at ground.

4.2.4 Feature Engineering

This step consists in finding the features that best exemplify the characteristics expected for the data. This stage is fundamental, because the selection of the features has

an imperative influence in the performance of the predictive algorithm applied to the model.

This phase is crucial when the amount of features is very large and, probably, it is not necessary to use all of them. Among all the reasons to apply feature selection, the most important are:

- The training of the algorithm is faster.
- Feature selection makes the model easier to interpret and to apply.
- If the right features are chosen, the model improves its accuracy.
- The overfitting problem is reduced.

In this project, the features are the signals recorded from the aircraft systems. However, the amount of signals is huge and, even after performing the cleaning and filtering, there is still a lot of data that can be considered irrelevant when it is taken individually. Due to this reason Flight Facets were created by grouping together the signals that share common properties and have the same type of values, which are required to analyze an specific characteristic of the aircraft's attitude during a specific time interval. This process is executed following the concept Feature construction that was explained in Section 3.3.

As it was mentioned in Section 2.3, eight Flight Facets were created: Cockpit, Hydraulic, Electric, Propulsion, Flight Control & Guidance, Flight Condition, Pilot Workload and Pilot Condition. In order to verify if this way of combining the signals is appropriate to improve the performance of the model, feature selection/dimensionality reduction algorithms are applied on the whole set of features.

There exists three categories of these algorithms:

- Filter Methods: They score each feature applying statistical techniques, then, the features are ordered considering their scores and can be selected or removed from the dataset. The algorithms that belong to this category are: Pearson's Correlation, Linear Discriminant Analysis (LDA), Analysis of Variance (ANOVA) and Chi-Square.
- Wrapper Methods: They select the features making different combinations that are evaluated and compared to other ones. Then, a predictive algorithm is applied to evaluate that combinations and to give a score depending on the accuracy obtained. Main algorithms are: Forward Selection, Backward Elimination and Recursive Feature Elimination.
- Embedded Methods: They determine the features that provide best accuracy to the model while it is developed. Regularization methods are the most general type

of this technique, they introduce extra constraints into the predictive algorithm to optimize it, obtaining fewer features. Most popular methods are: LASSO and RIDGE regression.

Considering all the algorithms mentioned before, together with the characteristics of the problem to be solved, the method selected is Recursive Feature Elimination. It is a greedy optimization algorithm whose goal is to identify the feature subset which gives the best results. It makes models and takes the best or the worst feature at each step. Then, the process is repeated creating another model with the remaining features until all of them are analyzed. Finally, this algorithm classifies the features considering the order in which they where eliminated.

4.2.5 Training Modeling

Once the best features were selected, the next step is to execute the training process to make the model to learn the rules that are the basis of the current problem.

First, the assessment of the Flight Facets is obtained via the method explained in Section 4.1.1, hereafter, Machine Learning algorithms are implemented for each of the components of the system. Taking into account the problem faced and its characteristics, the best algorithms for this type of problem are:

- Support Vector Machines: Each data item is plotted in a space, whose number of dimensions is the number of features, being the value of the coordinate the value of the feature. Then, classification is performed to find the hyper-plane that separates the two classes identifying them. The main task is to find the correct hyper-plane, so, it is necessary to select the one that separates best the two classes and, also, the one that maximizes the distance from the closest point of each class to the hyper-plane [14].
- K-Nearest Neighbors: It is a method that simply searches the observations closest to the one that is trying to predict and classifies the point of interest based on the majority of data around it. It is an instance based algorithm, that is, it does not explicitly learn a model, it memorizes the training instances used as knowledge basis for prediction phase [15].
- Stochastic Gradient Descent: Gradient descent applies to the process of descending a slope until reaching the minimum point on a surface. It is an iterative algorithm which starts from a random point on a function and travels down ts slope in steps until it reaches the lowest point of that function [16].

- Logistic Regression: It is a statistical model that applies a logistic function to model the probabilities for classification problems with two possible outcomes [17].
- Random Forest Classifier: It is a type on ensemble learning method, that means, simpler models combine themselves to construct a robust model. This algorithm works growing several trees, then, each tree gives a classification and the tree proposes that class. Finally, the forest selects the classification that has more votes over all the trees in the forest [18].
- Gradient Boosting Classifier: The algorithm starts training a decision tree in which a weight is designated to each observation. Once the first tree is evaluated, the observations that are difficult to classify increase their weights, while the ones that are easy to classify decrease their weights. The next tree grows based on this weights, and this process is repeated for a desired number of iterations. Successive trees try to classify observations that previous trees did not. Lastly, the prediction given by the final model is the weighted sum made by the previous tree models [19].

In Section 5.2, the best algorithms for this project will be selected and tuned to demonstrate how this model works.

Starting with Action Assessment, let's assume that the same Action was performed in m Missions, and the assessment of each of the Flight Facets is known and represented in Table 4, where each of the features (columns) is one different Flight Facet and the training instances (rows) are the Action. Additionally, the final result of the assessment of these actions is evaluated by an expert and it is seen in Table 5, this column is known as the label.

Table 4: Same Action for different missions.

	$\boldsymbol{FF_1}$	$\boldsymbol{FF_2}$	$\boldsymbol{FF_3}$	$\boldsymbol{FF_4}$	$\boldsymbol{FF_5}$	$\boldsymbol{FF_6}$	$\boldsymbol{FF_7}$	$\boldsymbol{FF_8}$
$\boldsymbol{A_{1_1}}$	1	1	1	1	1	1	1	1
$\boldsymbol{A_{1_2}}$	1	1	1	0	1	1	0	1
$\boldsymbol{A_{1_3}}$	1	1	0	1	1	1	1	1
$\boldsymbol{A_{1_4}}$	1	1	1	1	1	1	1	1
$\boldsymbol{A_{1_5}}$	0	0	1	1	1	0	0	0
$\boldsymbol{A_{1_6}}$	1	1	1	1	1	1	1	1
$\boldsymbol{A_{1_7}}$	0	0	1	0	0	0	0	0
$\boldsymbol{A_{1_8}}$	1	1	1	1	0	1	1	1
$\boldsymbol{A_{1_9}}$	0	1	1	1	1	1	1	1
$\boldsymbol{A_{1_{10}}}$	1	1	1	1	1	1	1	1
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$\boldsymbol{A_{1_m}}$	1	1	1	1	1	1	1	1

Table 5: Expert Action's Evaluation.

$\boldsymbol{A_{1_{EVAL}}}$
1
0
1
1
0
1
0
1
0
1
$\vdots$
1

As it is observed in Equation 8, the only unknown is α_{jk}, then, the training process is performed to determine the value of this variable, since, it is going to establish how the Flight Facet k contributes to the assessment of the Action j. After executing the training to the previous data, the value of each one of the weights that influence in this specific action is obtained, Table 40.

$$\Phi(A_j) = \frac{1}{N_{FF}} \left[\sum_{k=1}^{N_{FF}} \alpha_{jk} \Phi(FF_k) \right] \tag{8}$$

Table 6: Flight Facets' weights.

	$\boldsymbol{FF_1}$	$\boldsymbol{FF_2}$	$\boldsymbol{FF_3}$	$\boldsymbol{FF_4}$	$\boldsymbol{FF_5}$	$\boldsymbol{FF_6}$	$\boldsymbol{FF_7}$	$\boldsymbol{FF_8}$
$\boldsymbol{\alpha_{1k}}$	α_{11}	α_{12}	α_{13}	α_{14}	α_{15}	α_{16}	α_{17}	α_{18}

In summary, Figure 20 shows the aforesaid process in a more visual way.

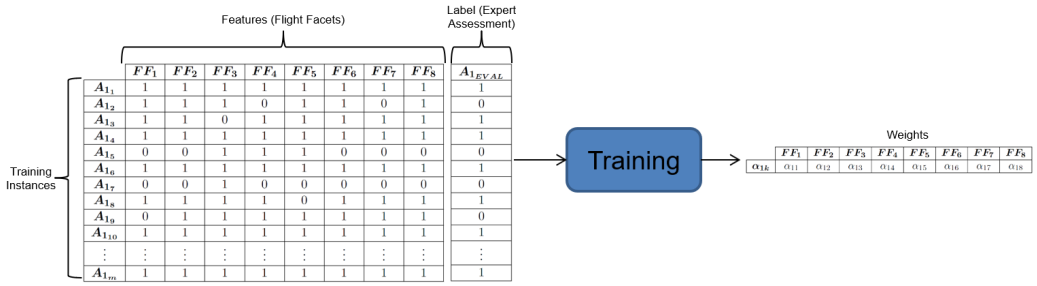


Figure 20: Action Training process.

Continuing with Procedure Assessment and following the same model, the same Procedure was executed for m Missions, and the assessment of the Actions is known. In this case, as it can be noticed in Table 7, the features (columns) are each of the actions required to fulfill the procedure and the training instances (rows) are the same Procedure performed in different missions. The label, represented in Table 8, is the evaluation given by the expert.

Table 7: Same Procedure for different missions.

	A_{14}	A_{15}	A_{16}	A_{17}	A_{18}
P_{1_1}	1	1	1	1	1
P_{1_2}	1	1	1	0	1
P_{1_3}	1	1	0	1	0
P_{1_4}	1	1	1	1	0
P_{1_5}	0	0	1	0	1
P_{1_6}	0	1	1	1	1
P_{1_7}	0	0	0	0	1
P_{1_8}	1	1	0	1	1
P_{1_9}	0	0	1	1	0
$P_{1_{10}}$	1	1	1	0	1
⋮	⋮	⋮	⋮	⋮	⋮
P_{1_m}	1	0	1	1	1

Table 8: Expert Procedure's Evaluation.

$P_{1_{EVAL}}$
1
1
0
1
0
1
0
1
0
1
⋮
1

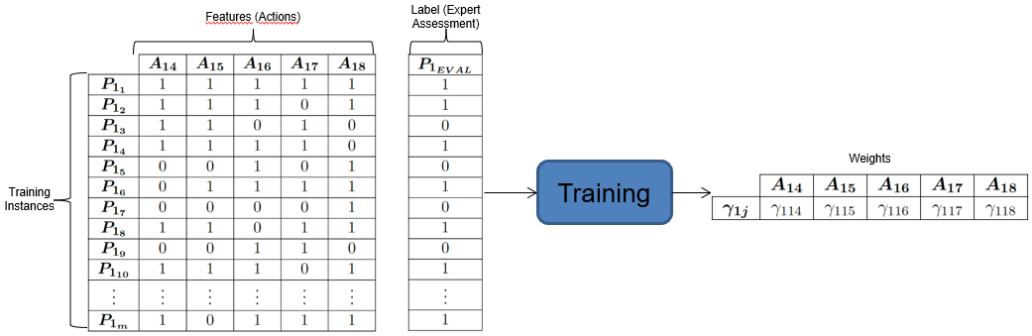
In this case study, the only unknown found in Equation 9 is γ_{ij}, which represents the influence that an Action j has in Procedure i. In the same way as for Action Assessment, the training model is implemented to obtain the value of the weights, Table 9, and to know how each Action has an effect on the Procedure.

$$\Phi(P_i) = \left[\sum_{i=1}^{N_{A_{P_i}}} \gamma_{ij} \Phi(A_j) \right] \frac{1}{N_{A_{P_i}}} \tag{9}$$

Table 9: Action's weights for Procedure.

	A_{14}	A_{15}	A_{16}	A_{17}	A_{18}
γ_{1j}	γ_{114}	γ_{115}	γ_{116}	γ_{117}	γ_{118}

To sum up, Figure 21 shows in a simpler way how the training process for Procedure Assessment is performed.



	A_{14}	A_{15}	A_{16}	A_{17}	A_{18}	$P_{1_{EVAL}}$
P_{1_1}	1	1	1	1	1	1
P_{1_2}	1	1	1	0	1	1
P_{1_3}	1	1	0	1	0	0
P_{1_4}	1	1	1	1	0	1
P_{1_5}	0	0	1	0	1	0
P_{1_6}	0	1	1	1	1	1
P_{1_7}	0	0	0	0	1	0
P_{1_8}	1	1	0	1	1	1
P_{1_9}	0	0	1	1	0	0
$P_{1_{10}}$	1	1	1	0	1	1
⋮	⋮	⋮	⋮	⋮	⋮	⋮
P_{1_m}	1	0	1	1	1	1

	A_{14}	A_{15}	A_{16}	A_{17}	A_{18}
γ_{1j}	γ_{114}	γ_{115}	γ_{116}	γ_{117}	γ_{118}

Figure 21: Procedure Training process.

The same method done for Procedure is applied to Task Assessment. In Table 10 the features are the Actions, these ones are not the same than the actions executed in a Procedure, and the training instances are the Tasks performed in different missions. The label shown in Table 11, as done previously, is the expert's evaluation.

Table 10: Same Task for different missions.

	A_7	A_8	A_9	A_{10}	A_{11}
T_{1_1}	1	1	1	1	1
T_{1_2}	1	1	1	0	1
T_{1_3}	0	1	0	1	0
T_{1_4}	1	0	1	1	1
T_{1_5}	0	0	1	0	1
T_{1_6}	1	1	1	1	0
T_{1_7}	0	1	1	0	0
T_{1_8}	1	1	0	1	1
T_{1_9}	0	1	1	1	1
$T_{1_{10}}$	1	1	1	0	1
⋮	⋮	⋮	⋮	⋮	⋮
T_{1_m}	0	1	1	0	1

Table 11: Expert Task's Evaluation.

$T_{1_{EVAL}}$
1
1
0
1
0
1
0
1
1
0
⋮
1

The unknown in Equation 10 is γ_{ij}, which symbolizes how a specific Action has an impact on the Task Assessment. After carrying out the training process the values of the weights, in Table 12, are obtained.

$$\Phi(T_i) = \left[\sum_{i=1}^{N_{A_{T_i}}} \beta_{ij} \Phi(A_j) \right] \frac{1}{N_{A_{T_i}}} \tag{10}$$

Table 12: Action's weights for Task.

	$\boldsymbol{A_7}$	$\boldsymbol{A_8}$	$\boldsymbol{A_9}$	$\boldsymbol{A_{10}}$	$\boldsymbol{A_{11}}$
$\boldsymbol{\beta_{1j}}$	β_{17}	β_{18}	β_{19}	β_{110}	β_{111}

To recapitulate, a straightforward version of the process followed to realize Task Assessment is illustrated in Figure 22.

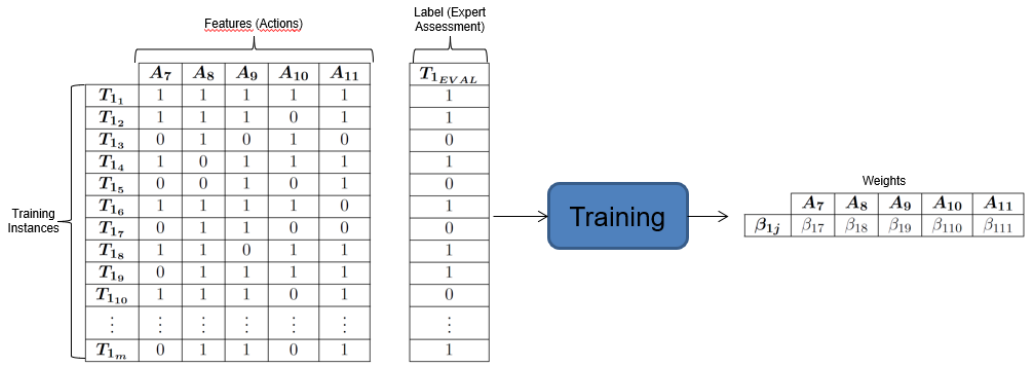


Figure 22: Task Training process.

Finally, with Procedure and Task assessments known, Mission Assessment can be obtained. In this case, m Missions were executed applying the same Tasks and Procedures, whose assessment was previously calculated. Table 13 represents as features (columns) the assessments for Task and Procedures, and as training instances all the missions with the same tasks and procedures. Also, it is necessary to have the evaluation, made by an expert, that specifies if the mission was successful or not, this is shown in Table 14.

Table 13: Mission with the same procedures and tasks.

	T_1	P_1	P_2	P_3
M_1	1	1	1	1
M_2	0	1	1	1
M_3	1	1	0	1
M_4	1	1	1	1
M_5	0	0	1	1
M_6	1	1	0	1
M_7	0	0	1	0
M_8	1	1	1	0
M_9	1	0	1	0
M_{10}	1	1	1	1
⋮	⋮	⋮	⋮	⋮
M_m	1	1	1	1

Table 14: Real Mission's Evaluation.

M_{EVAL}
1
1
0
1
0
1
0
1
0
1
⋮
1

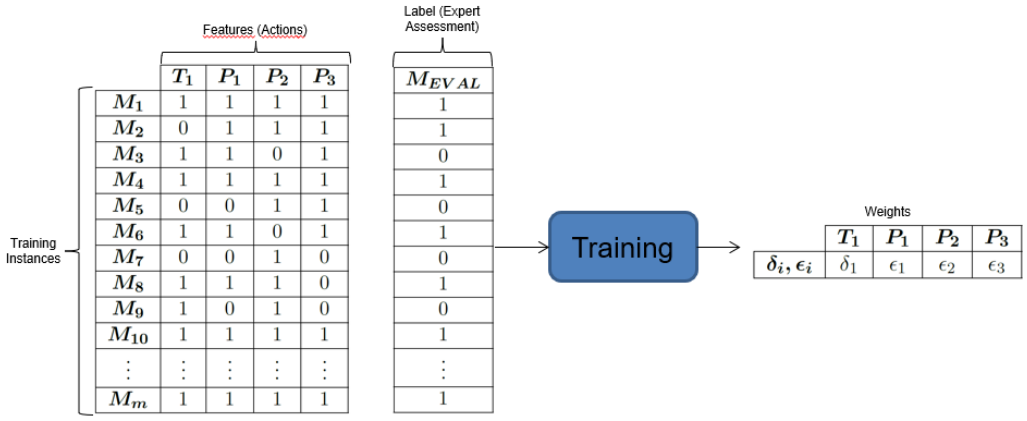
Equation 11, which determines Mission Assessment, has two unknowns δ_i and ϵ_i that define the influence that a Task and a Procedure, respectively, has on the assessment of a Mission. The training process is implemented to obtain the value of these parameters as shown in Table 15.

$$\Phi(M) = \frac{1}{N_T}\sum_{i=1}^{N_T}\delta_i\Phi(T_i) + \frac{1}{N_P}\sum_{i=1}^{N_P}\epsilon_i\Phi(P_i) \tag{11}$$

Table 15: Task's and Procedure's weights for Missions.

	T_1	P_1	P_2	P_3
δ_i, ϵ_i	δ_1	ϵ_1	ϵ_2	ϵ_3

As done for previous assessments, Figure 23 illustrates the process followed in a simple and summarized way.



	T_1	P_1	P_2	P_3	M_{EVAL}
M_1	1	1	1	1	1
M_2	0	1	1	1	1
M_3	1	1	0	1	0
M_4	1	1	1	1	1
M_5	0	0	1	1	0
M_6	1	1	0	1	1
M_7	0	0	1	0	0
M_8	1	1	1	0	1
M_9	1	0	1	0	0
M_{10}	1	1	1	1	1
⋮	⋮	⋮	⋮	⋮	⋮
M_m	1	1	1	1	1

	T_1	P_1	P_2	P_3
δ_i, ϵ_i	δ_1	ϵ_1	ϵ_2	ϵ_3

Figure 23: Mission Training process.

4.2.6 Predictive Modeling

After performing the training process, the algorithm has learned all the rules that compose the assessment of each one of the components, and the value of each one of the weights (α_{jk}, β_{ij}, $\gamma_{i,j}$, δ_i and ϵ_i) is calculated. The prediction consists in determining the result of the label with known values of the features.

Starting with Action Assessment, Table 16 contains the values of the features after performing their previous assessment. Then, as the value of each of the weights, α_{jk}, is known, the algorithm executes the prediction following the rules learned during the training, giving a final result as shown in Table 17.

Table 16: Flight Facets' Assessment.

FF_1	FF_2	FF_3	FF_4	FF_5	FF_6	FF_7	FF_8
1	1	1	0	1	1	1	0

Table 17: Result of applying prediction for an action.

$A_{Pred_{ASSESS}}$
1

The result obtained specifies that the Action was executed successfully even though

two Flight Facets had an unsuccessful assessment. This situation points out that those Flight Facets have a very low influence compared to the others since their assessment do not affect the final result of Action Assessment. Figure 24 represents the process explained.

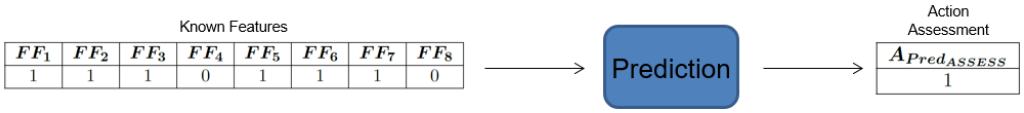


Figure 24: Action Prediction process.

The same process is applied to Procedure Assessment, the features are the actions required to fulfill the Procedure, represented in Table 18. Since the value of β_{ij} is known after executing the training, the algorithm is able to predict the Procedure Assessment as shown in Table 19.

Table 18: Action Assessment for prediction.

A_{14}	A_{15}	A_{16}	A_{17}	A_{18}
1	0	1	1	1

Table 19: Result of applying prediction for a procedure.

$P_{Pred_{ASSESS}}$
1

Looking at the result obtained, it is observed that the Procedure was fulfilled satisfactorily, while the second action was executed wrongly or not completed. So, it can be said that second action has not a high influence in the development of the procedure. Figure 25 is a simple chart representing the process followed to perform the prediction.

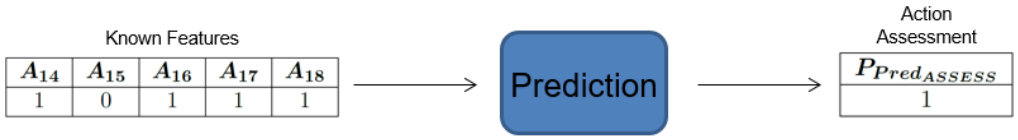


Figure 25: Procedure Prediction process.

The prediction of Task Assessment is performed following the same method than for Procedure Assessment. The features are the actions whose assessment is known (Table 20) and, by means of the trained algorithm, the value of $\gamma_{i,j}$ is defined and, then, the result for Task Assessment is obtained, as shown in Table 21.

Table 20: Task Assessment features.

A_7	A_8	A_9	A_{10}	A_{11}
1	0	1	0	0

Table 21: Result of applying prediction for a Task.

$T_{Pred_{ASSESS}}$
0

In this particular case, the result obtained means that the task was not performed correctly. Looking at the features, it is observed that only the second Action has an unsuccessful result influencing directly in the final result, so, this action has a huge influence in the final development of the Task. As done for Procedure Assessment, Figure 26 illustrates the process previously explained.

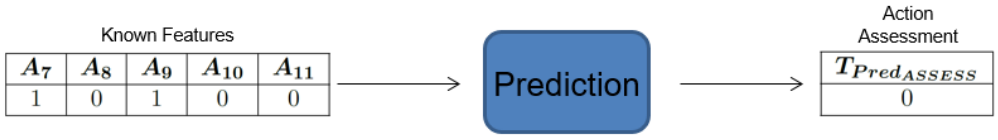


Figure 26: Task Prediction process.

Mission Assessment is performed taking into account the assessment of both procedures and tasks. The features, contained in Table 22, are the assessment of Procedures and Tasks, while Table 23 shows the result obtained after the trained algorithm, which previously computed the values for δ_i and ϵ_i, is applied.

Table 22: Mission Assessment features.

T_1	P_1	P_2	P_3
0	1	1	1

Table 23: Result of applying prediction for a Mission.

$M_{Pred_{ASSESS}}$
1

The result obtained indicates that the mission was completed successfully, despite having a wrong assessment in the case of the first task. This means that this task has not much influence in the development of the mission. In Figure 27 a chart of the process followed for Mission Assessment is depicted.

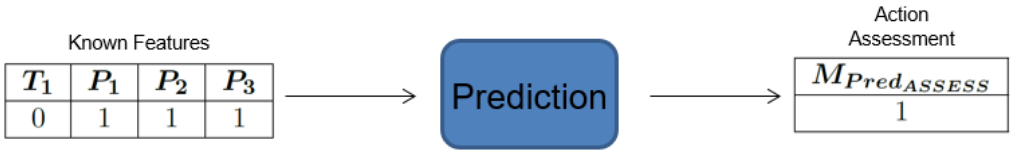


Figure 27: Mission Prediction process.

4.2.7 Data Visualization

This model has two main results: The weights (α_{jk}, β_{ij}, γ_{ij}, δ_i and ϵ_i) of each of the components of Mission Assessment and the value of the scores. Both results are shown in Section 5.3. The value of the weights are collected in tables and, in the case of Procedures, Task and Mission, bar graphs are included to show these values in a visual way. The scores are also listed in tables, however, they can be represented using confusion matrix, which is explained in Section 5.2.1.

Finally, the whole process of Global Mission Assessment is represented in a chart in Section 5.3.5 explaining the whole system when a new Mission Assessment is required to be predicted.

5 Prototype

As previously mentioned, the objective is to create a model that, once the result of the mission is known, describes the behaviour of each flight facet indicating the failure o failures that may have occurred during the development of the procedures followed by the pilot and that have affected the aircraft.

In order to carry out this process, it is necessary to create a prototype of the model that will be validated, evaluating it based on the results it will provide. This prototype will be made through the application of Machine Learning algorithms, in addition to others created specifically for this task that will be explained in detail in the following sections.

5.1 Available Technologies

Nowadays, there are many software programs that make it simpler to develop complex algorithms in a precise, fast and efficient way. In addition, the development of computers in terms of hardware and software has allowed to reduce the time of work, that would traditionally take several hours, to a few minutes, or even seconds.

There are several programs and libraries related to Machine Learning that allow these algorithms to be applied in an efficient, useful and fast way, and provide a lot of information on how to implement them. Among the most prominent, the following ones are found [20]:

- Scikit-learn: It is a software machine learning library for Python programming language that has a various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means and DBSCAN. It is designed to inter-operate with the Python numerical and scientific libraries: NumPy and SciPy.
- TensorFlow: It is an open-source toolkit that allows the generation of machine learning channels to build scalable systems to manipulate data. It simplifies the creation of statistical Machine Learning and deep learning solutions using its CUDA GPUs interface. As its name says, the most elementary data type is a multidimensional array called tensor. It supplies assistance and functions for Machine Learning applications like Computer Vision NLP and Reinforcement Learning.
- Shogun: It is and open-source machine learning library that offers a wide range of efficient and unified machine learning methods. It supports many languages (Python, Octave, R, Java/Scala, Lua, C#, Ruby, etc) and platforms (Linux/Unix, MacOS and Windows) and integrates with their scientific computing environments.

- Accord.NET Framework: It is a .NET machine learning framework combined with audio and image processing libraries completely written in C#. It is a complete framework for building production-grade computer vision, computer audition, signal processing and statistics applications even for commercial use. A comprehensive set of sample applications provide a fast start to get up and running quickly, and an extensive documentation and wiki helps fill in the details.

- Apache Mahout: It is a distributed linear algebra framework and mathematically Scala DSL designed to let mathematicians, statisticians, and data scientists quickly implement their own algorithms. Apache Spark is the recommended out-of-the-box distributed back-end, or can be extended to other distributed back ends.

- Apache Spark MLlib: Spark, besides being a strong data streaming platform, supplies various machine learning characteristics with MLlib. The platform provided, together with its APIs, allows users to apply machine learning techniques on data generated in real-time.

- Oryx 2: It utilizes Lambda Architecture to process Machine Learning in real time and widely. It eases the development of end to end model in the case of collaborative filtering, regression, classification along with clustering.

- H2O.ai: It provides a multi-layer artificial neural network, which is composed of various components and parameters that are liable to be modified based on the data collected. Also, it contains adaptive learning rate and rate annealing to produce greatly predictive results. It supports Convolutional Neural Networks and Recurrent Neural Networks.

- Pytorch: Its main features are Deep Neural Networks and Tensors. It enables the development of prototypes for research and the building of software pipelines.

- RapidMiner: It provides an unified and inclusive environment where diverse tasks, such as data preparation, deep learning, machine learning and predictive analysis, can be executed. Its most famous characteristic is lightning-fast speed to manage revenue, reduce costs and prevent risks. Besides creating models, it also can be used to optimize the performance of a model by bagging, boosting and building the model ensembles.

- Weka: It is the acronym of Waikato Environment for Knowledge Analysis. It is an open-source GUI interface, written in Java, whose easy implementation make possible to apply algorithms using a minimum number of programming lines. It is composed of various machine learning algorithms which can be implemented in any moment.

- KNIME: Known as Konstanz Information Miner is an open-source platform based on data analytics. It is easy to understand, helping the users to comprehend the data and

the implementation of data science workflows, and is always adding new components. In addition, it can combine several data sources to perform data analysis, modeling and data visualization with simple programming.

- Keras: It is a neural network library that can be implemented in Python. It is well-known for its modularity, speed and facility to use, additionally, it is often used for fast implementation and fast design of prototypes. Compared to previous libraries, Keras has an user-friendly interface which makes the implementation of neural networks straightforward.

Among all these software and libraries, the most used and known programming language is Python, which is slightly focused on object oriented programming. Python includes several libraries that are easily understable and very intuitive. they provide an useful interface to manage, process and visualize the dataset. Python is easy to use and it provides a better structure and format to large programming scripts than others programming languages.

The library, related to Machine Learning, that is utilized in the development of this project is Scikit-learn, which is one of the most used and popular open-source libraries nowadays. Scikit-learn is defined as a toolbox created to perform Machine Learning techniques and data mining. There are another scikit libraries, however, this package is composed of some dependant packages like Numpy and Scipy and it is also compatible with several operating systems. It also has an easy understable API that allows to use diverse modules by importing them for any project developed using Python. Among all its features, the main ones are [21]:

- Default values: All algorithms can be applied without configuring the parameters, since they have default values. This is used to perform an instant analysis to know how an algorithm behaves. On the other hand, these parameters can be adapted to obtain the best possible results.

- Consistency: Available objects share a solid and simple interface which contains methods that are the same for every object. Taking into account the problem addressed in this project, one of the required objects are the classifiers which are able to define object assignment processes and apply methods required to solve the problem. One of the main interfaces, related to classifiers, is prediction which makes possible to predict the output after creating the training model.

- No class modifications: Datasets are characterized as arrays or matrices, so, data structures can be processed directly by Scikit-learn, without performing any preprocessing technique.

- Data representation: In Scikit-learn, data is represented as a multi-dimensional array or using a dataframe structure, which is a 2D heterogeneous tabular data structure with labeled axes whose size is mutable.

- Data Flow: Data structures can be reused when the same dataset, even modified several times, is taken as an argument for the classifier.

5.2 Simple Prototype Implementation

In this Section the process followed to create a prototype is described. The prototype is developed considering the mission example described in Section 4.2 and represented in Figure 18. This mission is composed of one task and three procedures which, in turn, are divided in several actions.

First, since the evaluation of the algorithms is made by means of the score of the prediction, the different scoring functions are explained and the best one, according to the problem addressed, is selected. Then, all the possible appropriate algorithms are tested with default values to determine which ones are the best for this project and the selected ones are tuned by checking its hyper-parameters until reaching the combination that gives the best results.

5.2.1 Scoring Functions

There are several scoring functions to evaluate the performance of a classifier algorithm, which are able to make comparisons between them. These functions will make easier the selection of the best algorithm for a specific problem.

In order to understand these functions, it is necessary to explain how they are calculated, therefore, the confusion matrix is introduced. This matrix is a representation of how the performance of the algorithm was. It shows four different values: True Positives (TP), False Positives (FP), True Negatives (TN) and False Negatives (FN). Figure 28 illustrates an example of how this matrix is displayed [21].

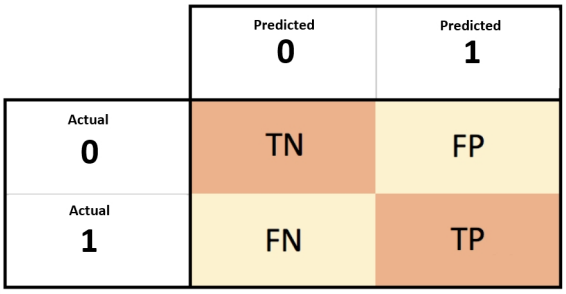


Figure 28: Confusion Matrix.

As it is observed, this matrix example represents a binary classification, like the one expected for this project. Its rows are the known values for the label while the columns indicate the label values obtained after performing the prediction. The definition of each one of the parameters is:

- True Positives (TP): Number of outputs that were classified in their correct positive class.
- False Positives (FP): Outputs classified as positives when their real value is negative.
- True Negatives (TN): Number of outputs that were classified in their correct negative class.
- False Negatives (FN): Outputs classified as negatives when their real value is positive.

Once all the previous parameters are defined, it is possible to define the different scoring functions that exist, depending on which of the previous values is wanted to optimize:

- Accuracy: It is the ratio of the true values to the total number of outputs. It is represented mathematically as

$$Accuracy = \frac{TP + TN}{(TP + FP + TN + FN)} \tag{12}$$

  which gives a value between 1 (best) and 0 (worst). This function is appropriate when the number of the samples are the same for both classes. If the amount of samples in one class is predominant, it is possible to have a lot of FP or FN while the value of the accuracy is close to 1. So, in these cases, it can be problematic to use this scoring function.

- Precision: It is a measure of how the classifier does not compute as positive an output that is considered as negative. It is expressed formally as

$$Precision = \frac{TP}{(TP + FP)} \tag{13}$$

the value obtained is considered good the closer to 1 it is.

- Recall: It represents the ability of the classifier to obtain all the positive samples. It expressed mathematically as

$$Recall = \frac{TP}{(TP + FN)} \tag{14}$$

whose result is between 1 and 0, being 1 the best value.

- F1: It is interpreted as the weighted harmonic average of recall and precision. Its mathematical formula is

$$F1 = \frac{2 \cdot Precision \cdot Recall}{(Precision + Recall)} \tag{15}$$

with 1 being the best result while 0 is the worst one. This function can offer a more realistic measure of the behavior of a test since it uses precision and recall.

Taking into account the problem addressed in this project, the appropriate functions are Precision, since evaluating any component of Mission Assessment as successful when it was a failure is very dangerous for future applications, and F1, as it takes into account two scoring functions to obtain its result.

5.2.2 Cross-validation technique selection

As it was explained in Section 3.3.1, there are four principal cross-validation techniques, which are: KFold, Stratified KFold, Shuffle Split and Stratified Shuffle Split [21]. The data used to represent the behavior of each technique is the Action's training set.

- KFold: The training set is split into K subsets, one of them is used as validation and the rest K-1 as training subsets. This process is performed K times and the final score is computed as the average of the results of each of the divisions performed. Figure 29 shows how KFold is applied to the dataset of this project, blue color represents the training set and the brown one is the validation set. The assessment illustrates the value of the expert's evaluation, being 0 the light blue color and 1 the brown one.

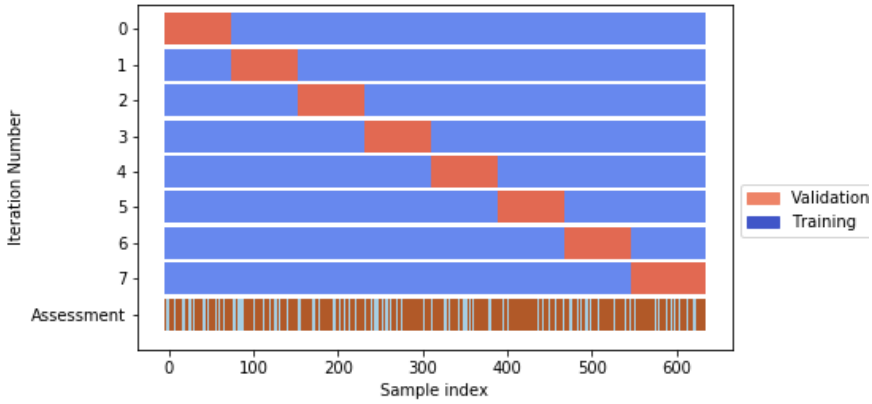


Figure 29: KFold technique.

- Stratified KFold: It is a modification of KFold and it assures that each class is equally represented in each of the divisions made. Since many algorithms classify by weighing the trained classes, those technique tries to make the fairest weighing possible. Figure 30 illustrates how this technique works for this specific data.

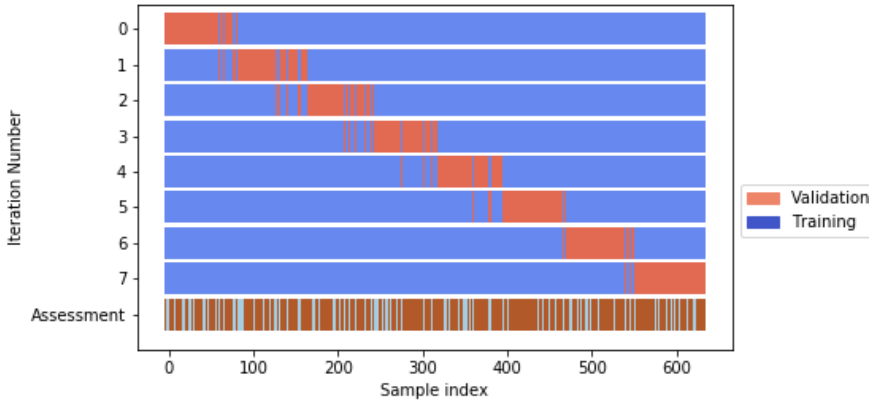


Figure 30: Stratified KFold technique.

- Shuffle Split: It generates a random number of training and validation sets. As the name suggests, all the samples are shuffled and split into those sets. The behavior of Shuffle Split is observed in Figure 31.

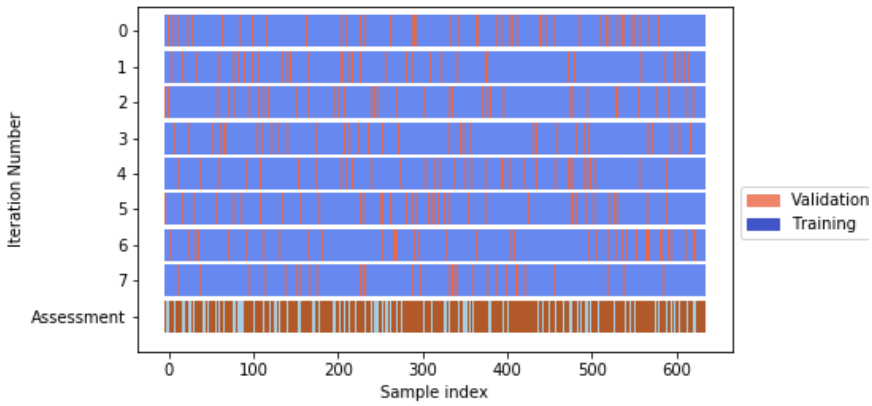


Figure 31: Shuffle Split technique.

- Stratified Shuffle Split: It combines Stratified KFold and Shuffle Split to have a more proportional rate for all the classes for both the training and validation set. In this way, it is possible to ensure that the algorithms are going to fit a training set where all the classes are represented. Figure 32 is a representation of this cross-validation method applied to project's data.

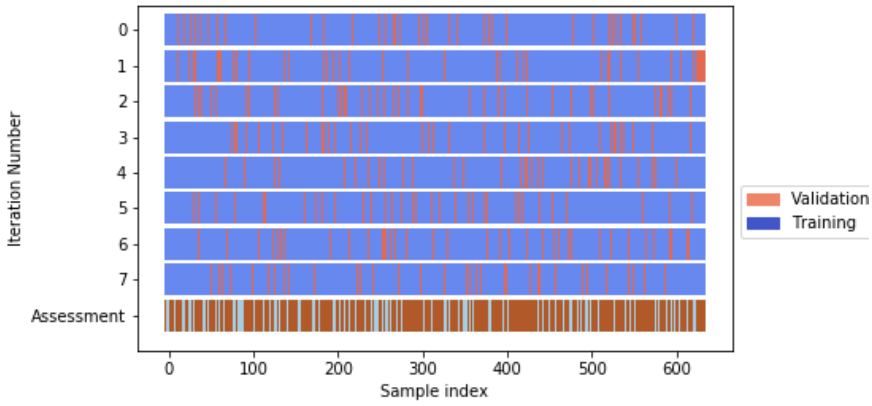


Figure 32: Stratified Shuffle Split technique.

5.2.3 Actions: Selection of the best algorithms and hyper-parameters tuning.

The main classifier algorithms to be applied in this project are the ones explained in Section 4.2.5. Then, a preliminary test to each of them was performed keeping their parameters as default, in order to check how good they are for the problem addressed.

Both the preliminary test and the tuning of the hyper-parameters are performed using the Action datasets which contain the assessment of the Flight Facets. Figure 33 shows the results obtained after applying the algorithms for the mentioned datasets, that have information from 10 to 1000 missions (making steps of 10). The algorithms are: Support Vector Classifier (SVC), K-Nearest Neighbor (KNN), Random Forest Classifier (RFC), Gradient Boosting Classifier (GBC), Gaussian Naive Bayes (GNB), Stochastic Gradient Descent (SGD) and Logistic Regression (LR).

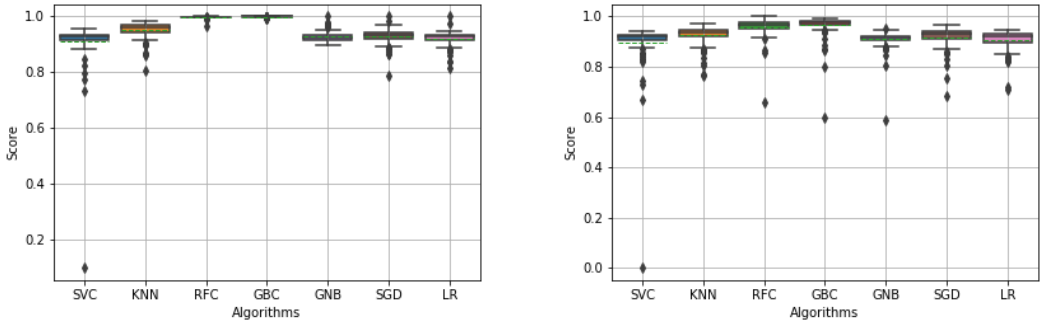


Figure 33: Actions scores of the training set (a) and for the test set (b).

The representation of each algorithm is performed by means of a box plot which is a chart used to visualize how numerical data is distributed by displaying data quartiles and average. It is useful to instantly identify how the data is dispersed, the mean values and the signs of skewness. A representation of this chart is depicted in Figure 34.

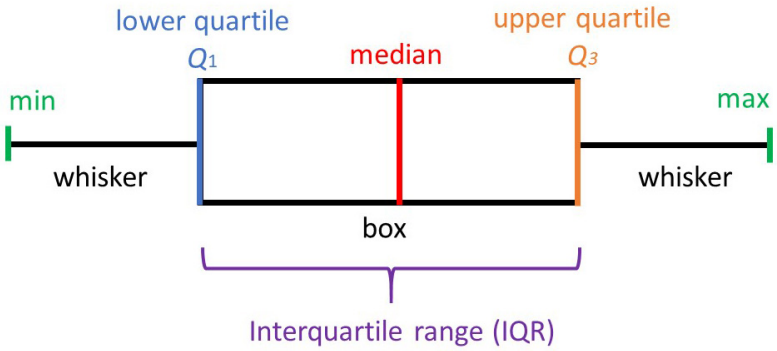


Figure 34: Box plot representation.

The definitions of the different elements of a box plot are [22]:

- Interquartile Range (IQR): It represents the box and it contains the middle 50% of the samples.
- Median: It symbolizes the mid-point of the data and is represented as a line dividing the box into two sections. It indicates that half of the points are greater than or equal to its value and the other half are less or equal.
- Lower or First Quartile (Q_1): 25% of the samples fall below this value.
- Upper or Third Quartile (Q_3): 75% of the samples fall below the value it represents.
- Whiskers: Both whiskers symbolizes the values of the samples that are outside the middle fifty percent, i.e. the upper and lower 25%.
- Minimum Score: The lowest value obtained not counting outliers.
- Maximum Score: The highest value of the samples without considering outliers.
- Outliers: They are values which are numerically distant from the rest of the data, and they are plotted outside the whiskers of the chart.

In Tables 24 and 25, the results for the training set and the test set respectively are shown numerically to facilitate the observation of the exact result for each of the elements of the box plot, among with extra characteristics, for all the algorithms.

Table 24: Action's training set results' main characteristics.

	SVC	KNN	RFC	GBC	GNB	SGD	LR
Number of samples	99	99	99	99	99	99	99
Mean	0.9108	0.9508	0.9985	0.9992	0.9276	0.9301	0.9218
Standard deviation	0.0890	0.0288	0.0035	0.0013	0.0174	0.0264	0.0256
Minimum value	0.1	0.8042	0.9667	0.9939	0.8957	0.7883	0.8155
25%	0.9143	0.9434	0.9984	0.9991	0.9174	0.9221	0.9152
50%	0.9281	0.9609	0.9991	1	0.9258	0.9337	0.9289
75%	0.9358	0.9681	0.9997	1	0.9337	0.9422	0.9352
Maximum value	0.9547	0.9856	1	1	1	1	1

Table 25: Action's test set results' main characteristics.

	SVC	KNN	RFC	GBC	GNB	SGD	LR
Number of samples	99	99	99	99	99	99	99
Mean	0.8980	0.9253	0.9581	0.9660	0.9090	0.9188	0.9065
Standard deviation	0.1007	0.0391	0.0376	0.0466	0.0389	0.0407	0.0407
Minimum value	0	0.7667	0.66	0.6	0.585	0.6845	0.7067
25%	0.9062	0.9199	0.9517	0.9678	0.9064	0.9107	0.8961
50%	0.9191	0.9374	0.9652	0.9774	0.9156	0.9311	0.9210
75%	0.9271	0.9479	0.9768	0.9831	0.9237	0.9395	0.9301
Maximum value	0.9409	0.9695	1	0.9911	0.9514	0.9667	0.9461

Taking into account the results obtained, the algorithms that have a better performance with the dataset used for this project are Random Forest Classifier (RFC) and Gradient Boosting Classifier (GBC). The next step is to tune the algorithms by analyzing the hyper-parameters that make them work.

Hyper-parameters tuning can be performed using two methods: grid search and random search. The first one is defined as a thorough search of the optimal parameters through a set of possible values previously specified. When using this method is recommendable to apply cross-validation to obtain the best possible performance. The second one performs a random selection of the values instead of electing them manually in a subset. The method used in this project is grid search by means of the Python's function **GridSearchCV**.

Starting with Random Forest Classifier, the first step is to observe how the algorithm behaves depending on the number of missions, which is the value that will determine the

number of samples of the dataset. Figure 35 shows how the score of the algorithm changes with the number of missions, i.e. number of training instances. It is seen how the score grows with the number of missions until reaching an almost constant value, this behavior is due to the fact that when number of missions is low there are very few training instances and the algorithm is not able to learn the rules that govern the results. The star points out the number of missions selected to tune all the hyper-parameters that rule this algorithm.

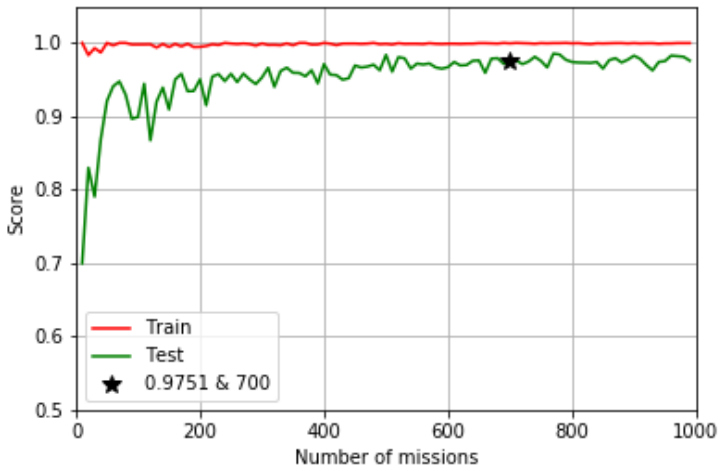


Figure 35: Random Forest Classifier's behavior with the number of missions for Action.

Among all Random Forest's hyper-parameters, there are some which do not have any effect on the performance of the algorithm for this type of problem, so, these parameters have their default value. However, the following hyper-parameters are the ones whose best value has to be determined:

- *n_estimators*: It specifies the number of trees that exist in the forest. The greater its value, the better the capacity to learn, however, having a great amount of trees makes the training process to be slower.
- *max_depth*: It indicates the value of the depth of the trees of the forest, as this number increases, the amount of splits is higher, allowing a better acquisition of the data's information.
- *min_samples_split*: It is defined as the minimum quantity of samples needed to split an internal node.

- *min_samples_leaf*: It specifies the minimum number of samples that are located at a leaf node.
- *min_weight_fraction_leaf*: It is the minimum weighted fraction of the addition of all the necessary weights which are at a leaf node.
- *min_impurity_decrease*: It indicates the threshold to split nodes when the impurity is above or below the given.

After performing the grid search, Table 26 shows the final value of each of the hyper-parameters. These values where selected because they obtain the best score value for each one of them. All the figures, which show the impact of each one of these parameters on the score, are included in Appendix A.1.

Table 26: Action hyper-parameters for Random Forest Classifier.

Hyper-parameter	**Value**
n_estimators	190
criterion	entropy
max_depth	60
min_samples_split	0.1
min_samples_leaf	0.1
min_weight_fraction_leaf	0.0714
max_features	auto
max_leaf_nodes	None
min_impurity_decrease	0.0
min_impurity_split	$1 \cdot 10^{-7}$
bootstrap	True
oob_score	False
n_jobs	None
random_state	None
verbose	0
warm_start	False
class_weight	None

The second classifier algorithm to be tuned is Gradient Boosting Classifier, which shares common hyper-parameters with Random Forest as both use decision trees. Then, the same process is followed to select the appropriate value to its hyper-parameters. Starting with determining the number of missions that should be applied to achieve the best scores. As it can be seen in Figure 36, the behavior of this algorithm is almost the same than for the

previous one, obtaining better scores as the amount of data increases, until achieving an almost constant value. In order to maintain a similar tuning process for both algorithms, 700 missions are also selected.

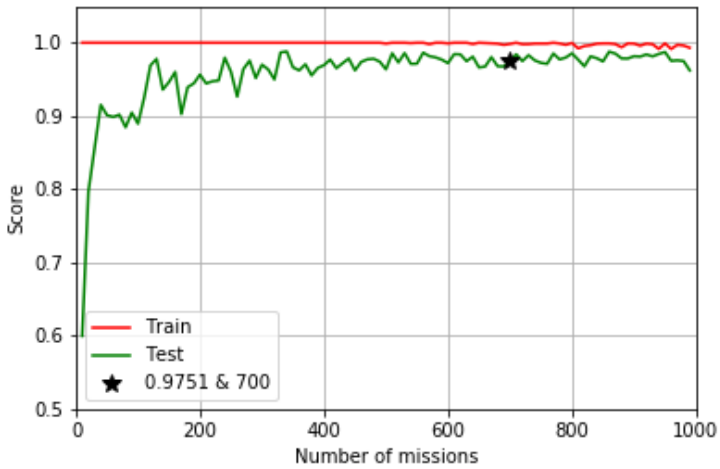


Figure 36: Gradient Boosting Classifier's behavior with the number of missions for Action.

In the case of Gradient Boosting, it also has several hyper-parameters. Then, after performing a previous research to set some of them their default value, the main hyper-parameter to be tuned are:

- *learning_rate*: It determines the impact of each tree in the result and controls the magnitude of the update of the output of each tree in the estimations. Lower values will increase the number of trees required to model all the relations, thus, extending the computational effort.
- *n_estimators*: It is the number of sequential trees to be modeled. The higher this number, the better performance will be, however, it can overfit at one point.
- *min_samples_split*: It defines the minimum required samples placed in a node to be taken into account for splitting.
- *min_samples_leaf*: It specifies the minimum necessary samples in a terminal node or leaf.
- *min_weight_fraction_leaf*: It is similar to *min_samples_leaf* besides it is defined as a fraction of the total number of samples.

- *max_depth*: It represents the maximum depth that a tree has and it is utilized to manage over-fitting problems.
- *max_leaf_nodes*: It indicates the maximum number of terminal leaves or nodes of a tree.

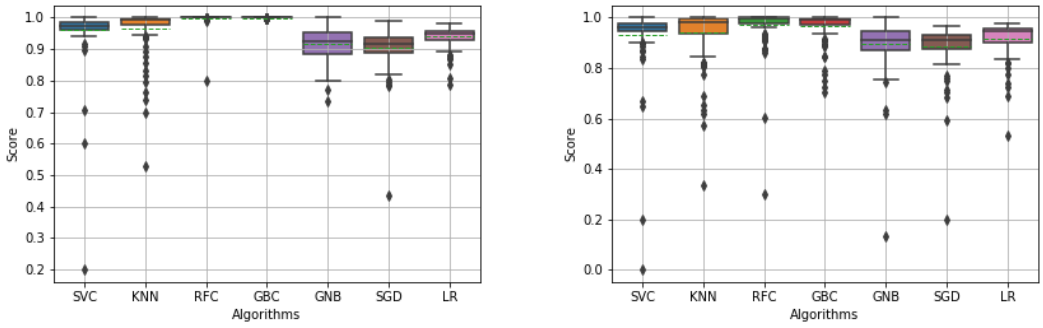
Once the grid search is finished, the best value for each parameter has been determined and they are listed in Table 27. These values where selected because they obtain the best score value for each one of them. All the figures, which show the impact of each one of these parameters on the score, are included in Appendix A.2.

Table 27: Action hyper-parameters for Gradient Boosting Classifier.

Hyper-parameter	**Value**
loss	deviance
learning_rate	0.9526
n_estimators	70
subsample	1.0
criterion	friedman_mse
min_samples_split	0.6143
min_samples_leaf	0.1571
min_weight_fraction_leaf	0.2347
max_depth	2
min_impurity_decrease	0.0
min_impurity_split	$1 \cdot 10^{-7}$
init	None
random_state	None
max_features	None
verbose	0
max_leaf_nodes	3
warm_start	False
presort	auto
validation_fraction	0.1
n_iter_no_change	None
tol	$1 \cdot 10^{-4}$

5.2.4 Procedures: Selection of the best algorithms and hyper-parameters tuning.

The same process is performed to procedures. As done before, the dataset is divided in training set and test set, and each algorithm is used in both of them to determine the ones that provide the best score are. Figure 37 shows the results using a box plot for both training set and test set.



(a) (b)

Figure 37: Procedures scores of the training set (a) and for the test set (b).

In Tables 28 and 29, the results for the training and test set are depicted numerically to make easier the interpretation of the results and the determination of the best algorithms.

Table 28: Procedure's training set results' main characteristics.

	SVC	**KNN**	**RFC**	**GBC**	**GNB**	**SGD**	**LR**
Number of samples	99	99	99	99	99	99	99
Mean	0.9594	0.9671	0.9976	0.9998	0.9157	0.9049	0.9393
Standard deviation	0.0917	0.0729	0.0201	0.0006	0.0518	0.0636	0.0321
Minimum value	0.2	0.53	0.8	0.9962	0.7338	0.4333	0.7864
25%	0.9624	0.9779	1	1	0.8846	0.8891	0.9297
50%	0.9739	0.9953	1	1	0.9229	0.9176	0.9497
75%	0.9864	0.9991	1	1	0.9523	0.9380	0.9575
Maximum value	1	1	1	1	1	0.9895	0.9798

Table 29: Procedure's test set results' main characteristics.

	SVC	KNN	RFC	GBC	GNB	SGD	LR
Number of samples	99	99	99	99	99	99	99
Mean	0.9335	0.9393	0.9702	0.9677	0.8938	0.8882	0.9175
Standard deviation	0.1319	0.1085	0.0849	0.0603	0.1035	0.0943	0.0685
Minimum value	0	0.3333	0.3	0.7033	0.1333	0.2	0.5333
25%	0.9456	0.9339	0.9787	0.9703	0.8702	0.8780	0.9029
50%	0.9606	0.9837	0.9957	0.9915	0.9132	0.9114	0.9457
75%	0.9754	0.9960	1	0.9973	0.9481	0.9317	0.9557
Maximum value	1	1	1	1	1	0.9689	0.9776

Considering the results, the algorithms which provide the best score and have the best characteristics are Random Forest Classifier and Gradient Boosting Classifier. Then the tuning process is performed applying the same process as for Actions.

Starting with Random Forest Classifier, the first step is to determine the number of missions that are going to be selected to perform the tuning. Figure 38 shows how the score is better as the number of missions increase, as expected. The star symbol represents the number of missions selected to carry out the process, in this case 600 missions.

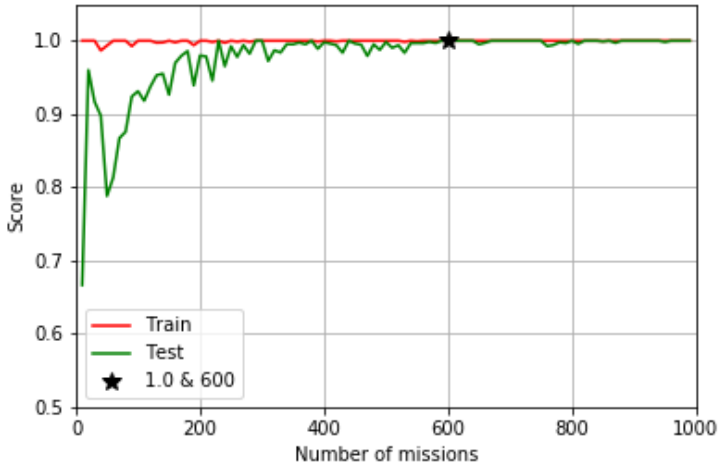


Figure 38: Random Forest Classifier's behavior with the number of missions for Procedure.

The hyper-parameters to be tuned are the same than the ones used in the case of Actions. Then, after performing the grid search, for Random Forest Classifier, the value of each hyper-parameter is collected in Table 30. The graphs that illustrates the behavior of the parameters are included in Appendix B.1.

Table 30: Procedure hyper-parameters for Random Forest Classifier.

Hyper-parameter	**Value**
n_estimators	130
criterion	entropy
max_depth	120
min_samples_split	0.1551
min_samples_leaf	0.1571
min_weight_fraction_leaf	0.1429
max_features	auto
max_leaf_nodes	None
min_impurity_decrease	0.0204
min_impurity_split	$1 \cdot 10^{-7}$
bootstrap	True
oob_score	False
n_jobs	None
random_state	None
verbose	0
warm_start	False
class_weight	None

Once the hyper-parameters of the Random Forest are tuned, the next step is to perform the same process to Gradient Boosting Classifier. As done before, the first step is to determine the number of missions which will give an appropriate score. Figure 39 shows how the score is higher when the value number of missions increases, since, the amount of data is higher, 500 missions are selected to perform the tuning.

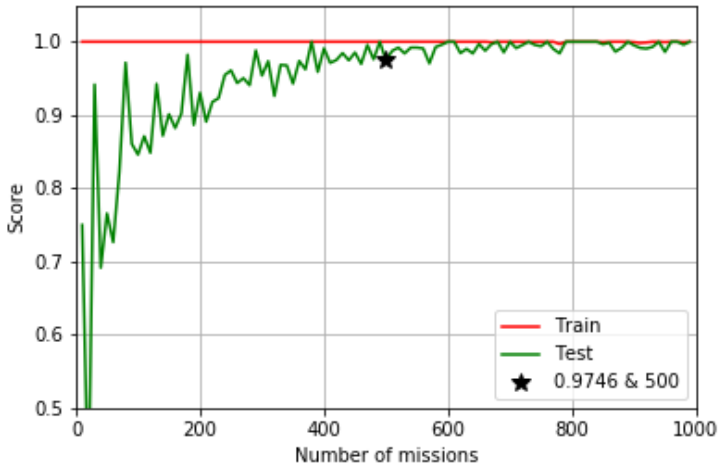


Figure 39: Gradient Boosting Classifier's behavior with the number of missions for Procedure.

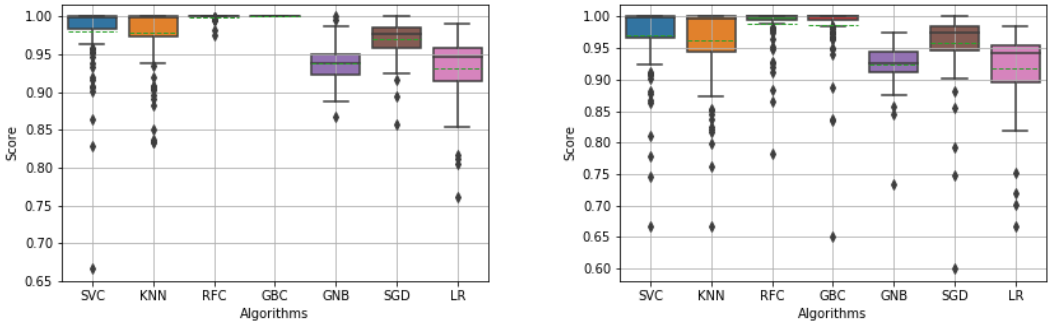
Table 31 contains the values of each hyper-parameter of Gradient Boosting Classifier. The tuning process is performed to the same parameters used in Actions and all the graphs which show their behavior are included in Appendix B.2.

Table 31: Procedure hyper-parameters for Gradient Boosting Classifier.

Hyper-parameter	**Value**
loss	deviance
learning_rate	0.9526
n_estimators	90
subsample	1.0
criterion	friedman_mse
min_samples_split	0.5776
min_samples_leaf	0.1
min_weight_fraction_leaf	0.0
max_depth	3
min_impurity_decrease	0.0
min_impurity_split	$1 \cdot 10^{-7}$
init	None
random_state	None
max_features	None
verbose	0
max_leaf_nodes	4
warm_start	False
presort	auto
validation_fraction	0.1
n_iter_no_change	None
tol	$1 \cdot 10^{-4}$

5.2.5 Tasks: Selection of the best algorithms and hyper-parameters tuning.

Following with tasks, the same method is emulated. So, the dataset is split in training and test set, and each algorithm is used in both of them to determine which are the ones that provide the best results and characteristics. Figure 40 shows the results using box plots for training set and test set.



(a) (b)

Figure 40: Task scores of the training set (a) and for the test set (b).

All the values that characterize the behavior of each algorithm for the training set and test set are listed in Tables 32 and 33, in order to determine which are the best ones.

Table 32: Task's training set results' main characteristics.

	SVC	**KNN**	**RFC**	**GBC**	**GNB**	**SGD**	**LR**
Number of samples	99	99	99	99	99	99	99
Mean	0.9813	0.9781	0.9993	1	0.9381	0.9701	0.9319
Standard deviation	0.0444	0.0408	0.0032	0	0.0238	0.0237	0.0416
Minimum value	0.6667	0.8333	0.975	1	0.8670	0.8575	0.7608
25%	0.9832	0.9743	1	1	0.9226	0.9588	0.9143
50%	0.9997	0.9991	1	1	0.9383	0.9767	0.9469
75%	1	1	1	1	0.9507	0.9856	0.9593
Maximum value	1	1	1	1	1	1	0.9899

Table 33: Task's test set results' main characteristics.

	SVC	KNN	RFC	GBC	GNB	SGD	LR
Number of samples	99	99	99	99	99	99	99
Mean	0.9704	0.9620	0.9886	0.9867	0.9245	0.9583	0.9184
Standard deviation	0.0578	0.0637	0.0319	0.0439	0.0317	0.0530	0.0570
Minimum value	0.6667	0.6667	0.7833	0.65	0.7333	0.6	0.6667
25%	0.9670	0.9434	0.9947	0.9943	0.9121	0.9466	0.8958
50%	0.9982	0.9681	1	1	0.9269	0.9738	0.9426
75%	1	1	1	1	0.9448	0.9848	0.9537
Maximum value	1	1	1	1	0.9737	1	0.9852

As done previously, the first algorithm to be tuned is Random Forest Classifier and the first step is to determine the number of missions, i.e. amount of data, to be used to adjust the parameters. Figure 41 shows how the score grows with the number of missions as expected, since the amount of data increases and the predictions are better.

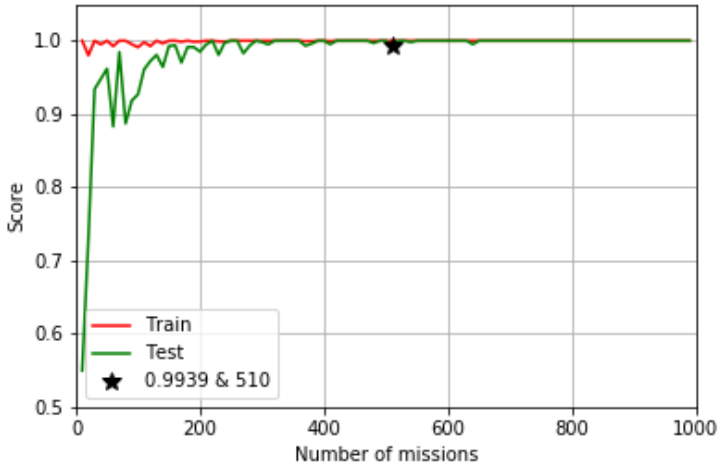


Figure 41: Random Forest Classifier's behavior the number of missions for Task.

The hyper-parameters are the same than the ones used in the case of Actions and Procedures. The value of each hyper-parameter is listed in Table 34. The graphs that represents the behavior of these parameters are included in Appendix C.1

Table 34: Task hyper-parameters for Random Forest Classifier.

Hyper-parameter	**Value**
n_estimators	130
criterion	entropy
max_depth	70
min_samples_split	0.1
min_samples_leaf	0.1
min_weight_fraction_leaf	0.0
max_features	auto
max_leaf_nodes	None
min_impurity_decrease	0.0
min_impurity_split	$1 \cdot 10^{-7}$
bootstrap	True
oob_score	False
n_jobs	None
random_state	None
verbose	0
warm_start	False
class_weight	None

Continuing with Gradient Boosting Classifier, the number of missions is computed to establish the amount of data to be used for the tuning. Figure 42 shows how the score increases with the number of missions.

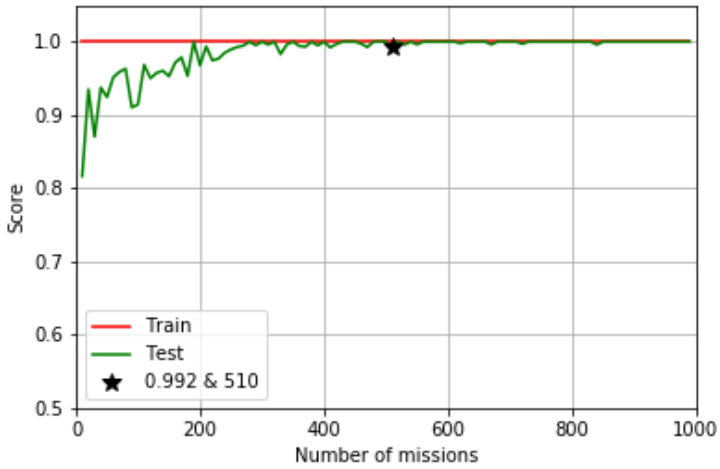


Figure 42: Gradient Boosting Classifier's behavior with the number of missions for Task.

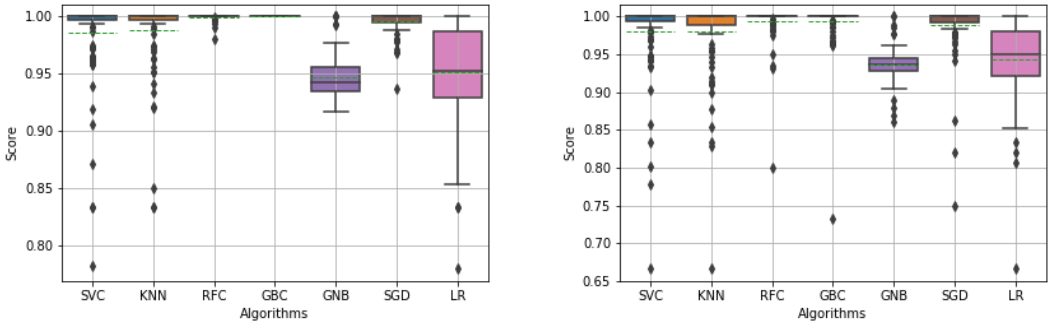
The value of each hyper-parameter, which are also the one tuned for Actions and Procedures, is listed in Table 35. The behavior of the parameters is represented in the graphs included in Appendix C.2.

Table 35: Task hyper-parameters for Gradient Boosting Classifier.

Hyper-parameter	**Value**
loss	deviance
learning_rate	0.5
n_estimators	110
subsample	1.0
criterion	friedman_mse
min_samples_split	0.3
min_samples_leaf	0.1
min_weight_fraction_leaf	0.0
max_depth	3
min_impurity_decrease	0.0
min_impurity_split	$1 \cdot 10^{-7}$
init	None
random_state	None
max_features	None
verbose	0
max_leaf_nodes	4
warm_start	False
presort	auto
validation_fraction	0.1
n_iter_no_change	None
tol	$1 \cdot 10^{-4}$

5.2.6 Mission: Selection of the best algorithms and hyper-parameters tuning.

Finally, after tuning all the parameters for all the previous components, the same operation is performed for the whole mission. The first step is to identify the algorithms that provide the best score. Figure 43 shows how each algorithm behaves for the given training set and test set.



(a) (b)

Figure 43: Mission scores of the training set (a) and for the test set (b).

The numerical values for all the relevant parameters which describe how good are the predictions of the algorithms are included in Tables 36 and 37.

Table 36: Mission's training set results' main characteristics.

	SVC	**KNN**	**RFC**	**GBC**	**GNB**	**SGD**	**LR**
Number of samples	99	99	99	99	99	99	99
Mean	0.9861	0.9882	0.9995	1	0.9470	0.9955	0.9519
Standard deviation	0.0370	0.0316	0.0023	0	0.0188	0.0092	0.0424
Minimum value	0.7823	0.8333	0.98	1	0.9168	0.9367	0.7797
25%	0.9973	0.9969	1	1	0.9348	0.9944	0.9289
50%	1	1	1	1	0.9429	1	0.9524
75%	1	1	1	1	0.9564	1	0.9871
Maximum value	1	1	1	1	1	1	1

Table 37: Mission's test set results' main characteristics.

	SVC	KNN	RFC	GBC	GNB	SGD	LR
Number of samples	99	99	99	99	99	99	99
Mean	0.9810	0.9809	0.9946	0.9938	0.9365	0.9882	0.9441
Standard deviation	0.0510	0.0483	0.0227	0.0280	0.0222	0.0346	0.0507
Minimum value	0.6667	0.6667	0.8	0.7333	0.86	0.75	0.6667
25%	0.9938	0.9892	1	1	0.9274	0.9927	0.9223
50%	1	1	1	1	0.9365	1	0.9506
75%	1	1	1	1	0.9456	1	0.9805
Maximum value	1	1	1	1	1	1	1

Looking at the results, Random Forest and Gradient Boosting classifiers are the algorithms that provide best results for the datasets applied. Then, as performed for all the components, they are tuned to determine the best value for their hyper-parameters.

Starting with Random Forest Classifier, the number of missions is determined. The behavior, illustrated in Figure 44, is the same than for the previous components, although the best score is achieved for less amount of data. Taking this into account, 210 missions are selected.

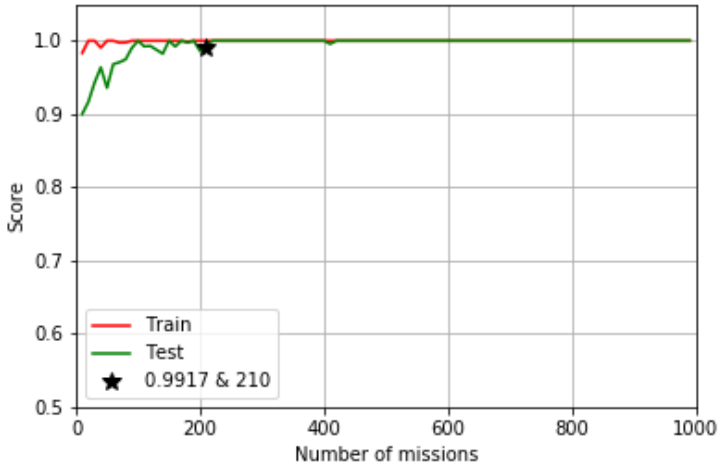


Figure 44: Random Forest Classifier's behavior with the number of missions for Mission.

As explained in Section 5.2.3, the hyper-parameters that have an appreciable effect on the performance of the algorithm are tuned. Since Mission Assessment is the main objective of this project, the process followed to determine each hyper-parameter's value is explained instead of being included in an Appendix.

- *n_estimators*: The score does not change with any value of this parameter, as it is seen in Figure 45.

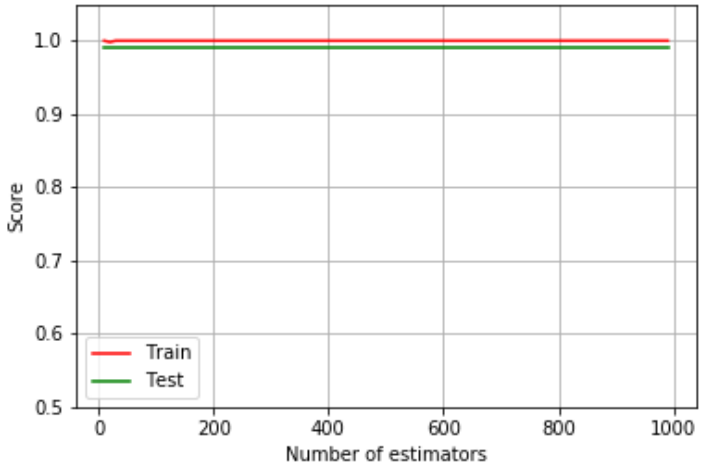


Figure 45: Behavior of Random Forest with *n_estimators*.

- *max_depth*: It has the same behavior than the previous parameter as it is shown in Figure 46.

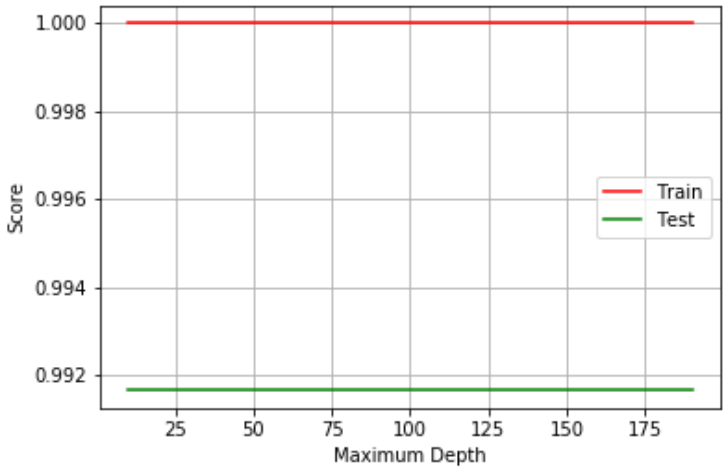


Figure 46: Behavior of Random Forest with *max_depth*.

- *min_samples_split*: Figure 47 shows its effect on the score, it is observed that the higher score is reached for its minimum value.

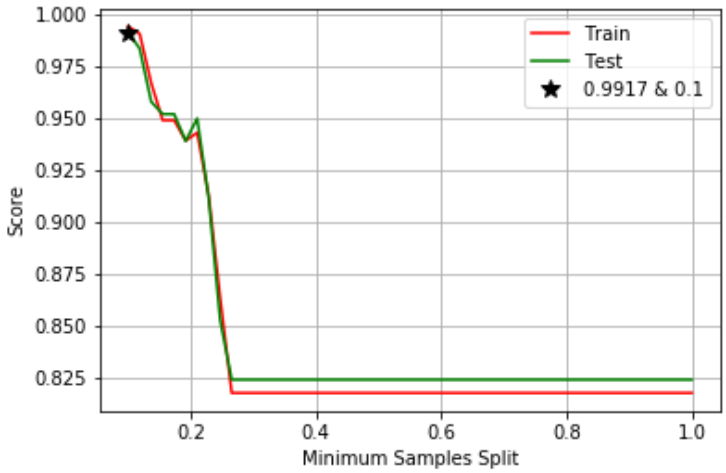


Figure 47: Behavior of Random Forest with *min_samples_split*.

- *min_samples_leaf*: The score is kept constant despite the different values of this

parameter. Its behavior is represented in Figure 48.

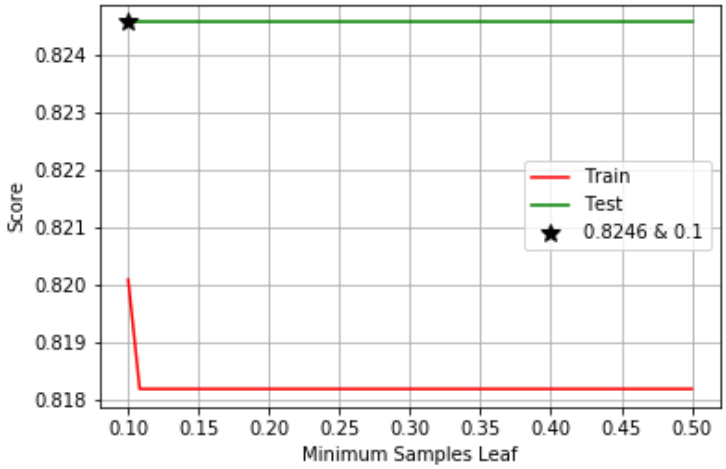


Figure 48: Behavior of Random Forest with *min_samples_leaf*.

- *min_weight_fraction_leaf*: Figure 49 illustrates how the score value oscillates for the lower values and, after, it reaches a constant behavior.

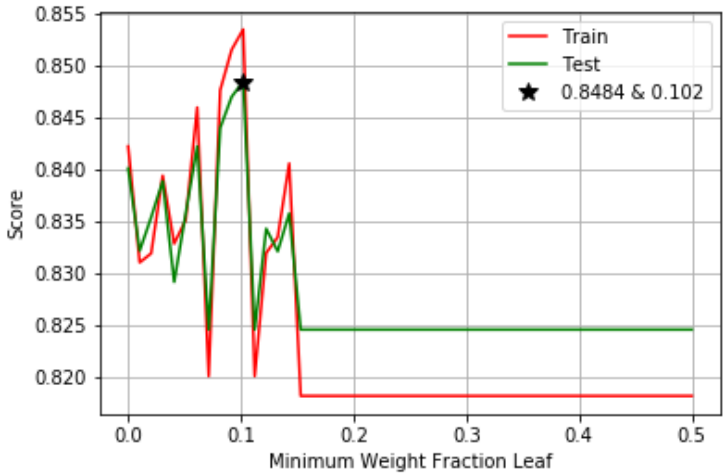


Figure 49: Behavior of Random Forest with *min_weight_fraction_leaf*.

- *min_impurity_decrease*: It has the same effect on the score than the previous parameter, as it is observed in Figure 50.

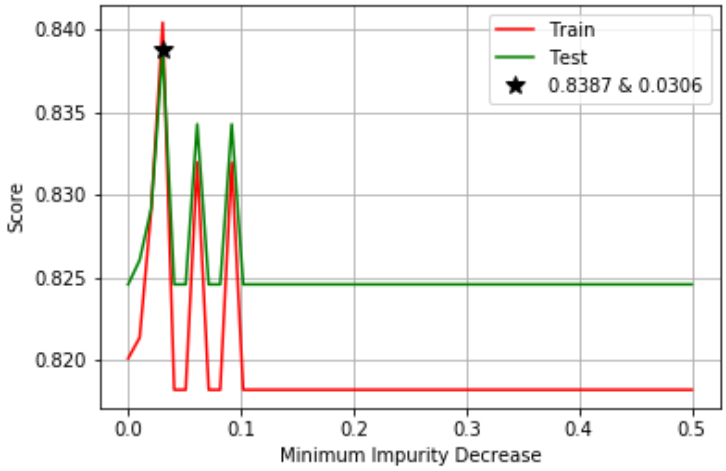


Figure 50: Behavior of Random Forest with *min_impurity_decrease*.

Once the tuning process is performed, the best value of each hyper-parameter has been determined and included in Table 38.

Table 38: Mission Hyper-parameters for Random Forest Classifier.

Hyper-parameter	**Value**
n_estimators	130
criterion	entropy
max_depth	70
min_samples_split	0.1
min_samples_leaf	0.1
min_weight_fraction_leaf	0.102
max_features	auto
max_leaf_nodes	None
min_impurity_decrease	0.0306
min_impurity_split	$1 \cdot 10^{-7}$
bootstrap	True
oob_score	False
n_jobs	None
random_state	None
verbose	0
warm_start	False
class_weight	None

Following the same method for Gradient Boosting Classifier, the number of missions is determined, obtaining in this case 220 missions to perform the tuning process.

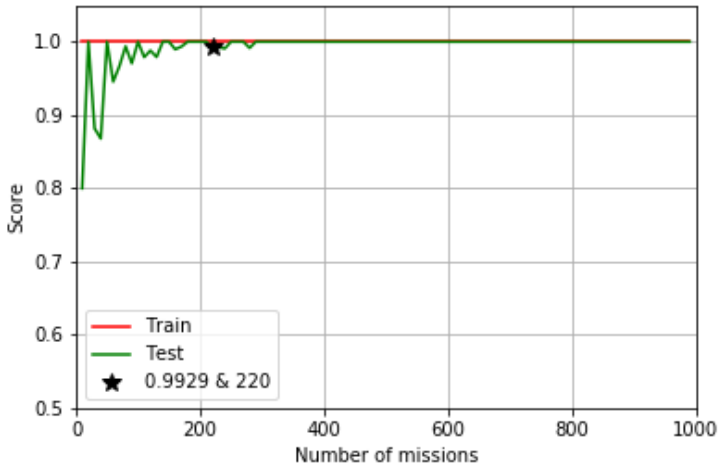


Figure 51: Gradient Boosting Classifier's behavior with the number of missions for Mission.

Once the number of missions has been selected, the next step is to perform the grid search to determine the best value of each of the hyper-parameters, which are the same as for Actions, Procedures and Tasks:

- *learning_rate*: Figure 52 shows how this parameter has an almost insignificant effect on the score, achieving the highest one for low values of this parameter.

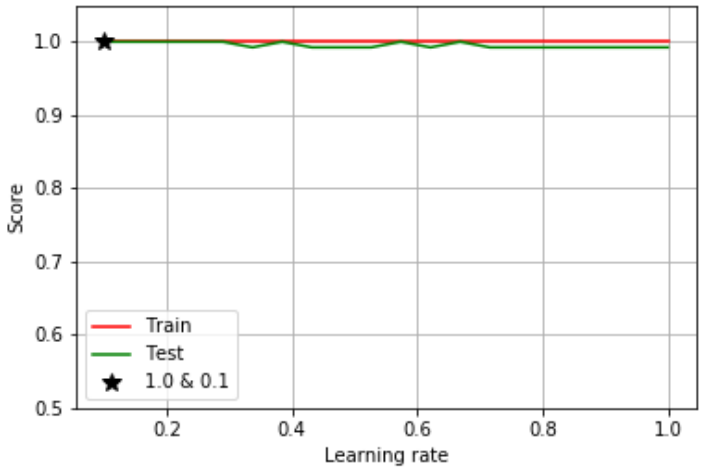


Figure 52: Behavior of Gradient Boosting with *learning_rate*.

- *n_estimators*: It is observed in Figure 53 how the score obtained increases with this parameter until reaching a constant score.

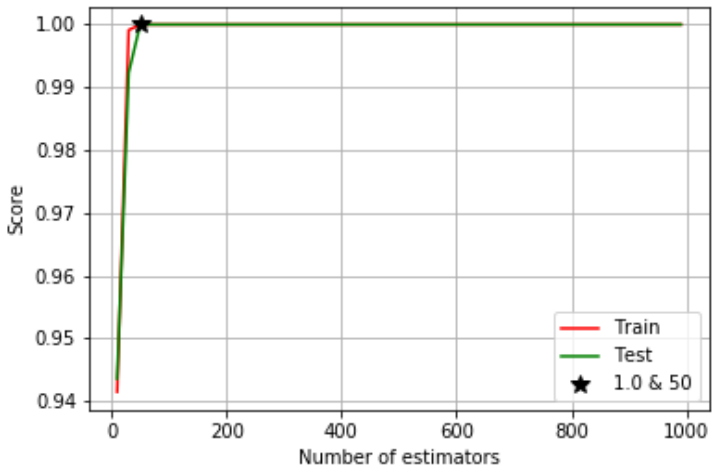


Figure 53: Behavior of Gradient Boosting with *n_estimators*.

- *min_samples_split*: The higher scores are reached for lower values of this parameter and, when this value increases, the score decreases abruptly as it is illustrated in Figure 54.

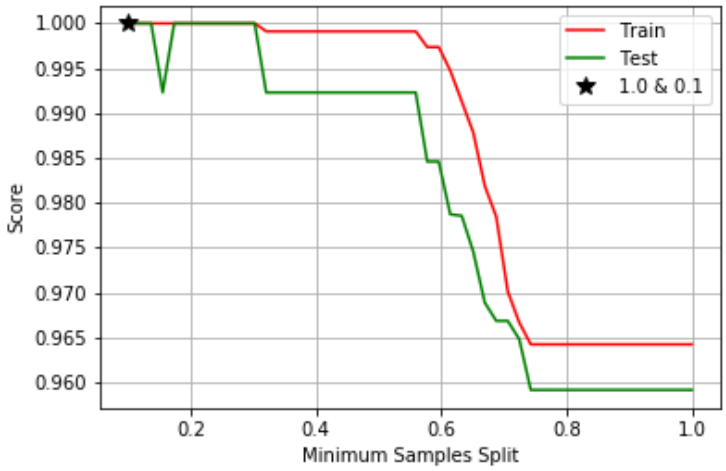


Figure 54: Behavior of Gradient Boosting with *min_samples_split*.

- *min_samples_leaf*: This parameter, represented in Figure 55, has a similar effect, obtaining the best score for its lower values. Also, the score decreases when the value of this parameter rises.

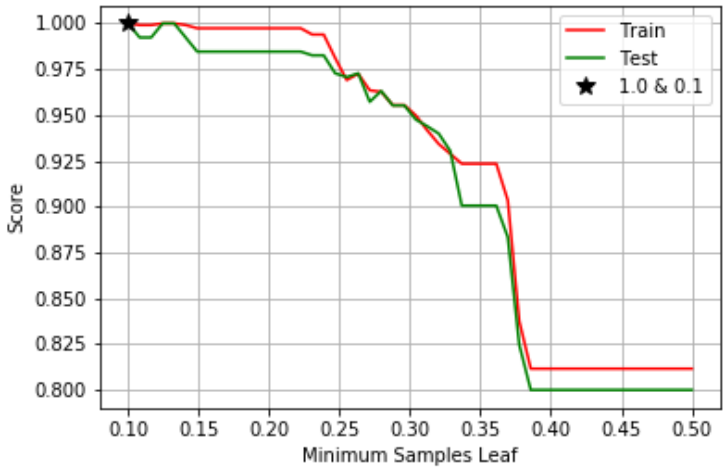


Figure 55: Behavior of Gradient Boosting with *min_samples_leaf*.

- *min_weight_fraction_leaf*: Figure 56 shows its influence in the score. It is almost identical to the two previous parameters, the best score is achieved for its lowest value.

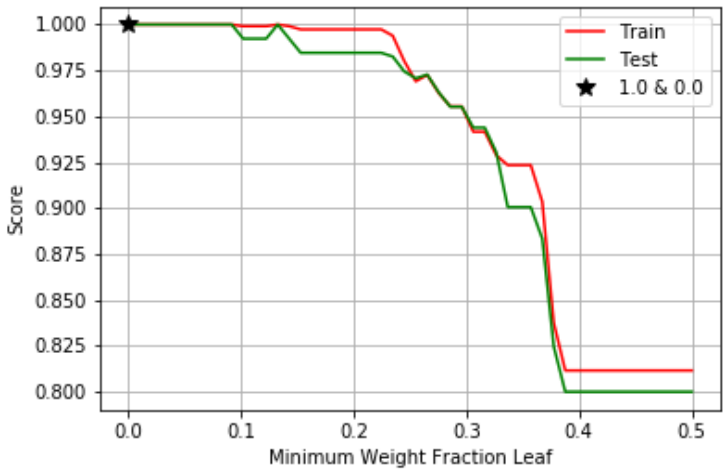


Figure 56: Behavior of Gradient Boosting with *min_weight_fraction_leaf*.

- *max_depth*: It is seen in Figure 57 how the score increases with this parameter until get a constant value.

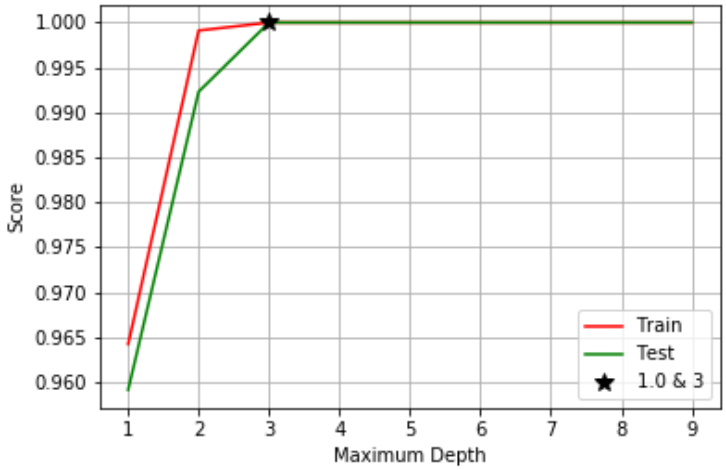


Figure 57: Behavior of Gradient Boosting with *max_depth*.

- *max_leaf_nodes*: Its behavior, depicted in Figure 58, is very similar to the previous parameter. The score is higher as the value increases and, finally, the score is constant all the time.

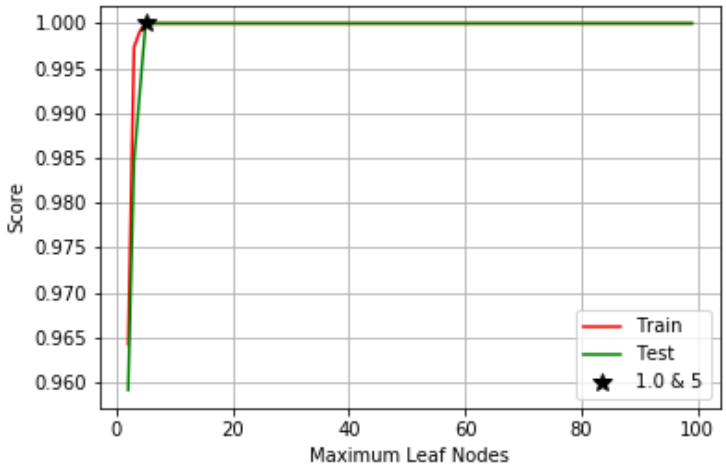


Figure 58: Behavior of Gradient Boosting with *max_leaf_nodes*.

After performing the grid search for Gradient Boosting Classifier, the value of each hyper-parameter is collected in Table 39.

Table 39: Mission hyper-parameters for Gradient Boosting Classifier.

Hyper-parameter	**Value**
loss	deviance
learning_rate	0.1
n_estimators	90
subsample	1.0
criterion	friedman_mse
min_samples_split	0.1
min_samples_leaf	0.1
min_weight_fraction_leaf	0.0
max_depth	3
min_impurity_decrease	0.0
min_impurity_split	$1 \cdot 10^{-7}$
init	None
random_state	None
max_features	None
verbose	0
max_leaf_nodes	5
warm_start	False
presort	auto
validation_fraction	0.1
n_iter_no_change	None
tol	$1 \cdot 10^{-4}$

5.3 Application of the prototype to Global Mission Assessment

Once all the parameters are tuned for both algorithms, the next step is to obtain the weights and to perform predictions to demonstrate that the results obtained are accurate and trustworthy.

The algorithms are implemented using the value of the hyper-parameters obtained for each of the components.

5.3.1 Action Results

The first expected result is the value of the weights which represent the influence that each Flight Facet has in Action Assessment. Regarding Equation 2, the value wanted to

be determined is α_{jk}.

The weights are computed after performing the training process, in order to have a good amount of data to obtain accurate and logic results. Figure **??** shows the representation of the value of the weights for the first action.

Table 40: Flight Facet weights for all the Actions.

	$\boldsymbol{\alpha_{j1}}$	$\boldsymbol{\alpha_{j2}}$	$\boldsymbol{\alpha_{j3}}$	$\boldsymbol{\alpha_{j4}}$	$\boldsymbol{\alpha_{j5}}$	$\boldsymbol{\alpha_{j6}}$	$\boldsymbol{\alpha_{j7}}$	$\boldsymbol{\alpha_{j8}}$
$\boldsymbol{A_1}$	0.0577	0.1568	0.0835	0.2956	0.0637	0.0616	0.2085	0.0726
$\boldsymbol{A_2}$	0.0419	0.2984	0.0591	0.1068	0.2413	0.1080	0.0904	0.0541
$\boldsymbol{A_3}$	0.0650	0.4474	0.2057	0.0464	0.0074	0.0459	0.0901	0.0922
$\boldsymbol{A_4}$	0.0621	0.0543	0.0827	0.1957	0.3356	0.1563	0.0495	0.0638
$\boldsymbol{A_5}$	0.0489	0.0529	0.0177	0.0756	0.3697	0.3488	0.0754	0.0107
$\boldsymbol{A_6}$	0.0537	0.0357	0.1236	0.3409	0.0997	0.1976	0.1314	0.0175
$\boldsymbol{A_7}$	0.0055	0.0643	0.0817	0.2082	0.0095	0.4800	0.1277	0.0230
$\boldsymbol{A_8}$	0.0653	0.1277	0.1257	0.0884	0.3867	0.1379	0.0203	0.0479
$\boldsymbol{A_9}$	0.0158	0.0888	0.0151	0.0139	0.2950	0.3853	0.1519	0.0340
$\boldsymbol{A_{10}}$	0.0639	0.3033	0.2918	0.0473	0.0902	0.0484	0.1045	0.0504
$\boldsymbol{A_{11}}$	0.0116	0.0324	0.0733	0.2449	0.0554	0.3943	0.1578	0.0303
$\boldsymbol{A_{12}}$	0.3258	0.0419	0.0207	0.0762	0.0109	0.0825	0.2387	0.2033
$\boldsymbol{A_{13}}$	0.0388	0.0908	0.1133	0.3255	0.0644	0.0772	0.2393	0.0505
$\boldsymbol{A_{14}}$	0.1999	0.0333	0.4303	0.0095	0.0608	0.1037	0.0469	0.1157
$\boldsymbol{A_{15}}$	0.0841	0.1105	0.0879	0.2965	0.0944	0.0756	0.1931	0.0578
$\boldsymbol{A_{16}}$	0.3026	0.0131	0.0915	0.0340	0.0642	0.0577	0.2127	0.2241
$\boldsymbol{A_{17}}$	0.2649	0.0291	0.0091	0.0724	0.0218	0.1257	0.2617	0.2152
$\boldsymbol{A_{18}}$	0.0046	0.0388	0.1058	0.3712	0.0799	0.1435	0.1818	0.0744

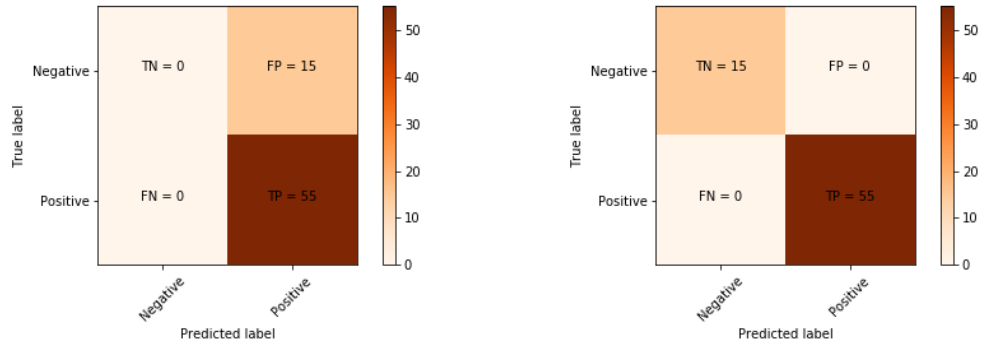
The values of the weights represent the influence and importance that the specific Flight Facet has on Action Assessment. Taking as an example the first Action (Reach 240 KDAS in the runway), the highest weights are α_{j4}, α_{j7} and α_{j2}, whose values are 0.2956, 0.2085 and 0.1568 respectively. Then, it indicates that any issue related to any of that Flight Facets is susceptible to have a crucial effect on the Action Assessment. In particular, Propulsion Flight Facet has the highest influence in the assessment.

After determining the weights, the algorithm is able to perform predictions based on the knowledge acquired. The way of evaluating how good a prediction is was explained in Section 5.2.1, so, the different scores obtained from both algorithms are computed and collected in Table 41.

Table 41: Prediction Scores for all the actions.

	Random Forest Score				**Gradient Boosting Score**			
	Accuracy	Precision	Recall	F1	Accuracy	Precision	Recall	F1
$\boldsymbol{A_1}$	0.7857	0.7857	1.0	0.88	1.0	1.0	1.0	1.0
$\boldsymbol{A_2}$	0.7429	0.7429	1.0	0.8525	1.0	1.0	1.0	1.0
$\boldsymbol{A_3}$	0.8143	0.7869	1.0	0.8807	0.9857	1.0	0.9792	0.9895
$\boldsymbol{A_4}$	0.7286	0.7286	1.0	0.8429	0.9714	0.9804	0.9804	0.9804
$\boldsymbol{A_5}$	0.8	0.8	1.0	0.8889	0.9857	0.9825	1.0	0.9912
$\boldsymbol{A_6}$	0.7	0.65	1.0	0.7879	1.0	1.0	1.0	1.0
$\boldsymbol{A_7}$	0.8286	0.7857	1.0	0.88	1.0	1.0	1.0	1.0
$\boldsymbol{A_8}$	0.7286	0.7286	1.0	0.8429	0.9857	1.0	0.9804	0.9901
$\boldsymbol{A_9}$	0.9714	0.9535	1.0	0.9762	1.0	1.0	1.0	1.0
$\boldsymbol{A_{10}}$	0.8143	0.7759	1.0	0.8738	0.9714	1.0	0.9556	0.9773
$\boldsymbol{A_{11}}$	0.7	0.6818	1.0	0.8108	0.9714	0.9574	1.0	0.9783
$\boldsymbol{A_{12}}$	0.8571	0.8039	1.0	0.8913	0.9857	1.0	0.9756	0.9877
$\boldsymbol{A_{13}}$	0.7429	0.7391	1.0	0.85	1.0	1.0	1.0	1.0
$\boldsymbol{A_{14}}$	0.8571	0.7561	1.0	0.8611	1.0	1.0	1.0	1.0
$\boldsymbol{A_{15}}$	0.7286	0.7286	1.0	0.8429	1.0	1.0	1.0	1.0
$\boldsymbol{A_{16}}$	0.9	0.8333	1.0	0.9381	1.0	1.0	1.0	1.0
$\boldsymbol{A_{17}}$	0.9571	0.9583	0.9787	0.9684	1.0	1.0	1.0	1.0
$\boldsymbol{A_{18}}$	0.8	0.7667	1.0	0.8679	1.0	1.0	1.0	1.0

Additionally, Figure 59 shows an example of the way the confusion matrices are displayed for the first Action.



(a) (b)

Figure 59: First Action confusion matrices for Random Forest (a) and Gradient boosting (b) classifiers.

Looking at the results, it is appreciated that Gradient Boosting algorithm gives best results in terms of scoring and, also, the number of false positives obtained is lower than for Random Forest.

5.3.2 Procedure Results

The first result is the value of the weights which represent the influence that each Action has in Procedure Assessment. Regarding Equation 3, the sought value is γ_{ij}.

The weights are computed after performing the training process. Figure 60 shows the representation of the value of the weights for the Procedures. In addition, Tables 42, 43 and 44 contains all weights' specific value for each action of the three Procedures respectively.

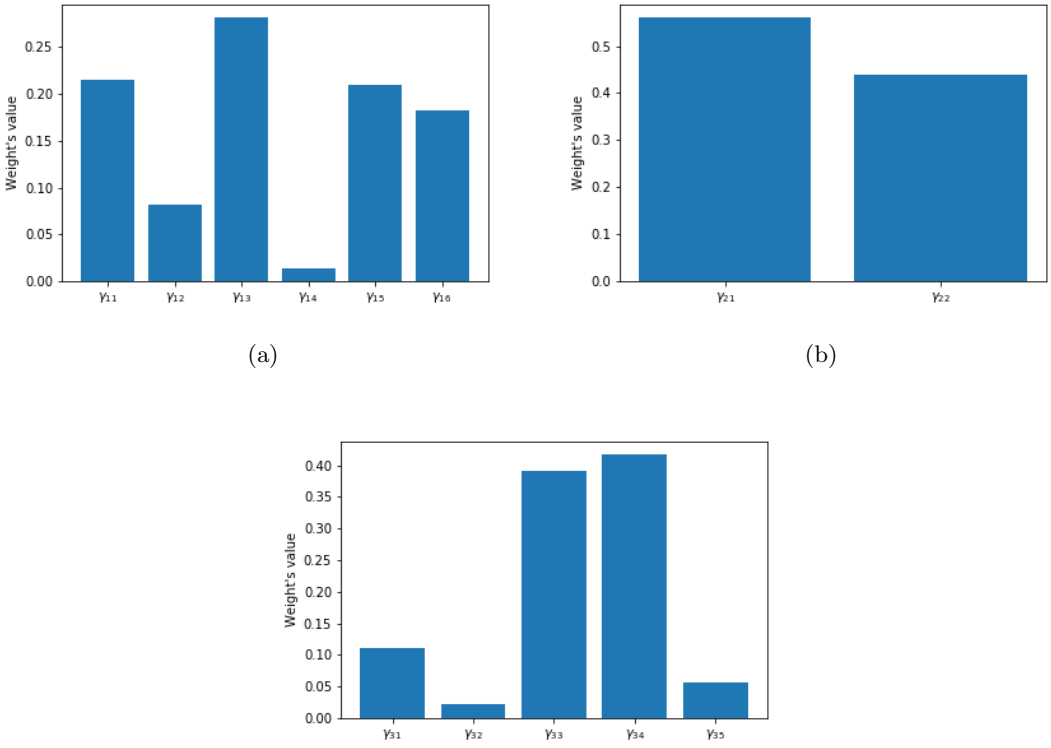


Figure 60: Value of Actions weights, γ_{ij}, for first (a), second (b) and third (c) Procedures.

Table 42: Actions weights for first Procedure.

	$\boldsymbol{\gamma_{11}}$	$\boldsymbol{\gamma_{12}}$	$\boldsymbol{\gamma_{13}}$	$\boldsymbol{\gamma_{14}}$	$\boldsymbol{\gamma_{15}}$	$\boldsymbol{\gamma_{16}}$
$\boldsymbol{P_1}$	0.2155	0.0823	0.2819	0.0136	0.2095	0.1820

Table 43: Actions weights for second Procedure.

	$\boldsymbol{\gamma_{212}}$	$\boldsymbol{\gamma_{213}}$
$\boldsymbol{P_2}$	0.2155	0.0823

Table 44: Actions weights for third Procedure.

	γ_{314}	γ_{315}	γ_{316}	γ_{317}	γ_{318}
P_3	0.1107	0.0229	0.3915	0.4176	0.0573

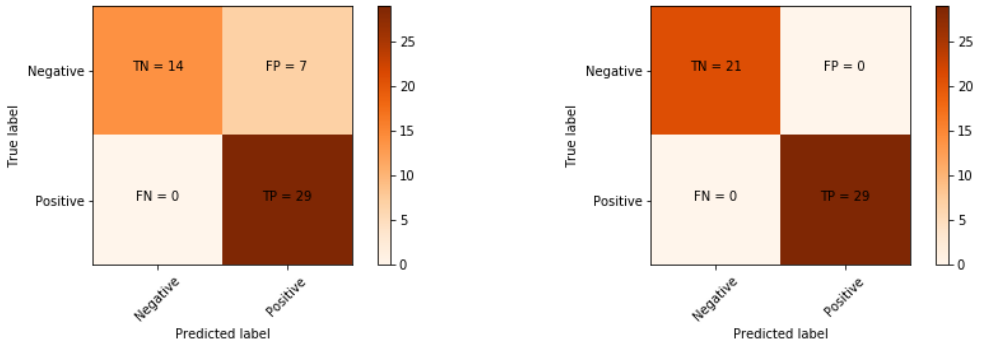
It is observed how for the first procedure almost all the actions have a significant influence. The second one, as it has only two actions, has a similar influence from both actions. The third procedure has two main actions that have a bigger effect on the assessment.

Once the values of the weights are known, it is possible to perform predictions since the algorithm has learned all the rules. So, the different scores for each algorithm are computed and listed in Table 45.

Table 45: Prediction scores for all the Procedures.

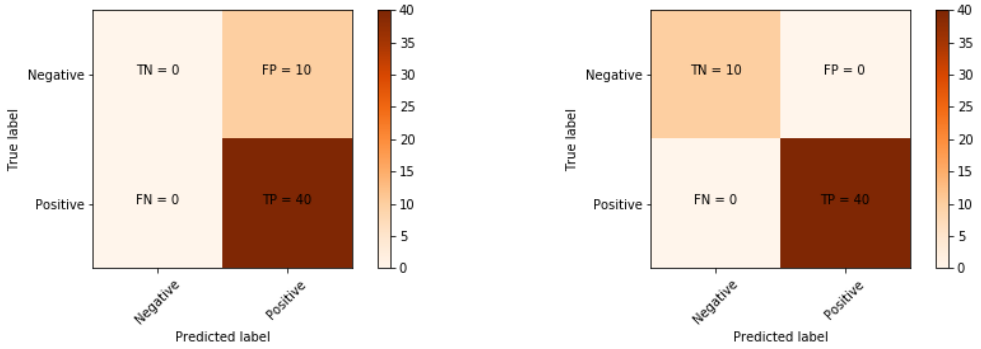
	Random Forest Score				**Gradient Boosting Score**			
	Accuracy	Precision	Recall	F1	Accuracy	Precision	Recall	F1
P_1	0.86	0.8056	1.0	0.8923	1.0	1.0	1.0	1.0
P_2	0.8	0.8	1.0	0.8889	1.0	1.0	1.0	1.0
P_3	0.78	0.78	1.0	0.8764	1.0	1.0	1.0	1.0

Additionally, Figure 61, 62 and 63 shows the confusion matrices for first, second and third Procedure respectively.



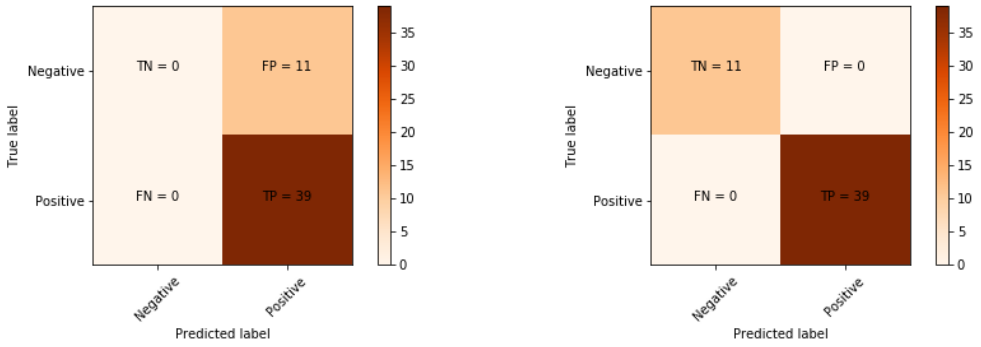
(a) (b)

Figure 61: First Procedure confusion matrices for Random Forest (a) and Gradient boosting (b) classifiers.



(a) (b)

Figure 62: Second Procedure confusion matrices for Random Forest (a) and Gradient boosting (b) classifiers.



(a) (b)

Figure 63: Third Procedure confusion matrices for Random Forest (a) and Gradient boosting (b) classifiers.

Considering the results obtained, Gradient Boosting is the algorithm that performs the best predictions for this type of data.

5.3.3 Task Results

As performed for the previous components, the value of each action weight is computed. For the case of Tasks, Equation 4, β_{ij} is calculated.

These weights are obtained from the training process. Figure 64 illustrates the five weight values for each one of the Actions. Additionally, their numerical value is listed in Table 46.

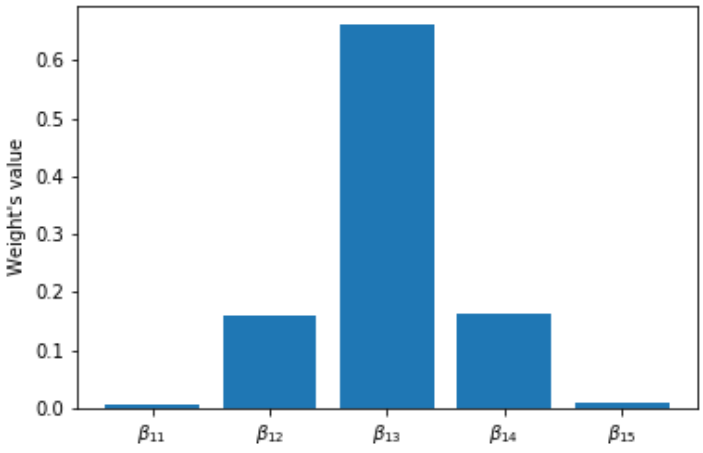


Figure 64: Value of the weights β_{ij} of each Action for the Task.

Table 46: Actions weights for the Task.

	$\boldsymbol{\beta_{17}}$	$\boldsymbol{\beta_{18}}$	$\boldsymbol{\beta_{19}}$	$\boldsymbol{\beta_{110}}$	$\boldsymbol{\beta_{111}}$
$\boldsymbol{T_1}$	0.0058	0.1589	0.6619	0.1639	0.0095

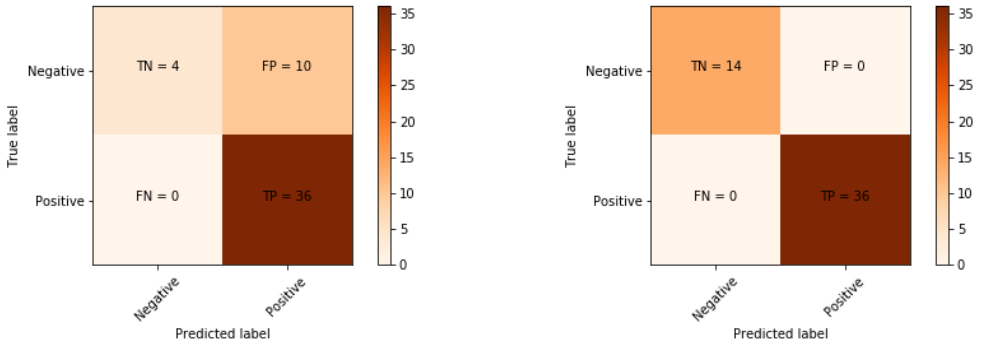
It is observed that Action number 9 is the most important one, so, any failure during the execution of that Action may lead to an unsuccessful Task.

Knowing the value of the weights it is possible to perform the prediction. The scores, for both algorithms, are included in Table 47, where it is seen that Gradient Boosting is the one which provides the best possible score.

Table 47: Task prediction score.

	Random Forest Score				**Gradient Boosting Score**			
	Accuracy	Precision	Recall	F1	Accuracy	Precision	Recall	F1
$\boldsymbol{T_1}$	0.8	0.7826	1.0	0.878	1.0	1.0	1.0	1.0

Figure 65 shows confusion matrices for both algorithms. They also prove that Gradient Boosting algorithm obtains better results with the type of data provided.



(a) (b)

Figure 65: Task confusion matrices for Random Forest (a) and Gradient boosting (b) classifiers.

5.3.4 Global Mission Results

Finally, Task and Procedures weights are calculated for the whole Mission, in this case the required values are δ_i and ϵ_i as it was shown in Equation 5. The results are illustrated in Figure 66 and their numerical values are listed in Table 48.

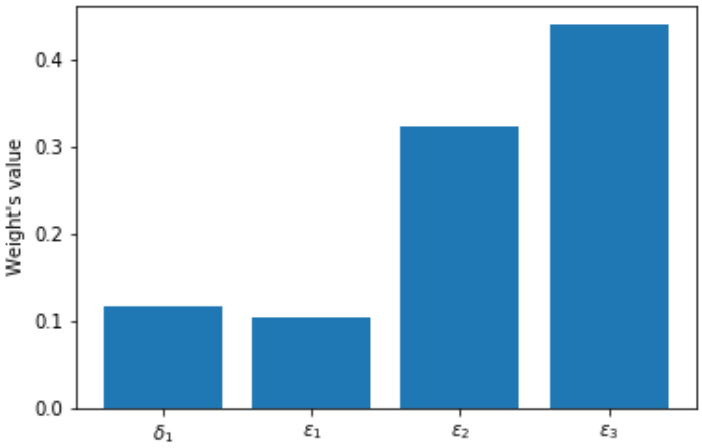


Figure 66: Value of the weights δ_i and ϵ_i for the Mission.

Table 48: Task and Procedure weights for Mission.

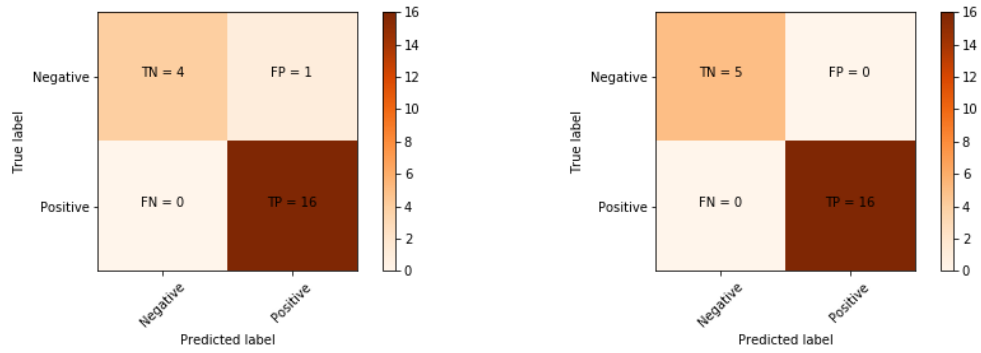
	$\boldsymbol{\delta_1}$	$\boldsymbol{\epsilon_1}$	$\boldsymbol{\epsilon_2}$	$\boldsymbol{\epsilon_3}$
$\boldsymbol{M_1}$	0.1165	0.1048	0.3229	0.4404

Following the same process explained for the previous components of Global Mission Assessment, predictions are made using the test set, including the known values of the weights. The different scores are collected in Table 49, obtaining the same conclusion than for Actions, Procedures and Tasks: Gradient Boosting provides the best predictions.

Table 49: Mission precision scores.

	Random Forest Score				**Gradient Boosting Score**			
	Accuracy	Precision	Recall	F1	Accuracy	Precision	Recall	F1
$\boldsymbol{M_1}$	0.9524	0.9412	1.0	0.9697	1.0	1.0	1.0	1.0

Looking at confusion matrices, represented in Figure 67, it is observed that both algorithms provide low values for false positives, however, Gradient Boosting does not have any of them, being the best one for the type of data used in this project.



(a) (b)

Figure 67: Global Mission confusion matrices for Random Forest (a) and Gradient boosting (b) classifiers.

5.3.5 Global Mission Assessment

Finally, knowing that Gradient Boosting Classifier is the algorithm which provides the best prediction results, it is possible to perform a complete automatic Global Mission Assessment.

Starting with the information originated from the recorded signals of the flight recorders. This data is used to generate the Flight Facet, and their assessment is made taking into account the Actions that compose the Mission, since Flight Facet Assessment is performed for a the specific time interval of each Action.

Once the aforementioned assessment is obtained, it is possible to do Action Assessment. To do so, the value of the Flight Facet's weights for every Action must be known, and the prediction is fulfilled using Gradient Boosting algorithm. The desired result consists in the assessment of each Action.

From known Action Assessment, it is possible to perform the same process to obtain Procedure and Task Assessment, using the values of Actions' weights and the same algorithm as before.

Finally, once the assessment for all the components is obtained, Global Mission Assessment is performed: A prediction on wether the Mission is satisfactory or not, considering each one of the previous assessed elements.

Figure 68 is the representation of the model developed for this project, when a new Global Mission Assessment is required.

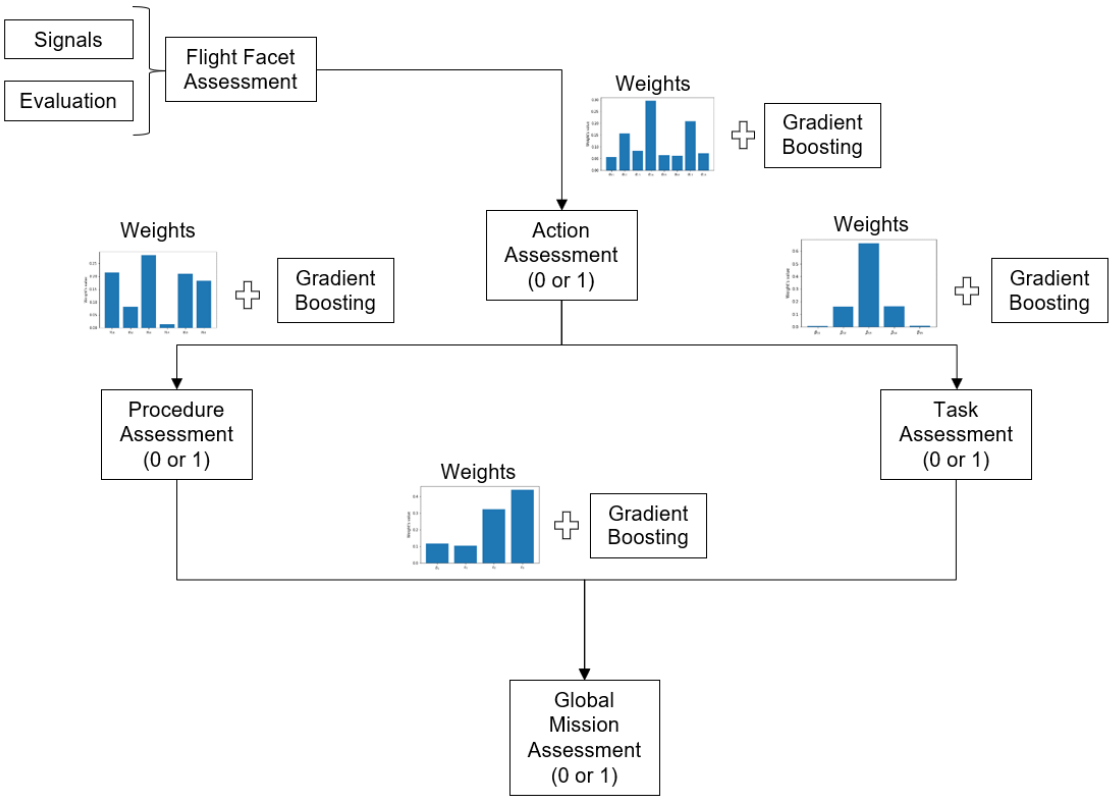


Figure 68: Representation of all the steps to perform Global Mission Assessment.

6 Conclusions

Mission Assessment is an important technique which allows improve the development of future Missions. It has been used by Air Forces to understand the circumstances of Missions to discover root causes and/or hidden factors. The information needed to apply this assessment is provided using Mission Debriefing Systems, however, since the amount of data keeps growing with time, it is becoming more and more complex to perform this assessment in a traditional way.

Taking this into account, a formal model is created by dividing Mission Assessment in the assessment of all the components that compose a Mission. This system starts with the creation of Flight Facets as a collection of all the signals which share common attributes. Then, for the successive components a mathematical model is defined to prove how Mission Assessment can be determined hierarchically, since a Mission is dependent on how Procedures, Tasks and Action were executed.

It has been proved that Machine Learning techniques can be applied to this type of problem, since it is possible to follow step by step all the stages that compose its lifecycle, which is adapted to the problem of this project. In particular, one of the most important steps is the selection of an appropriate algorithm, since this election is fundamental when performing the training process and the prediction, as each of the algorithms has its own features and behaves different depending on the data used.

One of the main drawbacks is data acquisition, as current Mission Debriefing Systems get just a portion of all the recorded signals. So, it is crucial to enhance these systems in order to have all the information about the development of a Mission. Due to this missing data, nowadays, this assessment is applied to Missions whose signals are available. Then, it is necessary to make improvements in the Debriefing process and with that, it will be feasible to perform Global Mission Assessment to all future Missions.

Also, it has been demonstrated that the application of Machine Learning techniques make possible a time and cost reduction, since the algorithm performs assessment predictions almost instantly. This fact allows more time to apply improvements when executing a Mission, if necessary.

Additionally, the results obtained in this project for Global Mission Assessment is just 0 or 1, unsuccessful and successful result respectively. So, one goal to be fulfilled in the future is to obtain the outcomes in terms of probability, in order to have more accurate definition for the assessment performed.

Finally, it is important to mention another crucial objective, which is to implement this

model in the opposite way. Starting from a known Mission Assessment, the goal is to detect what has caused a failure. Then, as it is shown in Figure 69, analyzing the assessment of each component from Mission to Flight Facet, it will be possible to discover the source of the issue, i.e. hidden factors.

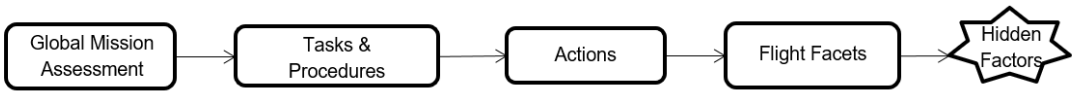


Figure 69: Representation of applying Global Mission Assessment backwards.

It is important to mention that this thesis is part of an Airbus Research and Technology group of projects, which main goal is to apply new technologies and tools, in order to improve tactical mission awareness. Because of this, I have had the opportunity to learn from people with great experience in this field, and to know how a project is carried out in a large company like Airbus.

References

[1] https://www.python.org/about/

[2] A. Tewari *Modern Control Design with MATLAB and SIMULINK*, John Wiley and Sons, LTD, 2002.

[3] https://www.skybrary.aero/index.php/Flight_Data_Recorder_(FDR)

[4] National Transportation Safety Board, *Cockpit Voice Recorders (CVR) and Flight Data Recorders (FDR*, https://www.ntsb.gov/news/pages/cvr_fdr.aspx

[5] M. Dub and J. Parizek, *Evolution of Flight Data Recorders*, Faculty of Military Technology, University of Defence in Brno, Czech Republic. Advances in Military Aircraft, Vol.13, No. 1, 2018.

[6] I. Moir and A. Seabridge *Aircraft Systems: Mechanical, electrical, and avionics subsystems integration, Third Edition*, John Wiley & Sons Ltd, The Atrium, Southern Gate, Chichester, West Sussex PO19 8SQ, England, 2008.

[7] Jun Wu, *AI, Machine Learning, Deep Learning Explained Simply.*, https://towardsdatascience.com/ai-machine-learning-deep-learning-explained-simply-7b553da5b960

[8] https://www.expertsystem.com/machine-learning-definition/

[9] Jason Brownlee, *Overfitting and Underfitting With Machine Learning Algorithms*, https://machinelearningmastery.com/overfitting-and-underfitting-with-machine-learning-algorithms/, 2016.

[10] Georgios Drakos, *Cross-Validation*, https://medium.com/@george.drakos62/cross-validation-70289113a072?, 2018.

[11] https://i.stack.imgur.com/F1QgU.png

[12] Sudeep Agarwal, *Understanding the Data Science Lifecycle*, http://sudeep.co/data-science/Understanding-the-Data-Science-Lifecycle/

[13] Saurav Kaushik, *Introduction to Feature Selection methods with an example (or how to select the right variables?)*, https://www.analyticsvidhya.com/blog/2016/12/introduction-to-feature-selection-methods-with-an-example-or-how-to-select-the-right-variables/, 2016.

[14] Sunil Ray, *Understanding Support Vector Machine algorithm*, https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/, 2017.

[15] *Clasificar con K-Nearest*, https://www.aprendemachinelearning.com/clasificar-con-k-nearest-neighbor-ejemplo-en-python/, 2018.

[16] Aishwarya V Srinivasan, *Stochastic Gradient Descent*, https://towardsdatascience.com/stochastic-gradient-descent-clearly-explained-53d239905d31, 2019.

[17] Jason Brownlee, *Logistic Regression for Machine Learning*, https://machinelearningmastery.com/logistic-regression-for-machine-learning/, 2019.

[18] Sunil Ray, *Powerful Guide to learn Random Forest*, https://www.analyticsvidhya.com/blog/2015/09/random-forest-algorithm-multiple-challenges/?utm_source=blog&utm_medium=understandingsupportvectormachinearticle, 2015.

[19] Harshdeep Singh, *Understanding Gradient Boosting Machines*, https://towardsdatascience.com/understanding-gradient-boosting-machines-9be756fe76ab, 2018.

[20] Dataflair Team, *Top 11 Machine Learning Software*, https://data-flair.training/blogs/machine-learning-software/, 2019.

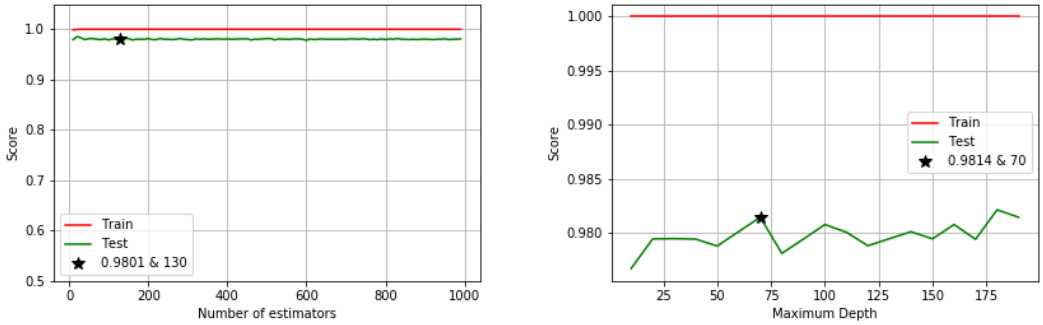
[21] Juan Zamorano Ruiz, *Comparativa y análisis de algoritmos de aprendizaje automático para la predicción del tipo predominante de cubierta arbórea*, Universidad Complutense de Madrid, Departamento de Sistemas Informáticos y Computación, 2018.

[22] Saul McLeod, *What does a box plot tell you?*, https://www.simplypsychology.org/boxplots.html, 2019.

Appendices

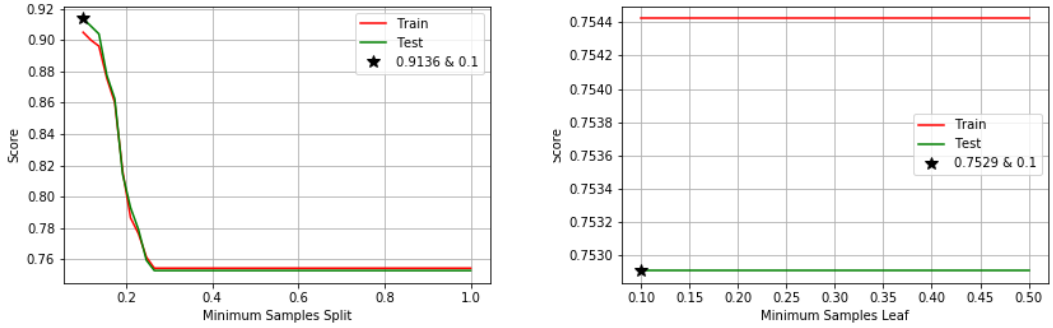
A Action hyper-parameters

A.1 Random Forest Classifier



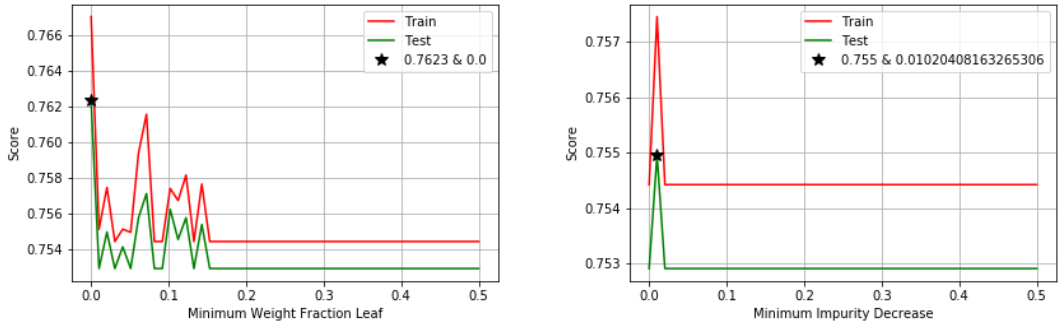
(a) (b)

Figure 70: Effect of *n_estimators* (a) and *max_depth* (b) in Action Assessment score.



(a) (b)

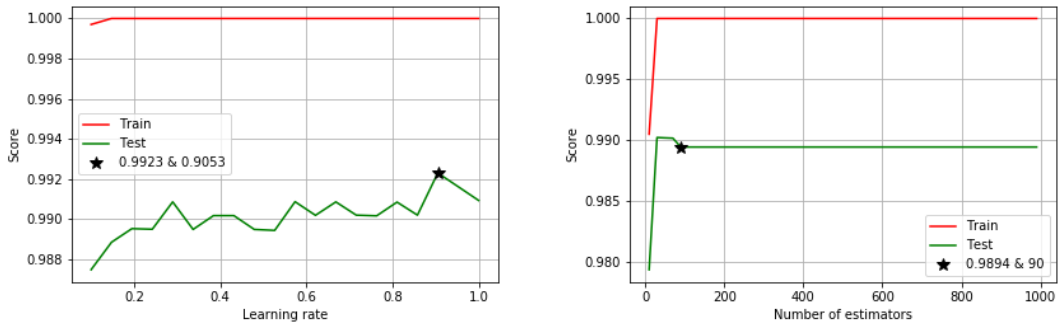
Figure 71: Effect of *min_samples_split* (a) and *min_samples_leaf* (b) in Action Assessment score.



(a) (b)

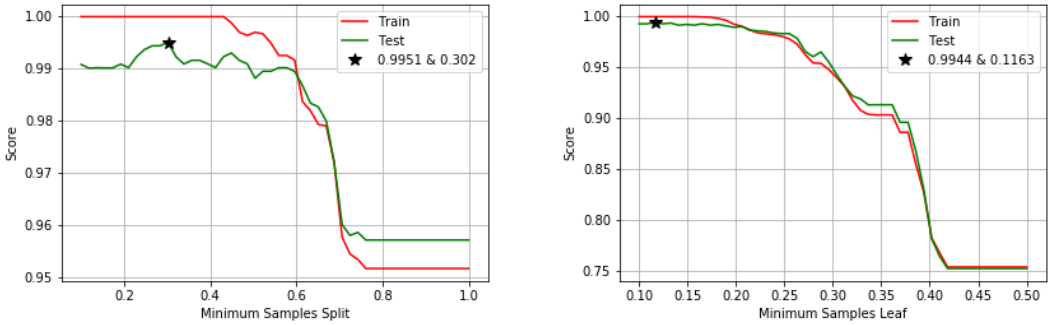
Figure 72: Effect of *min_weight_fraction_leaf* (a) and *min_impurity_decrease* (b) in Action Assessment score.

A.2 Gradient Boosting Classifier



(a) (b)

Figure 73: Effect of *learning_rate* (a) and *n_estimators* (b) in Action Assessment score.



(a) (b)

Figure 74: Effect of *min_samples_split* (a) and *min_samples_leaf* (b) in Action Assessment score.

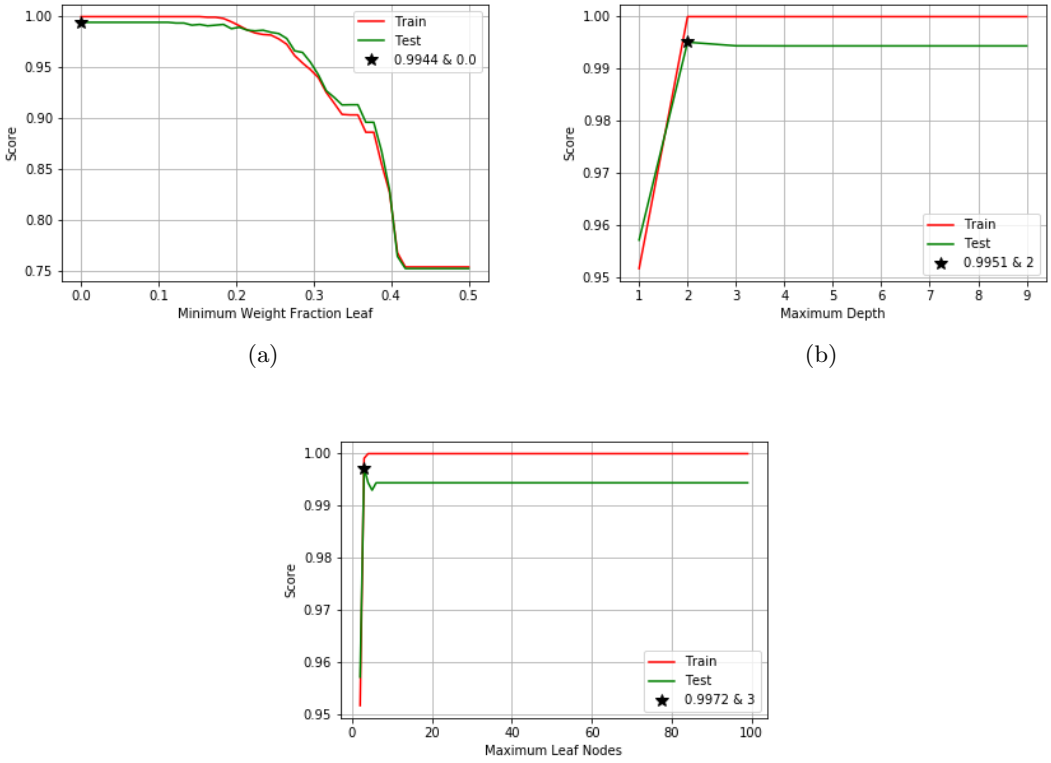
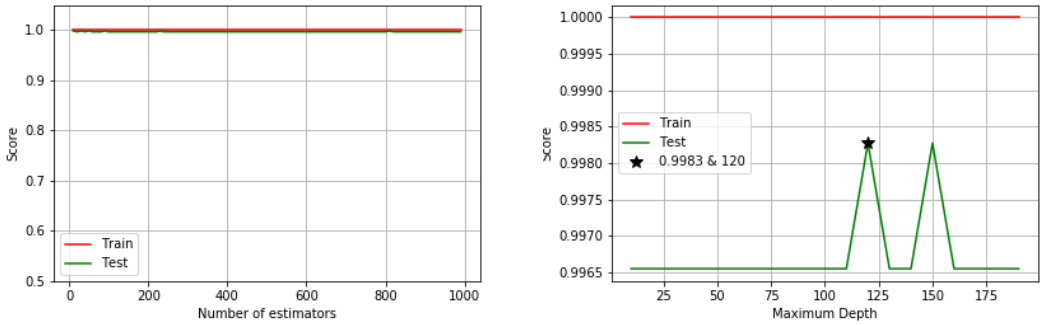


Figure 75: Effect of *min_weight_fraction_leaf* (a), *max_depth* (b) and *max_leaf_nodes* (c) in Action Assessment score.

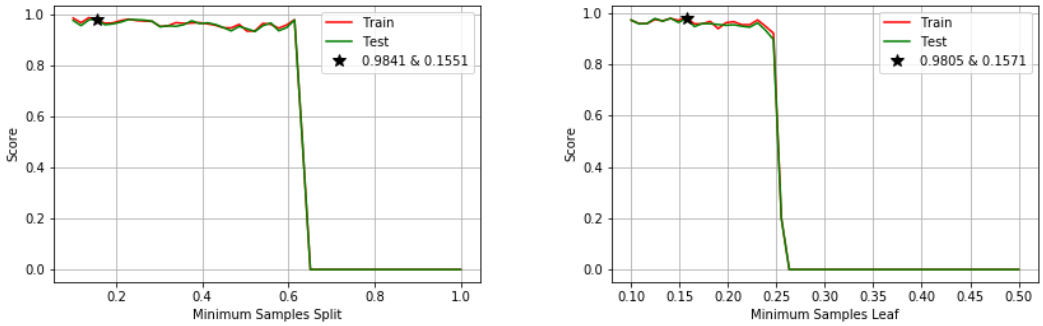
B Procedure hyper-parameters

B.1 Random Forest Classifier



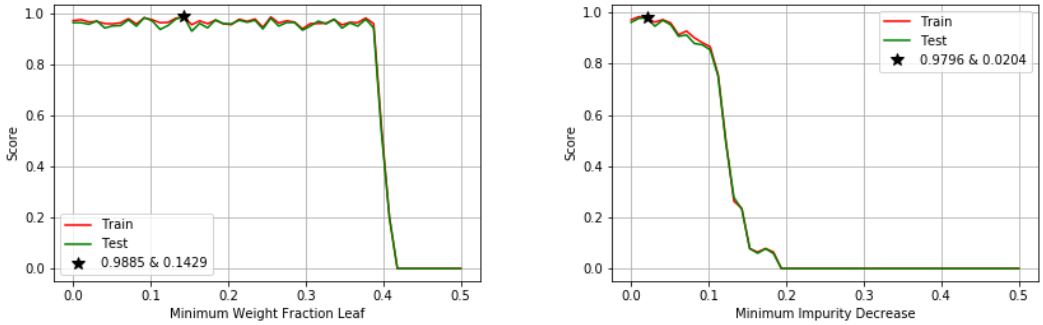
(a) (b)

Figure 76: Effect of *n_estimators* (a) and *max_depth* (b) in Procedure Assessment score.



(a) (b)

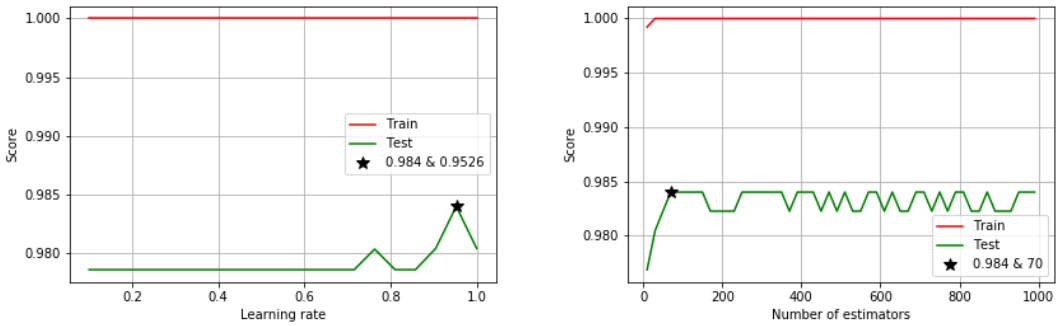
Figure 77: Effect of *min_samples_split* (a) and *min_samples_leaf* (b) in Procedure Assessment score.



(a) (b)

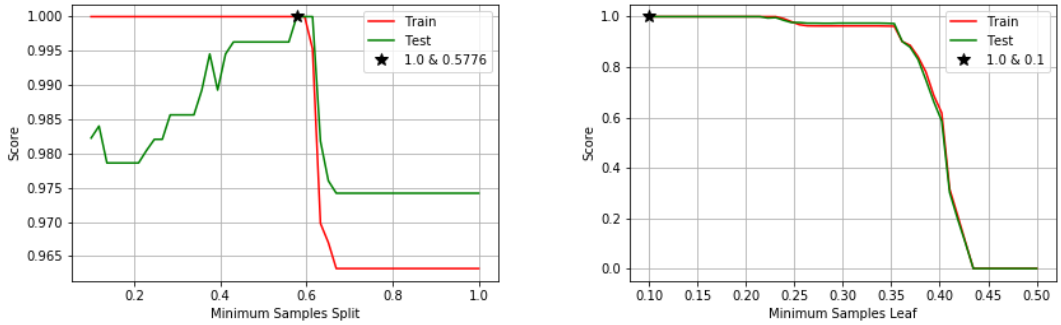
Figure 78: Effect of *min_weight_fraction_leaf* (a) and *min_impurity_decrease* (b) in Procedure Assessment score.

B.2 Gradient Boosting Classifier



(a) (b)

Figure 79: Effect of *learning_rate* (a) and *n_estimators* (b) in Procedure Assessment score.



(a) (b)

Figure 80: Effect of *min_samples_split* (a) and *min_samples_leaf* (b) in Procedure Assessment score.

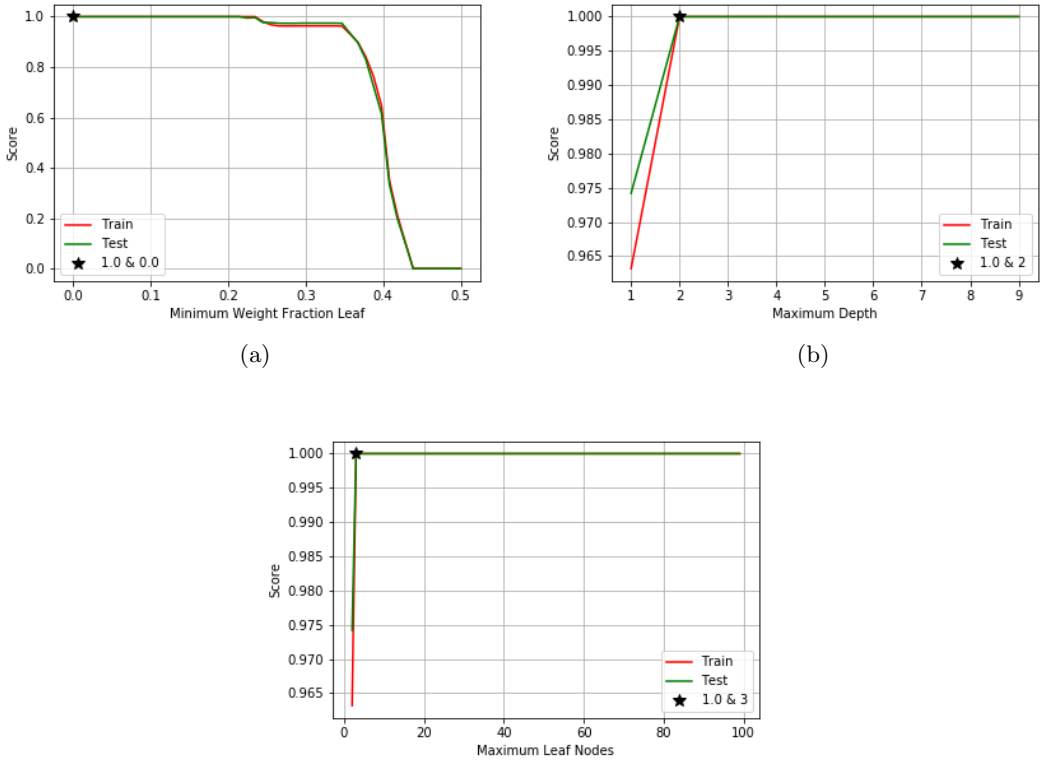
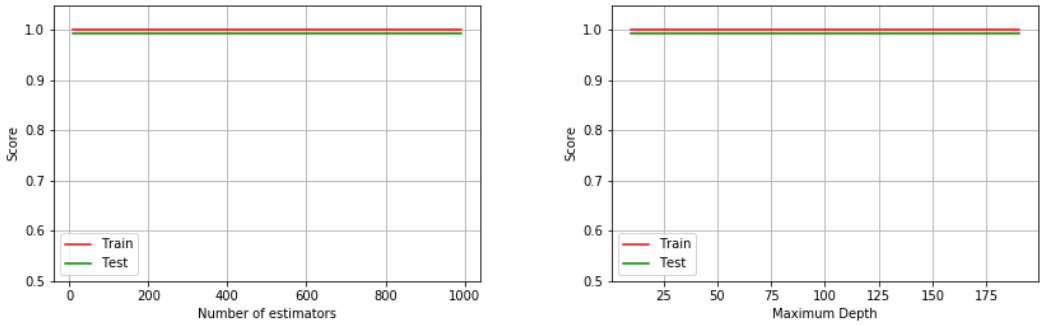


Figure 81: Effect of *min_weight_fraction_leaf* (a), *max_depth* (b) and *max_leaf_nodes* (c) in Procedure Assessment score.

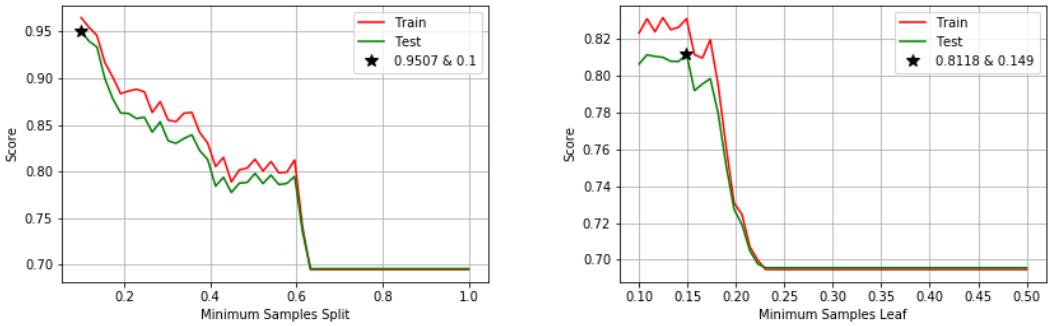
C Task hyper-parameters

C.1 Random Forest Classifier



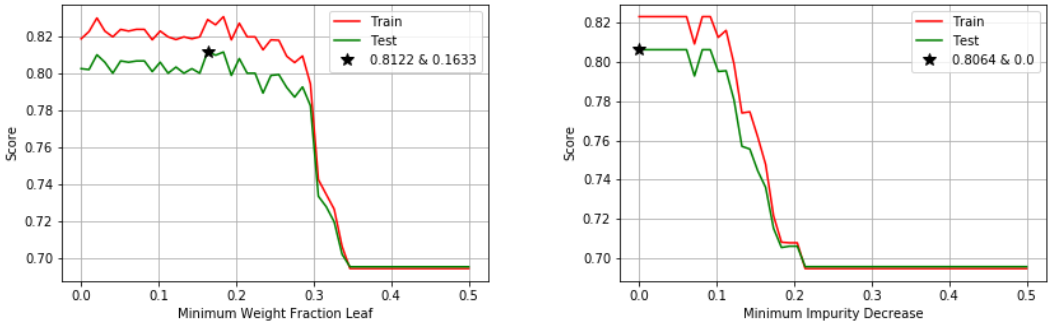
(a) (b)

Figure 82: Effect of *n_estimators* (a) and *max_depth* (b) in Task Assessment score.



(a) (b)

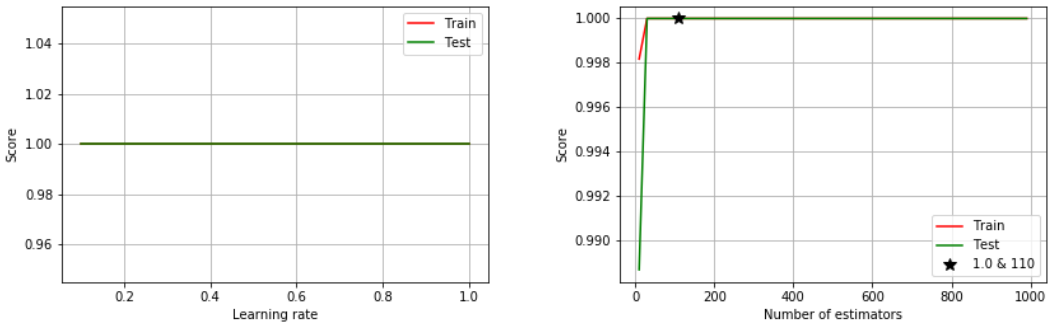
Figure 83: Effect of *min_samples_split* (a) and *min_samples_leaf* (b) in Task Assessment score.



(a) (b)

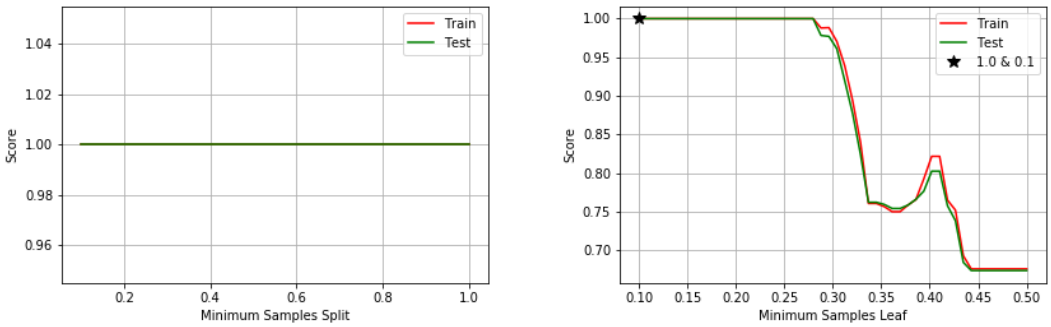
Figure 84: Effect of *min_weight_fraction_leaf* (a) and *min_impurity_decrease* (b) in Task Assessment score.

C.2 Gradient Boosting Classifier



(a) (b)

Figure 85: Effect of *learning_rate* (a) and *n_estimators* (b) in Task Assessment score.



(a) (b)

Figure 86: Effect of *min_samples_split* (a) and *min_samples_leaf* (b) in Task Assessment Score.

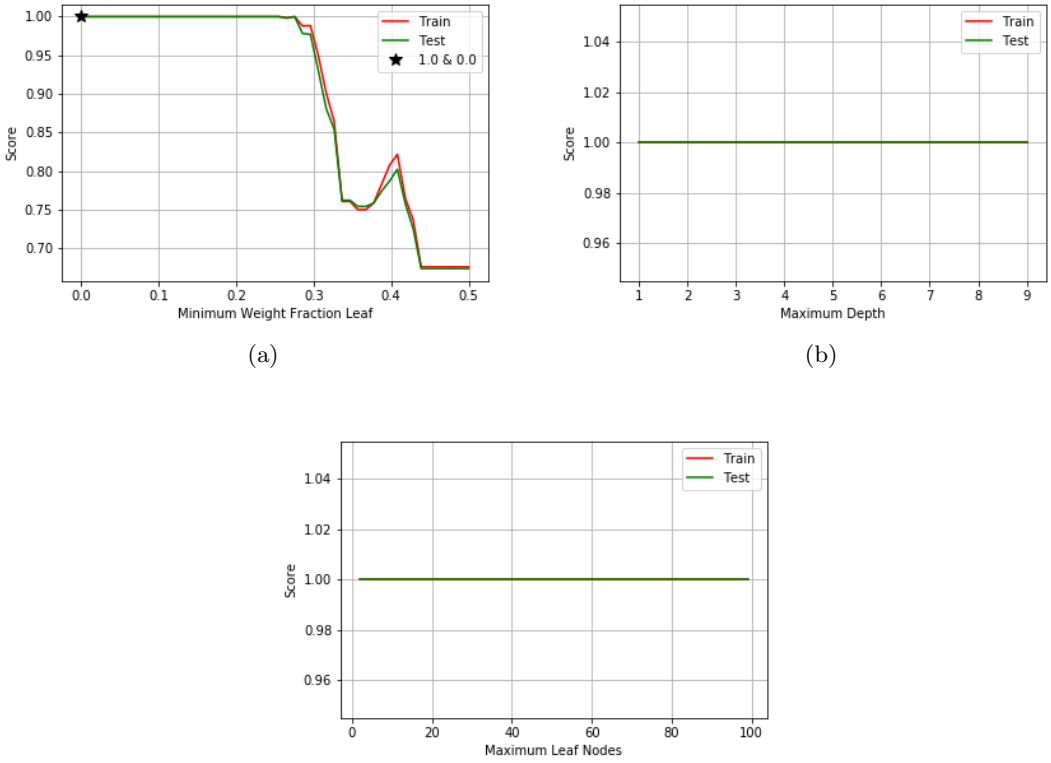


Figure 87: Effect of *min_weight_fraction_leaf* (a), *max_depth* (b) and *max_leaf_nodes* (c) in Task Assessment score.