Application of learning methods for predictive modelling with human factors in aviation industry

Rui Pedro dos Reis Nogueira

Thesis to obtain the Master of Science Degree in

Mechanical Engineering

Supervisors: Prof. Duarte Pedro Mata de Oliveira Valério
Prof. Mário Rui Melício da Conceição

Examination Committee

Chairperson: Prof. Carlos Baptista Cardeira

Supervisor: Prof. Mário Rui Melício da Conceição

Members of the Committee: Prof. André Calado Marta

Prof. Luís Filipe Ferreira Marques Santos

October 2021

To those who pull me up...

Acknowledgments

First of all, I would like to extend my honest gratitude to my supervisors, Prof. Duarte Valério and Prof. Rui Melício, for their patience, being constantly accessible, and their cooperation throughout this challenging journey with setbacks and progress.

Besides my supervisors, I would like to extend my thanks to Prof. Luís Santos for a welcomed assistance to improve this dissertation, and Tomás Madeira, whose impeccable work for his dissertation was an inspiration and motivation for my own work, and whose work helped shape the direction of this dissertation.

On a personal level, I would also want to extend my deepest gratitude to my family, whose resolve, perseverance and unconditional support was crucial for being able to reach this moment in my life, to my friends, whom from difficult circumstances and through these years, were still very supportive and shared phenomenal experiences, to all people who have had an impact in my life and shaped me. This accomplishment without them would not have been possible. Thank you.

Resumo

A aviação teve aumentos de procura consideráveis nos últimos anos, tendo padrões de segurança bastante rigorosos, para mitigar o risco e prevenir falhas humanas. Porém, existe a necessidade de desenvolver modelos preditivos capazes de prever potenciais falhas ou situações de risco para melhorar procedimentos em termos de segurança. Logo, o objectivo desta dissertação é propor modelos capazes de prever se uma ocorrência foi fatal e a sua dimensão, tendo em conta informação relevante do voo e factores humanos que precederam o acidente. Para tal, foi retirada do repositório da Aviation Safety Network (ASN), uma organização especializada em matéria de ocorrências de aviação, 1105 relatórios entre 2007 e 2017. Correlações entre falhas humanas e as causas contributivas da ocorrência foram propostas através da estrutura de classificação de fatores humanos chamada Human Factors Analysis Classification System (HFACS), e a base de dados é adaptada para aplicações em machine learning. Para a modelação, são utilizados algoritmos de aprendizagem supervisionada como Random Forest (RF) e redes neuronais artificiais (ANN) e semi-supervisionada como Active Learning (AL). São utilizadas funções de optimização da estrutura dos algoritmos para melhorar a performance do modelo, sendo medida através de métricas como precisão, sensibilidade, eficácia e F1-score. O melhor modelo preditivo (Modelo 1), através do RF, conseguiu eficácia de 90%, macro F1-score de 87% e sensibilidade de 86%, enquanto ANN, por menor capacidade de prever acidentes fatais, teve piores resultados. Para o Modelo 2, apenas AL obteve resultados promissores após bastante treino devido a um universo de dados mais pequeno.

Palavras-chave: segurança aviónica, modelos preditivos, fatores humanos, aprendizagem automática

Abstract

Aviation demands has increased over the years, and its safety standards are the most rigorous, by managing risk and preventing failures from human factors. However, there is more need to build models capable of predicting potential failures or risky situations in order to improve safety standards. The aim of this dissertation is to propose a model capable of predicting fatal occurrences and the degree of mortality, taking into account the human factors that contributed for the incident and information about the flight. The database was provided by the Aviation Safety Network (ASN), an organization which gathers reports about aviation occurrences, consisting of 1105 reports between 2007 and 2017.

Correlations between leading causes of incident and the human element are proposed, thanks to the Human Factors Analysis Classification System (HFACS). A classification model system is proposed, with the database preprocessed for the use of machine learning techniques. For modelling, supervised learning algorithms Random Forest (RF) and Artifical Neural Networks (ANN) and semi-supervised learning Active Learning (AL) are considered. For optimizing their respective structure, optimization methods are applied for hyperparameter analysis to improve the model. The performance is measure with precision, recall, accuracy and F1 score.

The best predictive model (Model 1), with use of RF, was able to achieve an accuracy of 90%, macro F1 87% and recall 86%, whereas ANN had worse results due to less ability for predicting fatal accidents. For Model 2, only AL had promising results after considerable training due to lower data sets.

Keywords: aviation safety, predictive modelling, human factors, supervised learning, machine learning

Contents

List of Tables

List of Figures

Nomenclature

AL	Active Learning
AMTs	Aviation Maintenance Technicians
ANP	Analytical Network Process
ASN	Aviation Safety Network
AUC	Area Under Curve
EASA	European Aviation Safety Agency
FAA	Federal Aviation Administration
FBN	Fuzzy Bayesian Network
GP	Gaussian Process
HFACS	Human Factors Analysis Classification System
HFACS-ME	Human Factors Analysis Classification System - Maintenance Extension
HRA	Human Reliability Analysis
ICAO	International Civil Aviation Organization
MLP	Multilayer Perceptron
MRM	Maintenance Resource Management
PSFs	Performance Shaping Factors
RF	Random Forest
ROC	Receiver Operating Characteristic
SMS	Safety Management System

Chapter 1

Introduction

1.1 Motivation

Air transportation has developed into a crucial method of long-distance travel, with widely known contributions for economic and social development in a global capacity. Technological and management systems in air travel have seen a significant improvements over the past 50-60 years due to the standardization of parts and processes, facilitated by a close relationship towards safety improvement by aviation manufacturers and regulators leading to one of the safest transportation methods (figure 1.1) [1].

As aircraft became remarkably reliable through the implementation and compliance of safety regulations by international regulatory agencies over the past 30 to 40 years, such as the Safety Management System (SMS) and Maintenance Resource Management (MRM), it became increasingly apparent that human factors had become the primary cause of accidents [2]. Due to the potential costs of an accident, from the price of an aircraft and its maintenance to, most importantly, the loss of human lives, it is argued that the industry must recognize as a philosophy that even one accident is already too much [3].

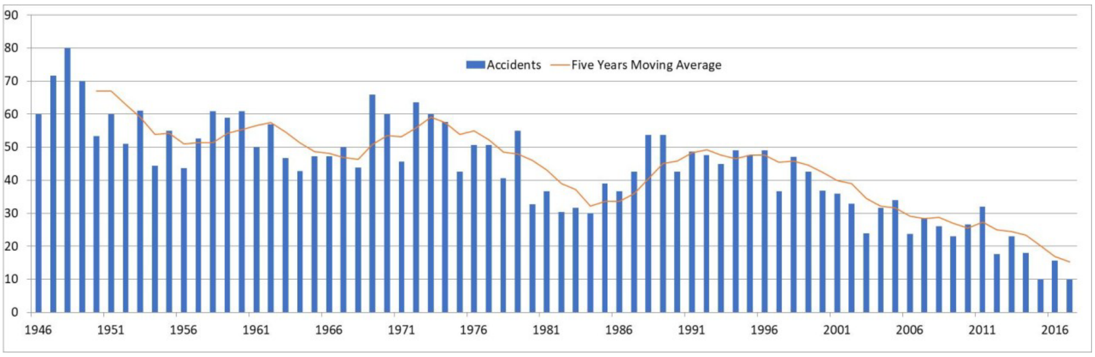


Figure 1.1: Number of fatal accidents between 1947 and 2016 (from Santos and Melicio [3])

However, with the sharp decrease of the accident rate, not only has the air traffic considerably increased but also the absolute number of accidents [4, 5], and market demands are such that it is required professionals in this industry who are available 24/7 and capable to work through large stretches of the day and/or night, and adequate regulation of the work effort and energy must be implemented [3, 6]. Approximately between 1975-1990, 70-80% of the accidents had human error causation, and studies within the industry helped prove a link with the causation of human error in aviation and the added work effort requirements [3, 4, 6, 7].

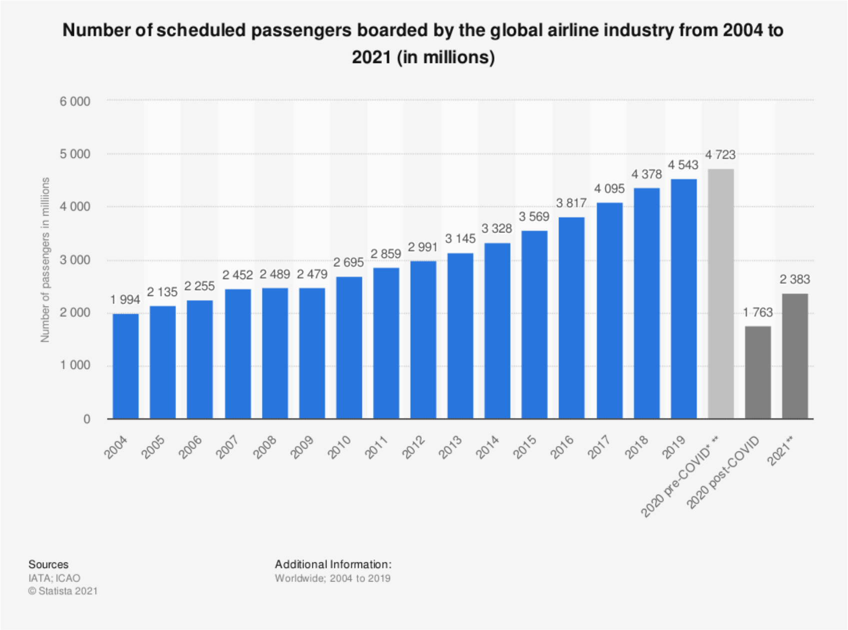


Figure 1.2: Air passenger traffic between 2004 and 2021, from Mazareanu [5]

Even though these reactive systems have been implemented with success, in order to further improve for achieving the doctrine described above, there is a need to build prospective models that are able to systematically reduce the risk of fatal accidents, by developing predictive together human factors models [8].

1.2 Problem Characterization

The problem addressed with this thesis is the modelling of a classification system that encompasses human factors with the circumstances of an aviation incident or accident, with the intent of building a predictive system that helps with increasing safety standards within the industry. In order to build that model, a database provided by the Aviation Safety Network (ASN). Unlike the work made by Madeira et al. [9], where the database was processed with the text reports from each occurrence, the database was expanded to include information regarding the state of the aircraft, damage and existence of casualties. With this information it was possible to build 2 models: the first one (Model 1) was to predict whether an occurrence was fatal and the second (Model 2) was to predict the proportion of people killed for a fatal occurrence.

For classification techniques, supervised and semi-supervised learning algorithms were used. Random Forest (RF) and Multilayer Perceptron (MLP), which is a type of neural network (NN) were used as the supervised learning for both models. The semi-supervised learning algorithm that was used for both models is the Active Learning (AL), that trains itself by querying data to be tested. All algorithms were developed and manipulated using Python.

1.3 State of the Art

1.3.1 Aircraft Industry and Safety

The ICAO defines an aircraft as a machine capable of fly gaining support from the reactions of air other than the reactions of air against the Earth's surface [1]. There are substantially different machines that can be deemed aircraft so the FAA defines aircraft classification as a broad organizing of aircrafts based upon their propulsion, flight or landing [10]. Airworthy is an aircraft in condition for safe operation, which meets all requirements and regulations enforced by respective state's regulatory agency and are recommend by ICAO [10]. There are three different avenues in which aircrafts may be utilized: military, civil and experimental.

One type of aircraft is the fixed-wing aircraft, in which the structure lifts because of the forward motion through the air, and aeroplanes and airplanes fall under that category, so for commercial use it is the most common type of aircraft. Another type of aircraft is the rotorcraft, which lifts thanks to the side rotary motion produced, so in terms of mechanical systems these two types of aircrafts are different. Rashid et al. made a statistical analysis in maintenance in helicopters, which are rotorcrafts, and they argued that, because of rotary wing systems, the findings of the analysis must take in consideration its special nature and criticality [11].

The FAA and EASA defines maintenance as any one or combination of the following operations (except pre-flight inspection) of either the entire aircraft or one of its structures: defect rectification, inspection, overhaul, repair or replacement [10, 12]. This means all maintenance procedures must be in compliance with thorough regulations, so the technical operations necessary to follow procedures shapes aircraft maintenance into a complex industry, where its organizational structure is critical [13].

Safety in aircraft industry is currently defined as the risk in which all aviation activities, either directy supports or are related to the operation of the aircraft, are reduced and controlled to an acceptable level [1]. In engineering, risk is perceived as a unique phenomenon off a process at a rate that can be measured through risk assessment [14].

Inadequate approaches to risk may manifest themselves into hazardous situations. Hazard is defined as a potential source for harm or serious injury on a person or people, which in the case of aviation hazard manifests as either accident or incident [15]. Accident is defined by the ICAO as an occurrence associated with the aircraft during the time any person boards it with the intent to fly, in which either a person may be seriously or fatally injured, or the aircraft suffers significant structural damage or it is missing [16]. An incident is defined as an episode where the safety of operating the aircraft may be affected [16].

Safety culture can be characterized as the set of behaviors, personal attitudes, social and technical proceedings concerned with diminishing potential conditions that are considered dangerous or can lead to injury, thus giving perception to an organization of hazard in operational process [17, 18]. Pidgeon [18] argues that the definition of safety culture is subjective, depending on values, costumes and routines of the stakeholders, justifying with the example of the Chernobyl disaster, where an investigation proved a difference between safety procedures in western Europe and the USSR [18].

1.3.2 Human Factors Models

The Reason Model, often referred as the ”Swiss Cheese” model, was designed for information processing related to the type of errors. It defines two types of approaches regarding errors: the person approach, where the focus is solely on the individuality, such as personal moral weakness, forgetfulness and distraction, and the system approach, where the focus lies on the conditions which promotes human error, with the intention of building layers of defense to manage risk and mitigate hazard, so it is a safety management model [19]. In regarding of getting to the genesis of the error, the first layer explores the active error that led to the accident, the *Unsafe Acts* of Operators, however there is a likelihood that there were more unsafe acts prior to the accident, which are the ”holes” in the cheese [19].

Reason [19] argues in his model that there are within a system latent failures which may lay dormant for a long time because of a flaw of design of a certain task or an improper routine behaviour by an operator that, if not address, may propagate itself to either a direct error or more latent failures which in turn propagate again into direct errors. Reason describes three layers of latent failures: the first one is *Preconditions for Unsafe Acts*, which covers poor communication practices and mental fatigue, the second is *Unsafe Supervision*, which explains why poor communication and/or coordination happened or why the crew might be destined to failure, and the third layer is *Organizational Influences* which looks at whether or not the organization can impact these type of errors [20].

That model was created for a power plant, but since it is a very theoretical model there are applications for plenty of fields [20]. For example, a technical report focusing on tackling fatigue in maintenance and repair operations, it is applied the Reason model for layers of protection from fatigue-related errors in which there are three layers [7]: the first layer is reducing fatigue, which implements fatigue countermeasures, the second one is reducing or capturing fatigue-related errors where it tries to reduce the probability of error by a maintainer or capturing the mistake once it is made, and the third one is aimed to minimize the harm caused by errors. Another interpretation of the holes in the ”Swiss Cheese” model is that the measures proposed that compose the layers are fallible meaning that there will be failures that will penetrate the first layer of defense and the remaining layers will try to impede further permeation through them [7]. This is a small example of how the concept of this model can be shifted, however one of the criticisms is that it is too theoretical which makes it difficult for real-life application.

Another methodology is the Human reliability analysis (HRA) where it is assumed that human performance depends on conditions under which the tasks or activities are carried out, defined as performance shaping factors (PSFs). A framework was modeled to determine the importance of those PSFs, with the case study of air traffic control for testing the model [21]. The importance was estimated through hierarchical comparison with the Analytical Network Process (ANP), an extension of the Analytical Hierarchical Process [22], which creates the dependencies of all PSFs in a network fashion, therefore it provides an overall view of the relationships inherent of the problem to tackle and it helps the decision maker evaluate the magnitude of issues [21, 22].

Human Factors Analysis and Classification System

Considering as cited that most accidents in aviation can be blamed at least in part on human factors, this model tries to provide a framework for human error so that its causality in accidents can be measured and assessed [20, 23]. The model is a evolution of the taxonomy of unsafe operations [4] in which is an exploration of all possibilities in which errors can occur from a human level [24] and its interactions can be described through the SHEL model [25]. The SHEL model is the relationship of four factors in maintenance that composes an accident: Software(e.g procedures, checklist layout, manuals), Hardware(e.g tools, test equipment, operating sense of controls and instruments), Environment (work patterns, management structure, physical environment of a hangar) and Liveware (person or people at the center of model) [25, 26]. The taxonomy compiles those relations through a sequential framework, achieving three levels of potential error (figure1.3), with each level being increasingly specific and descriptive, from supervisory practices to operators actions, because its failures might lead to an accident [24]. The Human Factors Analysis and Classification System (HFACS) framework was tested on accidents of both the military and in particular on the commercial airliner's aircrew, with similar results [27], as the framework proved to be reliable in identifying human factors issues that were hidden, highlight critical parts of human factor failure that required intervention and improved data quality and quantity.

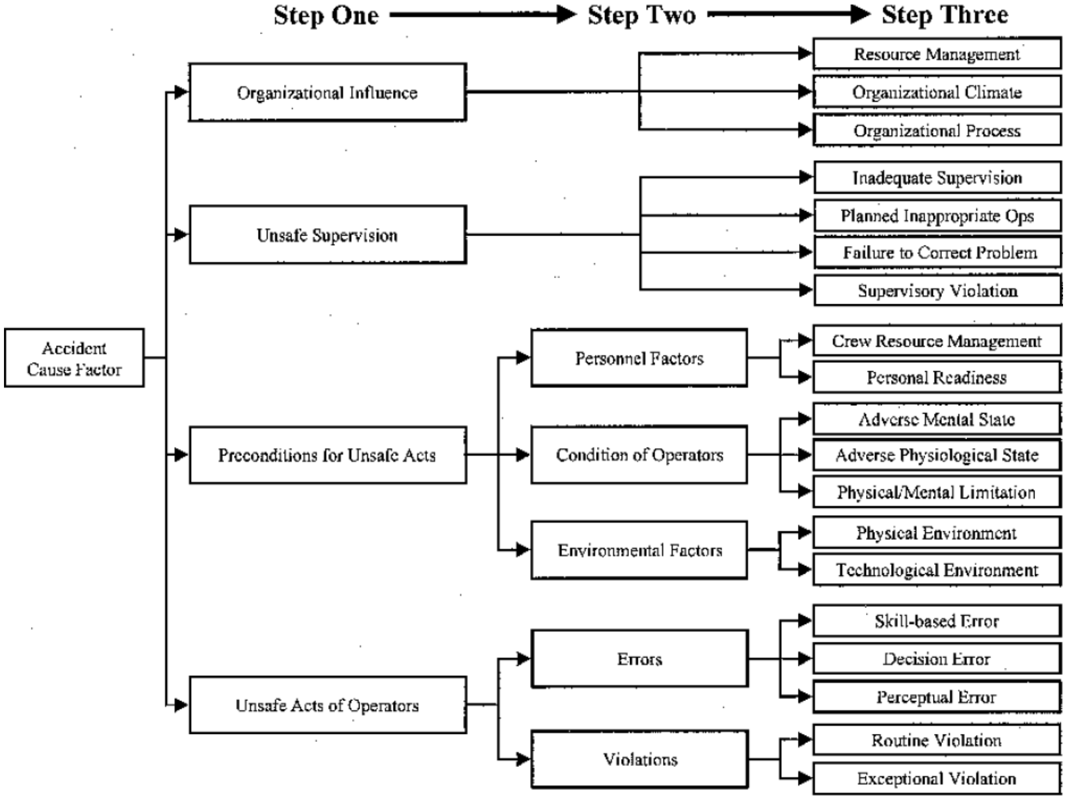


Figure 1.3: The HFACS framework after the application of the taxonomy [4]

Since its inception, there have been attempts to make the taxonomy even more comprehensive in order to better analyze the causes of accidents, and a fifth element was proposed in Step One as seen in figure 1.3, as the authors considered external factors such as legislation gaps, administration oversight failures and social factors [28].

The model, despite being developed for aviation analysis, throughout the scientific literature has seen a variety industries adapting its taxonomy. For maintenance related mishaps in aviation, the *Maintenance Extension* (ME) in HFACS-ME was created [29]. It maintains the same principles of the Reason model, but the terminology is different: the latent mishaps leading to an active failure are Supervisory, Maintainer and Working Conditions with the Maintainer Act being the active failure [29]. Schmidt et al. [29] conducted a study on several maintenance mishaps that were reported and like the HFACS, it was successful in building a profile of the nature of the errors and accurately related the latent conditions and active failures in all classes of severity of the mishaps, which validates the model.

More recently in 2010, Rashid et al. [11] has proposed an improvement of the taxonomy of the HFCAS-ME in the form of a fourth order in a study of maintenance errors in the helicopter industry. This fourth order was designated as *Specific Failures* are the "latent conditions or active failures that immediately precede maintenance-related safety occurrences", accumulating causal factors of human errors for a "higher resolution of analysis".

Even more recently (2018), a taxonomy for human risk factors for Aviation Maintenance Technicians (AMTs) was developed with an holistic perspective by considering human factors in ergonomics and corporate performance feature [30]. The aim was developed to involve all organization stakeholders, regulators and manufacturers for the taxonomy, with a strategy of mitigating causal risk factors. This helps to present a framework that manages corporate sustainability strategies and technicians' well-being, improving organizational management practices and sharing new sources of error [30]. The model is developed through a function with dependent variable y and independent variable x:

$$y = f(x) = \sum_{i=1}^{n} x_i \tag{1.1}$$

where y can be defined as flight safety risk parameters (organizational culture and human performance) and x are the factors that shape human risk in maintenance procedures, like environmental factors, corporate social responsibility, physical/psychological condition of AMTs, hazardous attitudes and task-related factors, among others. Another study performed was a survey conducted on experts in the maintenance industry, ranking risk factors among each other to focus on managerial and operational weaknesses to improve operations with limited resources [31].

Application of Predictive Models

There have been attempts to build prospective models to reduce human risk factors. For accident prevention in aviation, it was proposed the Aviation Maintenance Monitoring Process (AMMP), for the purpose of a proactive oversight of human error causal factors [8]. The process was built upon the ANP model because the software works as a decision-making tool with multi-criteria that has interdependence between them, considering the causal risk factors through accident reports made by Rashid et al. [8, 11].

For the high-risk chemical processing industry, where human factors have a significant contribution to unsafe operation of the process system, models were implemented using Fuzzy Bayesian Network (FBN) and HFACS [32, 33]. They were tested on accidents occurred operating a natural gas pipeline by modifying the HFACS taxonomy for the Oil and Gas (O&G) industry [32], and was also used for human and organizational factors for the same industry, using as case study a release of liquid hexane from a storage tank [33]. They were also capable of synthesizing the human factors of both case studies into specific causal factors, such as routine violations or operational processes, which the models had signaled to be more prevalent [32].

For air traffic control operators, there was a network-based approach to deal particularly with fatigue as human risk, with use of the artificial immune system method with extreme gradient boosting algorithm for its implementation. The network is consisted of all factors that can add up to the increase of fatigue, such as environmental factors(for example temperature, weather, humidity), working conditions, sleep patterns and personal issues outside of the workplace, with the conclusion that around 27% of operators could reduce their fatigue by shifting their responsibilities [3, 34].

There have been usages of predictive models using neural networks, such as an integrated model to determine the risk of error in maintenance procedures. A model of maintenance mistakes was proposed and it has a similar structure to the HFACS taxonomy, and its parameters are then considered the structure of the neural network function, where the evaluation results are computed from [35]. Another methodology was used with neural networks and HFACS to build a model for human factors evaluation in maritime accidents [36]. The HFACS taxonomy was re-designed onto a fault tree analysis structure, where factors were broke down into basic, intermediate and top events, because this helped to develop the structure of the neural network, with satisfactory results in terms of dealing with uncertainties and dynamics of the problem being studied and the models developed. However, it was noted the need to gather more data for a more comprehensive analysis of the methodology's impact [36].

There have been also approaches in developing new models, most notably semi-supervised learning algorithms. For example, in studying human factors in the aviation industry, an approach was made by extracting text data from reports using text related methods such as Natural Language Processing, and modeling it using semi-supervised methods, such as Label Spreading and supervised learning algorithms such as Support Vector Machine to classify and predict human factors [9]. Another application of a semi-supervised learning method was the incorporation of a fault classification system called Fisher discriminant analysis in a industrial plant with the Active Learning (AL) method. It was able to improve the overall performance with the learning of new data previously unavailable, and despite having limitations with non-linear data, which is the most conventional type in industrial data, the experiment was proved to be valid and successful [37].

1.4 Thesis Outline

As stated in section 1.2, the aim of this thesis is to provide a predictive model for accident and incident occurrences in the aviation industry. In chapter 2, it is described with detail the algorithms used to implement the predictive models, with explanations of how Neural Networks, Random Forests and Active Learning work. There is also a description of criteria commonly used to evaluate the performance of the models, taking into account that the problem at hand is a classification one. Chapter 3 has a detailed explanation of how the ASN's database was expanded and the pre-processing of data regarding human factors. Chapter 4 will have the results of models applied and their validity is determined, and the optimization of the algorithms considering the database is also achieved, and finally the last one (chapter) will conclude whetger the aims proposed for this thesis were achieved and future references to build upon the results and findings of this thesis are suggested.

Chapter 2

Learning Algorithms for Predictive Models

In the previous chapter, in section 1.3, several paths were described on how predictive models were implemented in industries with inherent risk safety hazards and also how prospective models have been used in the aviation industry to improve safety standards. Despite the existence of physical models capable to assess risk behaviour conducing to safety risks [3, 7], data-driven modelling has been more popular as a method to solve new problems. It is possible to implement models of this type through supervised learning algorithms, a type of machine learning where a model is tested through known data and untested data is then used to validate them. There are three main types of supervised learning algorithms that can be applied:

- Support Vector Machine (SVM)
- Random Forest (RF)
- Artifical Neural Network (ANN)

Of the three algorithms mentioned, only two are going to be implemented with this work. The first one (SVM) has been applied to similar work for the aviation industry, studying labelling of human factors on reports of aviation occurrences [9]. The other two models are very adaptable to a variety of problems, with greater computational quality. In section 2.1 a description of how the RF algorithm functions and of its capabilities are presented. In a similar way, section 2.2 also goes into detail on the mechanics of the ANN algorithm and how it works.

Semi-supervised learning has garnered more attention recently as an alternative to algorithms where no set of rules is necessary and the modelling process is conditioned, and as an alternative to algorithms that require much more data that may not be available. In section 2.3, Active Learning (AL), a semi-supervised learning algorithm, is presented. Finally, section 2.4 focuses on how these supervised learning algorithms can be optimized by dealing with intrinsic parameters capable of being shaped before initiating its learning process.

2.1 Random Forest Algorithm

Random Forest is a supervised learning algorithm introduced by Breiman in 2001 made up of a collection of tree-structured classifiers, which are defined as decision tree, applied throughout a given dataset on multiple sub-samples [38, 39].

2.1.1 Decision Tree

The decision tree is built up from a number of nodes, connected by branches, descending from the root node, placed at the top by convention, to the leaf nodes [40]. Features are tested at the decision nodes, leading into a branch. Those branches can lead up onto another decision node or concluding in a leaf node. The algorithm represents supervised learning, which requires a training data set provided with values of the target variable.

Originally, the specific decision trees to be used are the Classification And Regression Trees (CART) algorithm, where each decision node produces two branches, so the tree is binary. Its growth happens through an ”exhaustive search of all available variables and all possible splitting values”, selecting the optimal measurement vectors that reduces the highest impurity [40, 41]. The process to generate a decision tree starts with splitting the root node into binary pieces. The splitting procedure is:

$$\Delta i(s,t) = i(t) - p_L i(t_L) - p_R i(t_R) \tag{2.1}$$

where s is the candidate split at node t, $\Delta i(s,t)$ is a measure of impurity reduction from split s, $i(t)$ represents the impurity before splitting, and $i(t_L)$ and $i(t_R)$ show the impurity of the left child node t_L and of the right child node t_R after halving node t by split s.

In order to measure these impurities, there are several approximations [41], however the criterion for split is by default the Gini impurity [39], which measures how often a randomly chosen element from a set would be incorrectly labeled from a random distribution of labels in the subset. It is calculated by computing the probability p_i of an item with label i being chosen times the probability $\sum_{k \neq i} p_k = 1 - p_i$ of a mistake in categorizing that item. For a set of items with J classes, where $i \in \{1, 2, ..., J\}$ and p_i the fraction of items labeled with class i in the set, the Gini impurity can be calculated as:

$$\mathrm{I}_G(p) = \sum_{i=1}^{J} \left(p_i \sum_{k \neq i} p_k \right) = \sum_{i=1}^{J} p_i(1 - p_i) = \sum_{i=1}^{J} (p_i - {p_i}^2) = \sum_{i=1}^{J} p_i - \sum_{i=1}^{J} {p_i}^2 = 1 - \sum_{i=1}^{J} {p_i}^2 \tag{2.2}$$

I_G reaches its lowest value possible when all cases in one node fall into a single target category. When that metric is achieved, it is assumed the node has been able to isolate one or more features under a specific metric or metrics, and that new nodes can be created by extension as a consequence. However, what often happens is an entire isolation of samples usually is not practical or possible, and that some level of impurity is tolerated.

When the new nodes are developed, a new branch is created from a parent node, where the splitting process, as described above, is performed. With the proper splitting process, the expectation is that the child nodes are purer than the parent node [41]. This process can be repeated until there are no more splitting criteria possible and those last two child nodes are defined as leaf nodes.

An appealing characteristic of decision trees is the clear and direct ability to interpret them, especially when considering its construction of decision rules that guides any path from the root node onto any leaf [40]. Decision rules are constructed in the logical form *if antecedent, then consequent*, where the path through the tree, passing by the attribute values from branches is the antecedent, and the consequent consists of ”the classification value for the target variable given by the particular leaf node” [40]. Since it is important to deflect from memorizing a training set, the CART algorithm has to prune nodes and branches that reduces the capability of generalizing classification results. If that happens, the model is more complex, and its subset of results becomes smaller and less representative of the entire dataset, leading into overfitting and a high adjusted overall error [40].

2.1.2 Random Forest

A random forest can be defined as a ”combination of tree predictors such that each tree depends on the values of a random vector sampled independently, with the same distribution for all trees in the forest”, and each tree votes for the most popular class at a given input (figure 2.1) [38]. The procedure goes as follows: a random vector θ_k is generated, independent of the past random vectors $\theta_1, ..., \theta_{k-1}$ but with the same distribution; and a tree is grown using the training set and θ_k, resulting in a classifier. This procedure repeats several times until a considerable number of trees are created.

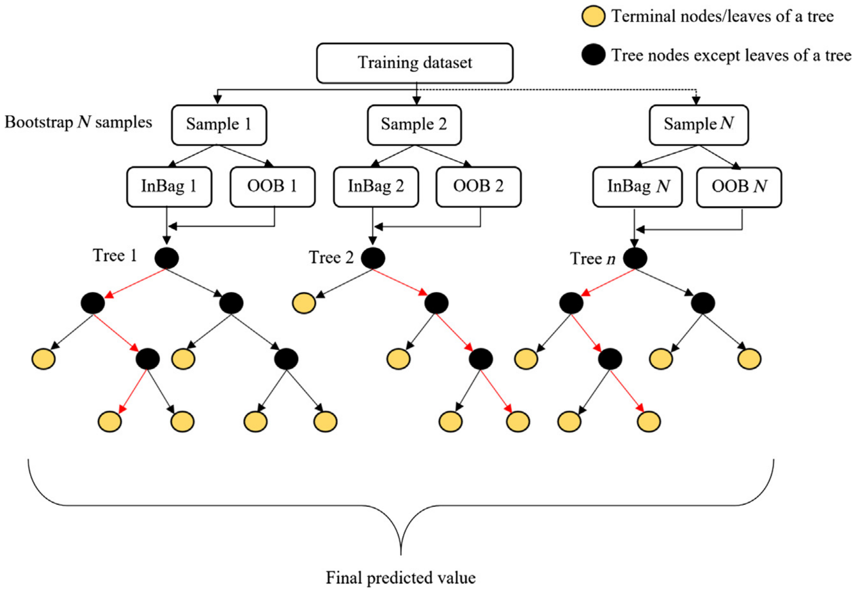


Figure 2.1: Structure of a random forest [41]

The number of trees that can be added to a random forest can increase without a limit, according to Breiman, and will not cause overfitting problems on the model. Instead, the generalization error of the forest will converge into a value as it gets immense, and that error depends on how strong the individual tree is and its correlation with other classifiers [38, 41]. The training algorithm for the random forest is bagging, also called bootstrap aggregation, which consists on creating, or replacing, subsets of training data through random samples, represented in figure 2.1 as ”Out of bag” (”OOB”) throughout the original data, and fitting new trees onto those samples [41]. Estimates from out-of-bag subsets of data will overestimate the error rate, but they will guarantee an unbiased estimation of the error and can estimate strength and correlation of the each tree, which helps understanding the overall accuracy and paths to improve it [38, 41]. To reduce as best as possible the generalization error, there needs to be an optimization of the number predictive variables, which is possible to achieve it through the analysis of each variable's relative importance, which can be determined by a sensitivity analysis of the model, by changing one of the variables and analyzing the change in accuracy thanks to the OOB error estimation [38, 41].

2.2 Artificial Neural Network

The artificial neural network is a widely used model, inspired by how the human brain processes and computes information in order to perform a specific task or function, because it is a radical different approach from conventional computers [42]. The human brain is capable of recognizing complex, nonlinear learning systems, through organizing and structuring interconnected series of neurons, and is capable of performing tasks such as classification, pattern recognition and perception with information gathered from the human sight, and knowledge is ”acquired from its environment through a learning process” and stored in synapses [40, 42]. Neural networks have several properties in addition to its nonlinearity: the model can be mapped in an input-output structure, which means it is possible to insert an input signal and receiving a response, which makes it suitable for supervised learning; the network can adapt itself as it receives more information by adjusting its synaptic weights, which are the links who connect the neurons [42].

2.2.1 Neuron

A neuron can be defined simply as a unit of information and processing, which is a fundamental part of the network. A neuron k is composed with three elements that are represented in figure 2.2:

1. Connecting links from an input signal x_j, which can be the output of another neuron or input from datasets, to the neuron k each of which containing a weight w_{kj} or their own.

2. A linear operation, a summation, for adding the input signals weighted by the respective synapses

3. An activation function $\varphi(\cdot)$ that limits the value of the neuron's output [42].

The model can also have a bias, b_k that influences the input of the activation function. Mathematically, a neuron k is described with a pair of equations

$$u_k = \sum_{j=0}^{m} w_{kj} x_j \tag{2.3}$$

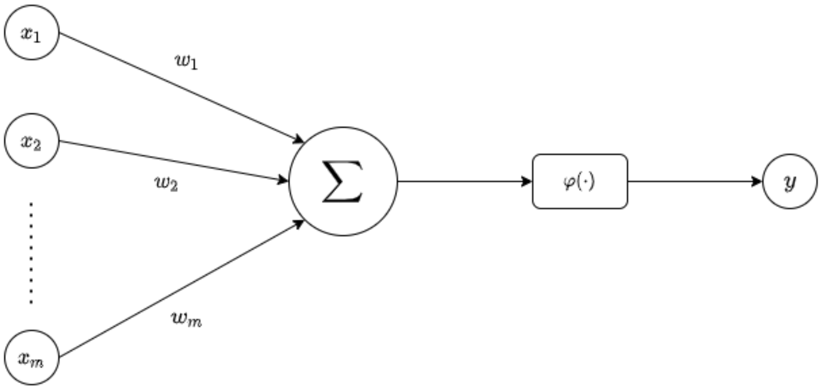


Figure 2.2: Mathematical nonlinear model of a neuron [42]

and

$$y_k = \varphi(u_k + b_k) \tag{2.4}$$

where u_k is the linear combiner output due to the input signals. The use of bias b_k has the effect of applying an affine transformation to the output u_k of the linear combiner is reflected as one of the neuron's synapses [42]:

$$b_k = w_{k0} \tag{2.5}$$

where the output is defined by the activation function of the neuron's induced local field:

$$y_k \approx \varphi(v_k) \tag{2.6}$$

There are several activation functions available, however only three will be used:

1. Rectified linear unit function. A specific type of a piecewise-linear function, that works as a ramp function, that can be described

$$\varphi = \begin{cases} 1 & \text{for} \quad v \geq 1 \\ v & \text{for} \quad 0 < v \leq 1 \\ 0 & \text{for} \quad v < 0 \end{cases} \tag{2.7}$$

   This is an approximation to a non-linear amplifier, where two situations may arise and create special forms in the function: the first one is that it evolves into a linear combiner if the operation does not saturate, computing in the linear region. The second form is, if the amplification is infinitely large, the function becomes a threshold function. It is the generic and mostly used function for activation of a neural network [43].

2. Sigmoid function. This is a strictly increasing function that enables a balance between linear and nonlinear behavior. The more common example of a sigmoid function is the logistic function:

$$\varphi(v) = \frac{1}{1 + exp(-av)} \tag{2.8}$$

where a is the slope parameter of the sigmoid function. This type of function enables a continuous range of values from 0 to 1, unlike the rectified linear unit function, ensuring the entire sigmoid function to remain differentiable throughout the interval and the range of output possible is only between 0 and 1, which narrows and normalizes the the input it receives, which is why it is often called a squashing function [40].

3. Hyperbolic tangent activation function. It is very similar to the sigmoid one, sharing the same shape of function. The difference is that it can take any real value as input and the interval between -1 and 1.

$$\varphi = \frac{e^{x}{-}e^{-}x}{e^{x}+e^{-}x} \tag{2.9}$$

where x is the input provided to the function. It used particularly as an alternative to sigmoid and rectified linear unit functions as an activation function for hidden layers.

2.2.2 Multilayer Perceptron

The perceptron is also known as a single layered perceptron, where it is only composed by an input layer that refers to input of data and an output layer, for the result of the model aplied. The difference between neuron and perceptron is that the generates an output capable of solving a classification problem, whereas the neuron only generates an output.

A multilayered perceptron (figure 2.3) is made of at least three layers: an input layer, one or more hidden layers, and an output layer. The input signal goes through the network in a forward direction (from input to output), on a layer-by-layer basis. The computational power of a multilayer perceptron is due to the fact that there are more than one layer of hidden neurons, with significant connectivity between them through their synapses, because any neuron from a specific layer is connected with all neurons from neighboring layers. That characteristic facilitates pattern recognition in the network, a crucial component in solving complex problems [42].

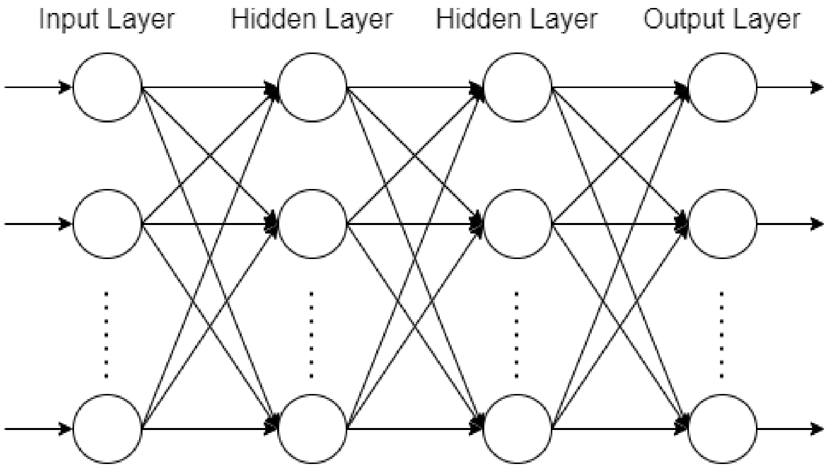


Figure 2.3: Architectural graph of a MLP

Every neuron located in the network performs two types of functions: the first one is a function signal that propagates forward from the input onto the output, which has had a previous and brief explanation.

The second is an error function, that begins in the output neurons, and propagates, layer by layer, in a backward course through the network [42]. So, respectively, the output and hidden neurons need to perform two computations: the function signal appearing at the output of a neuron, and an estimation of its gradient vector, which calculates the variation of the error with respect to the weights connected. This flow of computations, the function signal on one direction and the error on the opposite direction creates a feedforward nature on the network, because there is not a loop or cycling of those functions [40].

2.2.3 Backpropagation Algorithm

The backpropagation (BP) algorithm represents mathematically how a neural network learns as model of supervised learning, because, for every observation from the training set, an output value y is produced and compared to the actual value of the variable d [40]. For an iteration n at the output of neuron i, the error signal is defined by

$$e_i(n) = d_i(n) - y_i(n) \tag{2.10}$$

and, as is characteristic of networks such as MLP, the error function is a component of all neurons. Yet it has a different value for output neurons than hidden ones, since the error function has to be determined recursively [42].

Output Neuron

With an output neuron, it is possible to compute directly the error function as seen in equation 2.10, and from that the error energy function is easily determinable:

$$E_{avg} = \frac{1}{N}\sum_{n=1}^{N} E(n) \quad \text{where} \quad E(n) = \frac{1}{2}\sum_{i=1}^{O} e_i^2(n) \tag{2.11}$$

for all neurons in the output layer (size O), where N is the size of the data set. This can be considered as a cost function as a measure of learning performance, in which the objective is to minimize it [42]. With the local field function (eq. (2.3)) rewritten for error flow of the network:

$$v_i = \sum_{j=0}^{h} w_{ij} y_j \tag{2.12}$$

where y_j is the output of the hidden neurons that will serve as input to the output neuron i, that can be computed from equation (2.6).

$$y_i = \varphi(v_i) \tag{2.13}$$

It is possible to understand that minimizing the error energy function depends mainly on the synaptic weights, and for the BP algorithm, optimization of the energy function is achieved by making corrections to the synaptic weights Δw_{ij} that is proportional to the partial derivative for every weight $\partial E(n)/\partial w_{ij}$ by the use of chain rule.

$$\frac{\partial E(n)}{\partial w_{ij}(n)} = \frac{\partial E(n)}{\partial e_i(n)} \frac{\partial e_i(n)}{\partial y_i(n)} \frac{\partial y_i(n)}{\partial v_i(n)} \frac{\partial v_i(n)}{\partial w_{ij}(n)} \tag{2.14}$$

Differentiating equations (2.10) to (2.13) for each term respectively gives:

$$\frac{\partial E(n)}{\partial w_{ij}(n)} = -e_i \varphi_i'(v_i(n)) y_i(n) \tag{2.15}$$

where φ_i' is the derivative of the activation function. Thus, the correction made with the partial derivative is computed with delta rule:

$$\Delta w_{ij}(n) = -\eta \frac{\partial E(n)}{\partial w_{ij}(n)} \tag{2.16}$$

where η is the learning rate parameter of the algorithm, which is a constant. The minus signal provides the direction in which the weight change has to follow in order to reduce the error [42]. The learning rate gives an opportunity to adjust the correction of the weight space. The delta rule can be further simplified:

$$\Delta w_{ij} = \eta \delta_i y_i \tag{2.17}$$

where δ_i is the local gradient, a component that indicates to where changes need to happen in the synaptic weight, by estimating the responsibility for a particular error on neuron i [40, 42].

$$\delta_i(n) = \frac{\partial E(n)}{\partial v_i(n)} = -e_i \varphi_i'(v_i(n)) \tag{2.18}$$

To summarize, for an output neuron i the calculation of a correction to the weights is very simple with a multiplication between the error value, the derivative of the neuron's activation function and the output generated by the network.

Hidden Neuron

For the hidden neuron, there is no desirable response unlike the output neuron, which makes the error signal function difficult to determine, because it is not directly connected to the desired output. The BP algorithm functions by recursively calculating the error signal in terms of the functions that are connected to a hidden neuron i. So the local gradient, as determined in equation (2.18), has to be rewritten:

$$\delta_i(n) = \frac{\partial E(n)}{\partial y_i(n)} \frac{\partial y_i(n)}{\partial v_i(n)} = -\frac{\partial E(n)}{\partial y_i(n)} e_i \varphi_i'(v_i(n)) \tag{2.19}$$

Considering the energy error function as determined in equation (2.11), and considering the output neuron as k the partial derivative $\partial E(n)/\partial y_i(n)$:

$$\frac{\partial E(n)}{\partial y_i(n)} = \sum_{k \in O} e_k \frac{\partial e_k(n)}{\partial y_i(n)} \tag{2.20}$$

For the derivative of the error function regarding the hidden node's output, it needs to be decomposed again using the chain rule:

$$\frac{\partial e_k(n)}{\partial y_i(n)} = \frac{\partial e_k(n)}{\partial y_k(n)} \frac{\partial v_k(n)}{\partial y_i(n)} \tag{2.21}$$

From equation (2.14) and (2.10), the result is the same for $\partial e_k(n)/\partial y_k(n)$ since it is differentiating the output neuron's error function in terms of its output. Differentiating (2.12), and adapting its terms for the hidden neuron case leads to:

$$\frac{\partial v_k(n)}{\partial y_i(n)} = w_{ki}(n) \tag{2.22}$$

And so the local gradient for the hidden neuron can be fully computed, as equations (2.12), (2.19) to (2.21) result in

$$\delta_i(n) = \varphi_i'(n)(v_i(n)) \sum_{k \in O} -e_k \varphi_k'(n) w_{ki} \tag{2.23}$$

Equation (2.23) represents mathematically the BP algorithm, with figure 2.4 assisting the visualization of the computation flow, because the local gradient with hidden neurons depends on previous information given from the all connections with the hidden neuron. This justifies the conception of error function streaming from the output of because, as previously explained, the activation function φ is a nonlinear one. This means that a minimization of E_{avg} will not produced a solution using the least-squares regression, so the solution proposed is to use the optimization techniques such as the gradient descent method [40, 44].

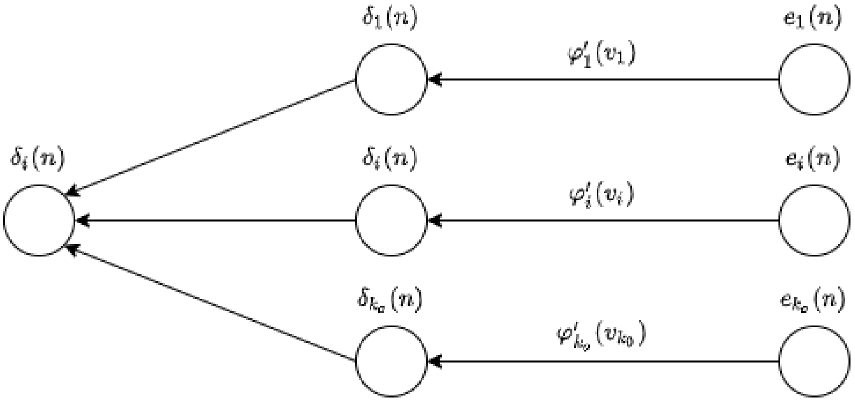


Figure 2.4: Diagram representation of the local gradient on a hidden neuron

2.2.4 Gradient Descent Method

In order to change arithmetically, there needs to be an adjustment to weight of synaptic over the entire training set, so the changes only happen after ”one complete presentation of the set”, or an epoch [42]. From equation (2.3) and (2.11), considering the input variable as constant during an epoch, the error function depends on the synaptic weight w_{ij}. For the neuron j, there is a vector of weights $w_j = w_{0j}, w_{1j}, ..., w_{Nj}$ with size $N+1$ (considering that a bias is applied). The gradient in the E_{avg} can

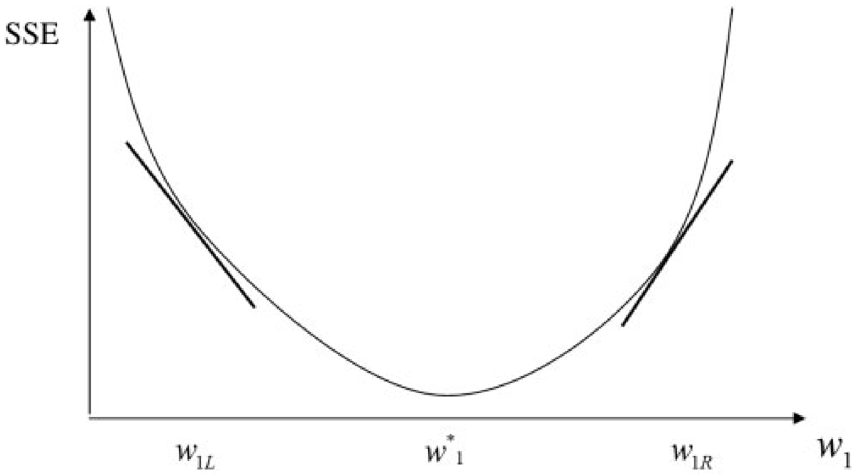


Figure 2.5: Plot of the error depending on the weights feeding a neuron [40]: it is also plotted the derivative that represents the sensitivity factor of the synaptic weight

be computed as:

$$\nabla E_{avg}(w_j) = \left[\frac{\partial E_{avg}}{\partial w_{0j}}, \frac{\partial E_{avg}}{\partial w_{1j}}, ..., \frac{\partial E_{avg}}{\partial w_{Nj}}\right] \tag{2.24}$$

So what happens is a recursive correction of the synaptic weights after an epoch:

$$w^*_{ij} = w_{ij} + \Delta w_{ij} \tag{2.25}$$

where w^*_{ij} is the corrected weight, proportional to the partial derivative $\partial E_(n)/\partial w_{ij}(n)$, which represents a sensitivity factor that determines the direction of search for the synaptic weight w_{ij} [42].

$$\frac{\partial E(n)}{\partial w_{ij}(n)} = -e_i(n)\varphi'_i(v_j(n)) \tag{2.26}$$

The average error function is a parabolic of the type $y = ax^2$ (figure 2.5). Considering that N is a natural number, it implies that $1/2N > 0$, and because there are two summing functions, the index will always be of the type $a > 0$, which means the curve will open upwards. So, in order to achieve the lowest possible error, if the partial derivative has positive slope, then the correction needs to be made leftwards, and the weight w_{ij} needs to be reduced, and $\Delta w_{ij}(n)$ needs to have a negative value. However, if the derivative has a negative slope, the correction goes rightwards, and $\Delta w_{ij}(n)$ is positive. So, according the delta rule determined in equation 2.16, the minus signal provides the direction for weight change that reduces the error, which is the gradient descent in weight space [42]. The learning rate gives an opportunity to adjust the correction of the weight space: if the slope is too steep, the adjustment necessary will be large [40].

Variations of Backpropagation algorithm

The learning rate of a neural network, setting the pace for a correction of the error through backpropagation of error, needs to find a balance between learning slowly for a smoother transition of change of the synaptic weights with the cost of several iterations needed for an accurate description of the neural network's behaviour. If the learning rate is higher, the pace of correction is higher which might lead to oscillatory and unstable behaviour of the synaptic weights [42]. However it is possible to increase the learning rate without running into the danger of instability, by introducing a momentum term α functioning as a feedback loop that takes into account past corrections

$$\Delta w_{ij}(n) = \Delta w_{ij} = \eta \delta_i y_i(n) + \alpha \Delta w_{ij}(n-1) \tag{2.27}$$

where the momentum varies in the range between 0 and 1

2.3 Active Learning Algorithm

Active Learning (AL) is a particular sub-environment of machine learning in which an algorithm can choose the data from which it will learn, therefore performing better with less training and data than a supervised learning algorithm [45]. In practice, from a small and labelled testing data, an AL system will add more data by asking queries from an oracle to label specific data. The goal is to achieve high accuracy from sparse labelled data, minimizing the expense of obtaining these type of data [45].

Depending on the problem scenario, there are three main frameworks from which a learner can ask queries:

- Membership Query Synthesis - The learner solicits labels for any unlabelled data point and a query is generated for the learner to evaluate it
- Stream-Based Selective Sampling - Assuming that obtaining unlabelled data is inexpensive, the input distribution may follow a stream-based approach, in which the learner decides from one data point onto another where to query it or discard it. That decision usually follows a query strategy which shapes the learning process.
- Pool-Based Sampling - The input has a small labelled data, and a larger pool of unlabelled data set is available. Queries are then drawn from the pool in a greedy way, by selecting the best data point from the entire pool based on a specific measure of queries. Whereas the Stream-Based approach examines the data sequentially and makes query decisions, the Pool-Based sampling evaluates the entire data in order to select a query [45].

There are several strategies for the AL algorithm to select a query, but two of them are the most used: the first one is the Uncertainty Sampling, a simple framework in which the learner queries data with the least certainty on how to label. For probabilistic models with multiple class labels, the sampling queries the least confident instance, while ignoring the rest of the label distribution; the second one is Margin Sampling, which tries to solve the shortcomings in Uncertainty Sampling for multi-class labelled data by considering the probability distribution of two least confident labels, with better results if the classifier considers smaller margins of probability between two labels because the model discriminates with better capability; the third one is the Entropy Sampling, a more general strategy, in which it tries to map the distribution of probabilities with the information given [45].

The second strategy for selecting a query is the Query-By-Committee. The committee is a group of models trained on a labelled set representing hypotheses of selection, where each member can vote on labellings of query candidates. The goal is to find the best model with the most precision. For the committee to be successful though, it is required to build a diverse set of models with disagreement among them [45].

2.4 Hyperparameter Tuning

Hyperparameter can be defined as a parameter whose value is set before the learning process. For example, in a MLP, the synaptic weight of a given node is a value derived from the learning process. These particular parameters are what defines the network:

- Number of hidden layers
- Number of hidden neurons
- Type of activation function (φ)
- Learning rate (η)
- Momentum (α)

For the Random Forest, the parameters are responsible for shaping the size of an individual tree, or the scope of a forest:

- The function to measure the quality of a node split in a tree
- Number of trees
- Depth of the tree
- Number of samples required to split an internal node or a leaf node
- If bootstrap samples will be used when building trees
- etc.

Hyperparameter tuning can be defined as an optimization problem with the intent of determining those parameters that lead into an optimal value. This pursuit can be computationally expensive and time consuming, specially if a brute force type of search is performed, where all possible data points are verified. An alternative could be a randomized search through several options, yet there are no guarantees of remotely achieving the optimum point. Therefore, this hyperparameter tuning problem can be solved through an algorithm designed for its optimization, for example the Bayesian optimization, where it has been tested for machine learning algorithms such as RF and NN with success [46].

The Bayesian optimization is a strategy for determining local maxima from computationally expensive functions, considering prior tested data. The maximum value, from a given search space, is determined by a combination of exploiting spaces with high values (exploitation) and exploring other areas with uncertainty (exploration), which are common in optimization algorithms. The prior distribution of Bayesian optimization is ensured with the Gaussian process(GP), which is considered flexible and easy to handle, therefore helping with a good fit of data in the algorithm [46]. The GP is a function where the variable is a Gaussian distribution:

$$f(x) \sim GP(m(x), k(x, x')) \tag{2.28}$$

$m(x)$ is the distribution's mean function and $k(x, x')$ the covariance function of two tested points x and x'. The function k is usually an exponential square function, and measures the degree of correlation between the two points. If there is a strong correlation, there is less uncertainty, however, if the points are further away, there is less correlation and more uncertainty. If the number of data points are large enough, it is possible to have a general sense of how to optimize function f [46].

Given a posterior information, the GP works in an iterative way and the acquisition function determines the next search. It can work as a exploitative function of known data, where the search goes until there is an upper limit. Or it could move towards greater variance, which leads to exploring the uncertainty of the function space [46].

Chapter 3

Data Analysis and Preprocessing

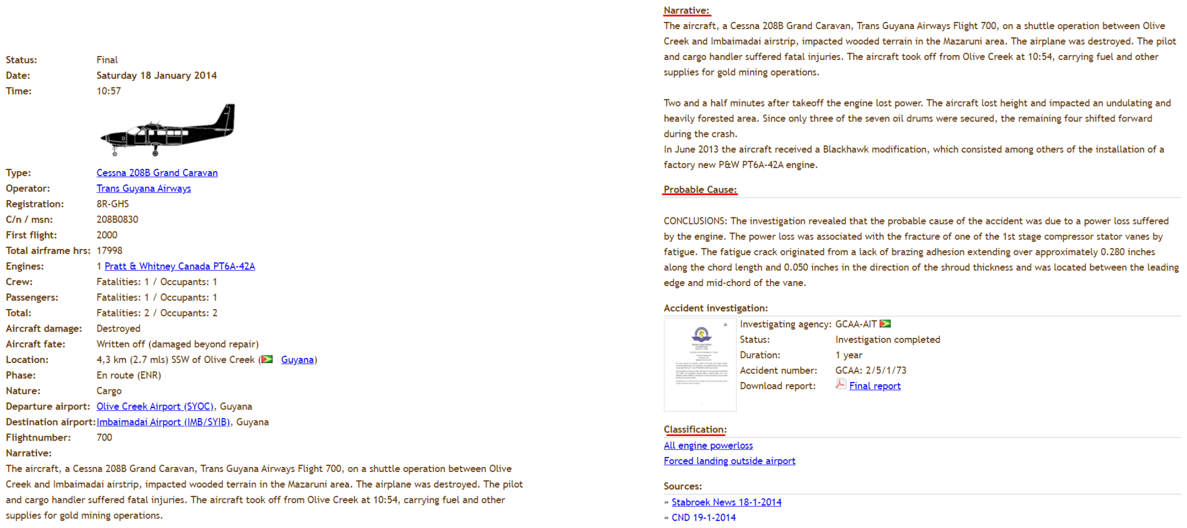
Taking into account that the ML learning methods detailed in chapter 2 are data-driven and the description of safety models in aviation industry in section 1.3, this chapter specifies how these models were designed. Section 3.1 explains how, from the Aviation Safety Network (ASN) and work performed regarding human factors with information from ASN, and also with more information about a specific occurrence, a database was built. Section 3.2 focuses on a particular set of information available and how a linkage with human factors was built, with help of the HFACS taxonomy. With a database built, section 3.3 delves into the construction of two models capable of implementing machine learning methods and finally, section 3.4 focuses on how those models can be evaluated.

3.1 Data Collection

The data used was provided by the Aviation Safety Network (ASN), which is ”a private, independent initiative created in 1996” [47]. Their goal is to ”provide everyone with a professional interest” with a current and dependable database that ”covers accidents and safety issues with regards to airliners, military transport planes and corporate jets”, with their information being gathered mostly from official sources such as local authorities and safety boards [47]. The database given by the ASN has 3242 data points, which are the occurrences suffered by a given aircraft between 2000 and 2020. Each data point (figure 3.1) (accident, incident or an other occurrence) has the narrative, causes, contributory factors precluding the occurrence and outcome on the aircraft produced by those failures.

In addition to the information provided in the database, there is also other components of particular relevance:

- information about the damage sustained by the aircraft;
- fate of the aircraft;
- phase of flight;
- the occurrence (incident, accident, other);



Status:	Final
Date:	Saturday 18 January 2014
Time:	10:57
Type:	Cessna 208B Grand Caravan
Operator:	Trans Guyana Airways
Registration:	8R-GHS
C/n / msn:	208B0830
First flight:	2000
Total airframe hrs:	17998
Engines:	1 Pratt & Whitney Canada PT6A-42A
Crew:	Fatalities: 1 / Occupants: 1
Passengers:	Fatalities: 1 / Occupants: 1
Total:	Fatalities: 2 / Occupants: 2
Aircraft damage:	Destroyed
Aircraft fate:	Written off (damaged beyond repair)
Location:	4,3 km (2.7 mls) SSW of Olive Creek (Guyana)
Phase:	En route (ENR)
Nature:	Cargo
Departure airport:	Olive Creek Airport (SYOC), Guyana
Destination airport:	Imbaimadai Airport (IMB/SYIB), Guyana
Flightnumber:	700

Narrative:

The aircraft, a Cessna 208B Grand Caravan, Trans Guyana Airways Flight 700, on a shuttle operation between Olive Creek and Imbaimadai airstrip, impacted wooded terrain in the Mazaruni area. The airplane was destroyed. The pilot and cargo handler suffered fatal injuries. The aircraft took off from Olive Creek at 10:54, carrying fuel and other supplies for gold mining operations.

(a) Front Page

Narrative:

The aircraft, a Cessna 208B Grand Caravan, Trans Guyana Airways Flight 700, on a shuttle operation between Olive Creek and Imbaimadai airstrip, impacted wooded terrain in the Mazaruni area. The airplane was destroyed. The pilot and cargo handler suffered fatal injuries. The aircraft took off from Olive Creek at 10:54, carrying fuel and other supplies for gold mining operations.

Two and a half minutes after takeoff the engine lost power. The aircraft lost height and impacted an undulating and heavily forested area. Since only three of the seven oil drums were secured, the remaining four shifted forward during the crash.
In June 2013 the aircraft received a Blackhawk modification, which consisted among others of the installation of a factory new P&W PT6A-42A engine.

Probable Cause:

CONCLUSIONS: The investigation revealed that the probable cause of the accident was due to a power loss suffered by the engine. The power loss was associated with the fracture of one of the 1st stage compressor stator vanes by fatigue. The fatigue crack originated from a lack of brazing adhesion extending over approximately 0.280 inches along the chord length and 0.050 inches in the direction of the shroud thickness and was located between the leading edge and mid-chord of the vane.

Accident investigation:

Investigating agency:	GCAA-AIT
Status:	Investigation completed
Duration:	1 year
Accident number:	GCAA: 2/5/1/73
Download report:	Final report

Classification:

All engine powerloss
Forced landing outside airport

Sources:

» Stabroek News 18-1-2014
» CND 19-1-2014

(b) Back Page

Figure 3.1: Example of an accident on January 18th 2014. In (a) information about the aircraft, flight and personnel are withdrawn. In (b) the underlined topics (Narrative, Probable Cause, Classification) were already withdrawn from ASN.

- the number of passengers and crew;
- if it was a fatal occurrence;
- if so, how many passengers and/or flight crew were killed.

To extract that information, it is required to search in each data point specifically in the area displayed in figure 3.1(b). This research required building an independent database, which is a time consuming process, so a 10 year gap was selected, specifically between 2007 and 2017, totalling 1105 occurrences gathered from the ASN database.

In order to maximize the information gathered by this independent database and the ASN database, it is crucial the data points were the same. They have a specific date associated with them which might seem as an unequivocal method for equivalence. However, either there were days where more than one accident occurred, or there were collisions between aircrafts which gives two data points. Therefore, for the first scenario, the narrative associated with it will provide an unequivocal matching, for the second scenario, the indexes associated with the databases are ordered to associate correctly the aircrafts who were involved.

ID	Events.text	Events_1.text	Events_2.text	Events_3.text
1				Result - Runway mishap
2				Result - Runway excursion
3			Weather - Icing	Result - Runway mishap
4		Airplane - Undercarriage - Landing gear collapse		Result - Damaged on the ground
5	Airplane - Undercarriage - Tire failure	Landing/takeoff - Takeoff - Aborted		Result - Runway excursion

Table 3.1: Five examples of data points of the ASN database

ID	Occurence	Fatality	CrewFatality	PassengerFatality	TotalCrew	TotalPassenger	Damage	Fate	Phase
1	Accident	No	0	0	2	1	Repairable damage	Repaired	Landing
2	Accident	No	0	0	6	0	Hull-loss	Damaged beyond repair	Landing
3	Accident	No	0	0	1	0	Repairable damage	Repaired	Landing
4	Other occurence	No	0	0	2	0	Hull-loss	Damaged beyond repair	Standing
5	Accident	No	0	0	2	5	Repairable damage	Repaired	Takeoff

Table 3.2: Five examples of data points of the independent database

3.2 Database Labeling

In order to proceed with the analysis proposed and to factor the human factors in aviation safety, the taxonomy HFACS described in section 1.3.2 needs to be utilized. Considering the information withdrawn from the database, which are the columns ”Events.text”-”Events_2.text” as seen in table 3.1, a human factors analysis can be performed from the contributory factors prior to the accident. It is essential to keep tabs of all possible contributory factors and relate them with either underlying conditions or probable causes that triggered or may trigger those events [19]. It is possible to have an elementary understanding of these underlying conditions using a library available online called SKYbrary which is ”an electronic repository of safety knowledge related to flight operations, air traffic management (ATM) and aviation safety”, helping regular users to connect with data previously available on websites of multiple aviation organizations [48]. From the same library, they also provide with a simple list of definitions for all orders of the HFACS taxonomy and all levels, referencing works made by Shappell and Wiegmann [23] and Reason [19] [20, 47].

From the analysis of each contributory factor, it was possible to notice different ways on how the human factors were applied. The majority of them were categorized simply by using the HFACS taxonomy and the underlying factors. For others, either part of the probable causes, or the contributory factors are maintenance failures. So the taxonomy to be adopted in those circumstances is the maintenance extension of HFACS (HFACS-ME) that was proposed by Schmidt et al. [29]. However, from the article, it was not possible to find theoretical definitions that provide a holistic view. Instead for each level and order there were practical examples presented, which makes it difficult to a more general and robust analysis for other cases [29]. Finally, there were also vague categories for which a review from the SKYbrary did not lead to a satisfying conclusion, and redundant ones part of other categories. All such cases were disregarded. The analysis of the entire categorization is available in table A.1 in Appendix A. With this analysis, the database has to be extended because, for every column attributed to contributory cause to an accident, three columns with the corresponding human factor will have to be generated. Its distribution throughout the database is presented in figure 3.2.

The labels on the x-axis are encoded for the analysis, and they are assigned in alphabetical order. Table A.2 in appendix A represents the distribution with the encoded label associated with it displayed in figure 3.2 and it is possible to note the frequency of labels such as ”Physical Environment” or ”Personal Readiness” over certain such as labels ”Lighting” and ”Infraction”. This can be attributed to the fact that there are certain contributory causes that tend to take place more often than others, that there are human factors labels in which they fit with more contributory causes.

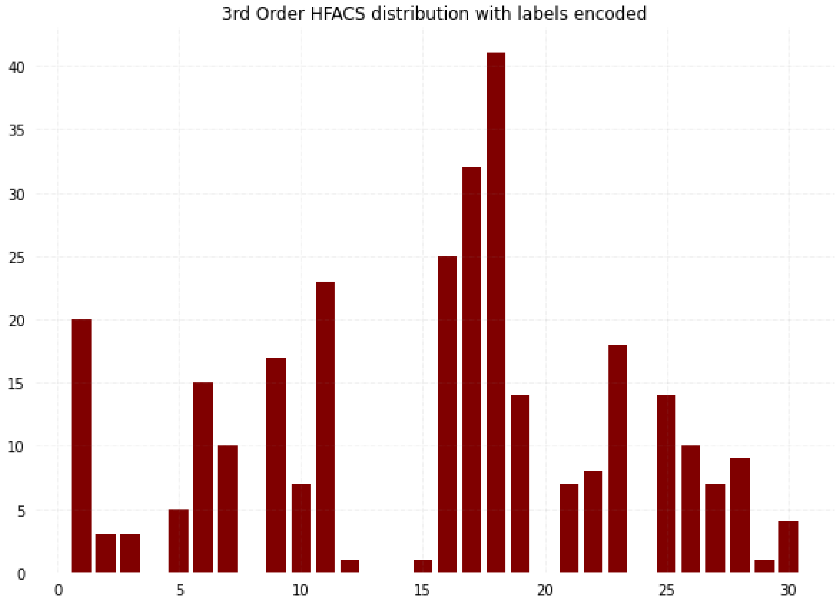


Figure 3.2: Distribution of human factors using the HFACS taxonomy and HFACS-ME extension

For the rest of the database, information which it was previously labelled such as damaged sustained by the aircraft, the phase of flight or what happened to the aircraft for the occurrence to be registered, an encoding process was developed suited to their respective column. For the entire encoding process the scikit-learn library for Python was used for that effect [39].

3.3 Database Modelling

Taking into account the information gathered for the conception of a database, it is clear that several inputs gathered do not have a linear linkage and so there is a need to infer relations between them. Not only that but the creation of models stem from hypothesis that were made with the intent of analyzing patterns that induce risk with aircraft procedures, so that safety standards can improve [9, 49]. Therefore, taking into account the aim of the modelling process, only certain learning processes can be considered.

Unsupervised learning can be referred as an ability to learn from unlabelled data without any sort of direction by having error signal to evaluate potential solution [50]. This type of approach can lead in the case of aviation safety to learn labels and explore patterns, particularly regarding human factors displayed on report of occurrences [9], however there are limitations on what type of acquisition patterns that unsupervised learning methods can produce [49], and the modelling process proposed requires to deepen the classification patterns and that is best suited for supervised learning methods.

Considering the objectives describe above, it was possible to create two models for analysis considering their features presented in tables 3.1 and 3.2, as illustrated by figure 3.3. They try to answer two questions:

1. Is it possible to predict whether an incident or accident produced any fatality? (Model 1)

2. If an occurrence was fatal, is it possible to estimate the percentage of people killed? (Model 2)

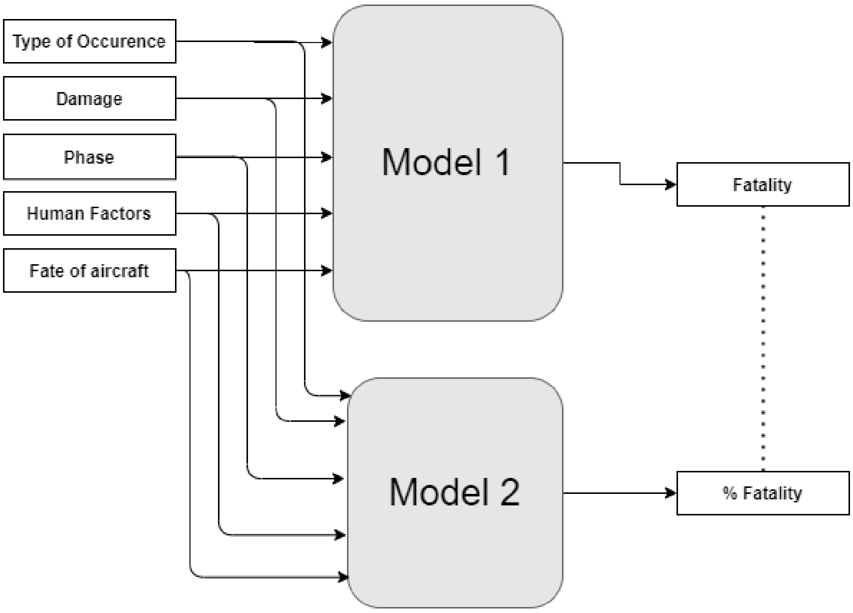


Figure 3.3: Schema of the database modelling

In order to properly study both models, supervised learning algorithms are well suited considering the existing database, with two algorithms being used as described in 2.1 and 2.2. The random forest algorithm has the capability to function as a ”white box”, where it is possible to analyze its performance of generating a variety of trees while maintaining a predictable outcome through bootstrap aggregation (bagging) , the importance of its features, which can lead into an analysis for feature selection. However, the algorithm is not robust enough due to the variety of trees, and the performance metrics may not be fully optimized.

The neural network has a ”black box” method, where the inputs are given and only the outputs and errors of the network's function are produced. Yet the BP algorithm and the shifts of synaptic weights are not disclosed, which make it difficult to trace its mechanisms. Through its training, the performance measures of the NN can be optimized by minimizing the loss function of the BP algorithm, or to maximize a specific performance criteria, such as accuracy for example, or there is the possibility of tracking the progression of both functions. However, it is difficult to understand which features are more important than others, and possible overfitting of the neural networks can be adjusted by changing its architecture rather than its features, or underfitting problems were the ML algorithm cannot learn properly with the training data provided.

Recently though, attempts to build interpretations of the ”black box” of neural networks are being made, specially when considering architectures that require more layers than MLP such as deep neural network [51, 52]. Since interpretation is defined as the mapping of an abstract concept into a domain that the human can make sense of, from a neural network perspective, the goal is to build mechanisms that help bring some understanding of the more deep layered neurons. For example, either by building prototypes of the input domain for the concepts learned by a model [52], or by combining with other logical structures and multi-criteria decision with the input and hidden layers. Nevertheless, this is a field of study that is centered in deep learning, where a more considerable amount data is required than the dataset for Model 1, and therefore it does not seem to be a relevant feature for the analysis that is proposed.

Instead of attempting to find the best algorithm for these models in terms of performance by comparison, the goal here is to provide a more holistic view of the models, because the one algorithm's advantages will complement the disadvantages of the other and vice-versa. For the implementation of both models, the parameters selected for the structure of the MLP were as follow:

- One hidden layer
- Adam algorithm as its optimizer
- Learning rate = 0.001
- Epochs = 100
- Batch size = 16
- Use of cross-validators that splits data in train/test

And for the RF algorithm:

- Number of trees = 1000
- Minimum number of samples required to split an internal node = 2
- Minimum number of samples required to be at a leaf node = 1
- Features being considered to find the best split: ”auto”, ”sqrt”, ”log2”
- The use of bootstrap samples when building trees

3.4 Performance Criteria

Classification models requires certain parameters to evaluate its validity, and a classifier is only valid if it can predict correctly a label when information is provided. There can be multiple labels from either several outputs or a single one, or there can be only two labels, which are considered as a binary classification problem. For the latter example, the prediction made will belong to one element in a set $\{0, 1\}$ where 0 indicates the negative class, and 1 the positive class. If the observation made classifies an instance as negative correctly, it is counted as a true negative (TN) observation. Likewise, if the observation is correctly predicted as a positive, it is counted as a true positive (TP). However, if an observation that belongs to the positive class and was predicted in the negative class, it is counted as a false negative (FN). In vice-versa, if the observation belongs to the negative class and was considered in the positive class, it is counted as a false negative (FN). These options can be placed in a matrix such as table 3.3, which is defined as a confusion matrix. A good performance measure that judges correct

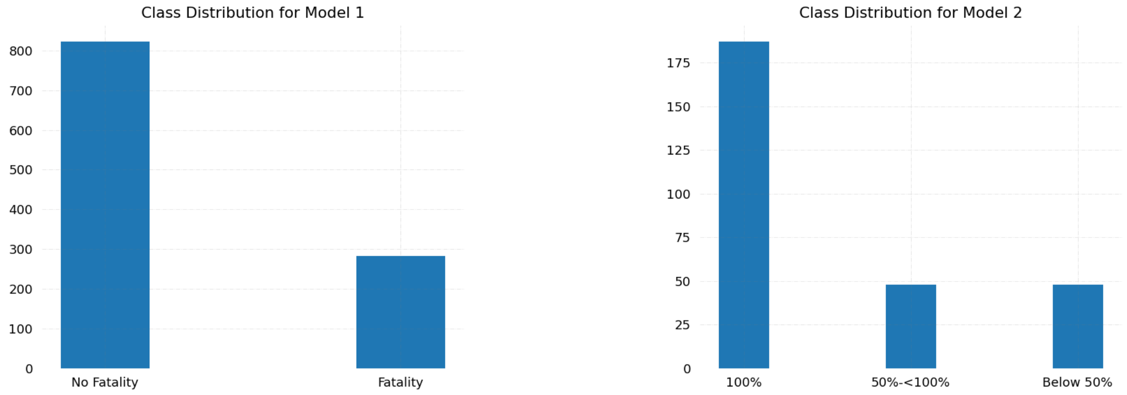
	True Class		
Predicted Class		0	1
	0	True Negative	False Negative
	1	False Positive	True Positive

Table 3.3: Confusion matrix for binary classification

labelling is accuracy, which can be defined as the percentage of correct predictions compared to the overall predictions made.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.1}$$

From this matrix there are several indicators that can be determined that illustrate the performance of the classifier. Considering the data set, there is a ratio of 3:1 in accident fatality (figure 3.4), meaning that for every accident which produced at least a fatality, there were 3 that did not. This means that the database is unbalanced and has an inherent biased towards the negative class. Therefore measures such as accuracy, which is defined as the ratio between the number of correct predictions made and the total number of predictions made, can be misleading and other measures might be more relevant [44].



(a) Model 1

(b) Model 2

Figure 3.4: Class distribution of the outputs of Models 1 and 2

Precision, recall and accuracy are measurements that can be defined as:

$$Precision = \frac{TP}{TP + FP} \quad \text{and} \quad Recall = \frac{TP}{TP + FN} \tag{3.2}$$

Precision can be characterized as the classifier's exactness, since it represents class agreement of the data labels, and recall is the ”effectiveness of a classifier to identify positive labels” [44]. With these parameters, it is possible to plot the precision-recall curve, which can give an indication of how good the classifier can predict positive values.

Another aspect to keep in mind for the binary classification for Model 1 is the fact that a misclassification of the minority class, which means failing to identify a fatal accident, is much costlier that a false alarm, or a misclassification of the majority class [53]. So it is important to consider other performance measurements, such as F-measure and for visual aid, plotting a precision-recall curve and the ROC curve

$$F_\beta = (1+\beta^2)\frac{precision \cdot recall}{(\beta^2 \cdot precision) + recall} \tag{3.3}$$

where β is a positive real factor. β can be translated as the amount of times recall is more important than precision. If $\beta = 1$ then both criteria achieve a harmonic mean and the F-score is deemed balanced.

This type of performance analysis that can be displayed with a confusion matrix can be extended to a multi-class analysis. With that there will be a confusion matrix with size $i \times i$ individual class C_i, where an assessment of performances can be made as TP_i,FP_i,TN_i,FN_i,$Accuracy_i$,$Precision_i$,$Recall_i$ [44]. It is possible to estimate the overall performance of multi-class models taking into account the distribution of those labels, by computing performances on average. Those averages can weighted, meaning the performance is computed for each label, and their average is weighted by the number of true instances for each label; they can also be macro averaged, which means the overall performance is calculated by averaging the performance of each individual label. The weighted form of averaging, on certain performance criteria such as accuracy, may not adequately reflect the quality of the criteria if there are severe class imbalances, because accurately predict a class that is overwhelmingly represented is something to be expected, and the high performance score is misleading.

Other performance criteria that deal with the class imbalances are the Receiver Operating Characteristic (ROC) and Precision-Recall (PR) graphs [53]. The ROC curve is defined as a plot of the False Positive Rate (FPR) on the x-axis and the True Positive Rate (TPR) on the y-axis.

$$TPR = \frac{TP}{TP+FN} \quad \text{and} \quad FPR = \frac{FN}{TP+FN} \tag{3.4}$$

From a binary classifier system, ”the discrimination threshold is varied from the most restrictive to the most lenient”, and from every possible threshold iteration, TPR and FPR are computed and a ROC value can be determined [53]. In a similar way, the Precision-Recall curve, which is defined with Recall as x-axis and Precision as y-axis, has the same principles in terms of variance in threshold and are computed similarly. Both curves can be summarized by an index called Area Under Curve (AUC). For the ROC, the AUC can be interpreted as the probability that the scores given will rate a random positive instance higher than a random negative one; for the Precision-Recall the AUC does not have an interpretation in terms of probabilities but instead its variance is related with ”the prevalence of the positive class” and ”the proportion of positive instances in the test set” [53]. With the respective curve and AUC, both metrics are suited for models with class imbalances, however, with a low sample combined with class rarity, Precision-Recall curves seem to be more capable than the ROC [53].

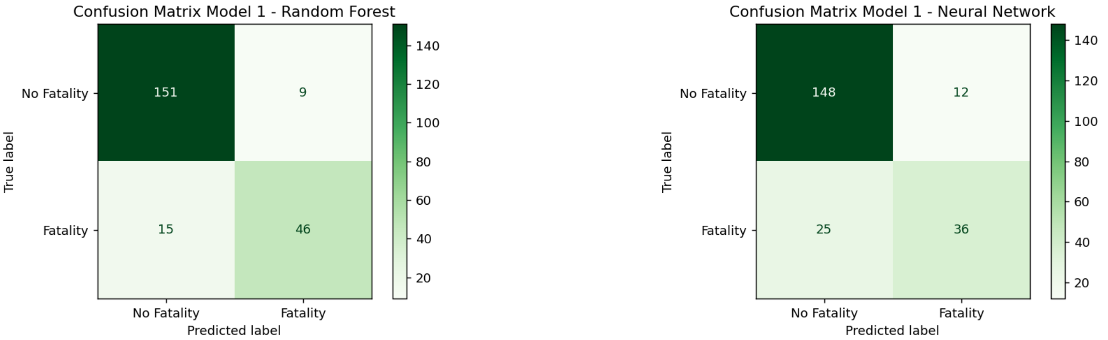
Chapter 4

Results

In this chapter, the results will be presented concerning the models being studied as described in section 3.3, and the evaluation is achieved through the performance criteria described in section 3.4. The first two sections, 4.1 and 4.2, evaluate the performance of Model 1 and Model 2 respectively, and section 4.3 deals with the optimization of both models performance with the hyperparameter tuning functions described in section 2.4.

4.1 Model 1

For model 1, a single-class binary output type of model was considered. To apply the supervised learning algorithms, 75% of the data set was used for training, and 25% for testing to validate them. To apply the RF algorithm the scikit-learn tool library for Python was used, and for the NN algorithm, the keras library was chosen to build the MLP. The information about the number of fatalities was dropped from consideration, and only the phase of flight, what happened to the aircraft, its damage, and the human factors that contributed were considered. After both algorithms are trained, a confusion matrix with the validation set is obtained for the respective algorithm implementation.



(a) Random Forest

(b) Neural Network

Figure 4.1: Confusion matrix for Model 1 with RF algorithm (a) and MLP algorithm(b)

The validity of the model can be confirmed by the MLP's binary cross entropy function, which is

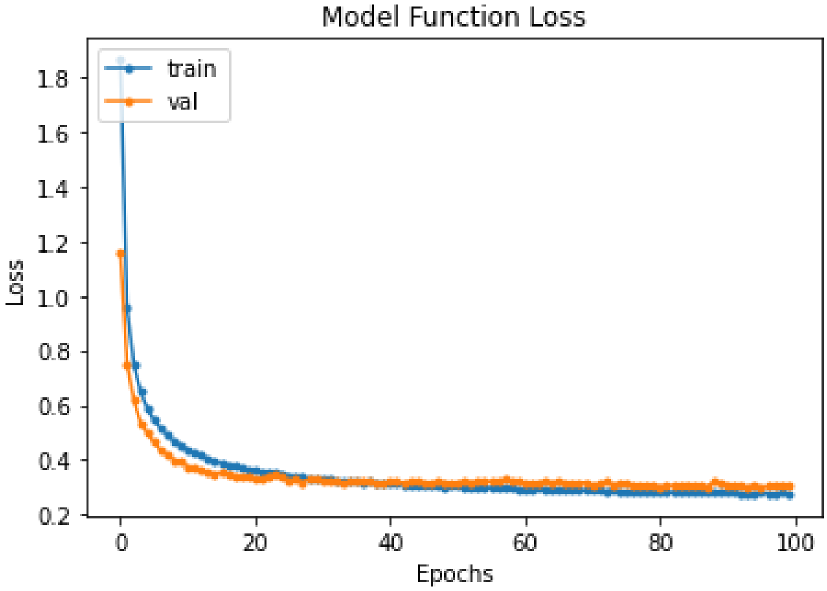


Figure 4.2: MLP function loss for Model 1

computed between true and predicted labels. Its behavior throughout the epochs can help determine the fit of data set, which helps shaping its structure (figure 4.2). Both training and validation curves have an exponential decay over the first epochs, which means the MLP finds quickly a good fit and does not seem to overfit, as the validation cross entropy curve does not seem to increase and the training curve decreases. From figure 4.1 it is possible to observe that both algorithms are similarly capable of detecting the predominant class (No Fatality). However, not only the RF algorithm is more capable of correctly identify the Fatality class than the MLP one, but doing so without predicting fatal occurrences as non-fatal. That key difference is further exemplified with table 4.1 From figure 4.1, considering the

Type of Learning			Precision	Recall	F1-score	Accuracy
Random Forest	Class	No Fatality	0.92	0.94	0.93	0.90
		Fatality	0.84	0.77	0.80	
	Averages	Macro	0.88	0.86	0.87	
		Weighted	0.89	0.90	0.89	
Multilayer Perceptron	Class	No Fatality	0.89	0.92	0.90	0.83
		Fatality	0.75	0.59	0.66	
	Averages	Macro	0.80	0.76	0.77	
		Weighted	0.83	0.83	0.83	

Table 4.1: Classification report for Model 1

class ”Fatality”, it is possible to determine that the False Positive (FP) for both algorithms are similar, meaning they both predict wrongly a non-fatal occurrence as fatal. However the difference is in False Negative, where a fatal occurrence is predicted as non-fatal. From equation 3.2, taking into account that FP will be similar for both algorithm, the difference is an increase of FN with the decrease of TP, which the Recall parameter. This is particularly important since it is more detrimental a FN than a FP, therefore the RF algorithm more suited than the MLP for this model.

In order to judge simultaneously the classifier and the dataset, a cross-validation was used so that all data could be used as testing data, while maintaining the same distribution of classes due to data imbalances. This was possible by using a Stratified K-Fold cross-validation (figure 4.3), more specifically a 5-fold validation, because it was important to maintain the 80/20 split between testing and training data of the model. However, the data itself used for testing and training will change in order to avoid certain biases of data that may appear on one split of data but may not in another split.

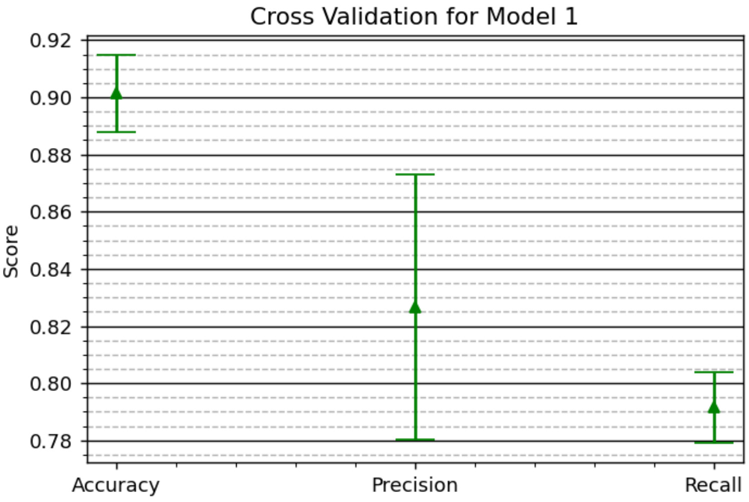


Figure 4.3: 5-Fold Cross Validation results for the RF algorithm

By comparing the standard deviation of both precision and recall with the accuracy, it is possible to conclude the higher sensitivity of metrics such as precision and recall is due to a lower quantity of positive labels. The mean of the parameters is higher for precision than recall which means that the classifier will evaluate, on average, more False negatives (FN) than False Positives (FP), meaning there will be more accidents that will be wrongly predicted as non-fatal, which is expected to the imbalances between the negative and positive class.

For both models, it was used the semi-supervised learning algorithm Active Learning (AL). For its application, the modAL library was used, which is an extension to the scikit-learn library, with the intent of simplifying the user's implementation of AL [39, 54]. Taking into account the data set, the pool-based sampling was chosen for the algorithm. Despite the existence of class imbalance in the data set, the minority class was deemed represented enough to select the uncertainty query strategy for AL algorithm, instead of more exploratory strategies. With this strategy, a total of 50 queries were made (figure 4.4)

It is notable the learning process accelerate after the 10th query for recall and f1-score and 30th query for precision, however after 50 queries, accuracy is 90%, precision 85% and recall 75% approximately. This is not much of a difference between approaches such as neural networks and it performes worse than random forest. So this sort of semi-supervised learning for this model, and with the amount of data, does not bring more of a new perspective to the model and with a lower score on an important metric such as recall, AL does not seem to suit this model well.

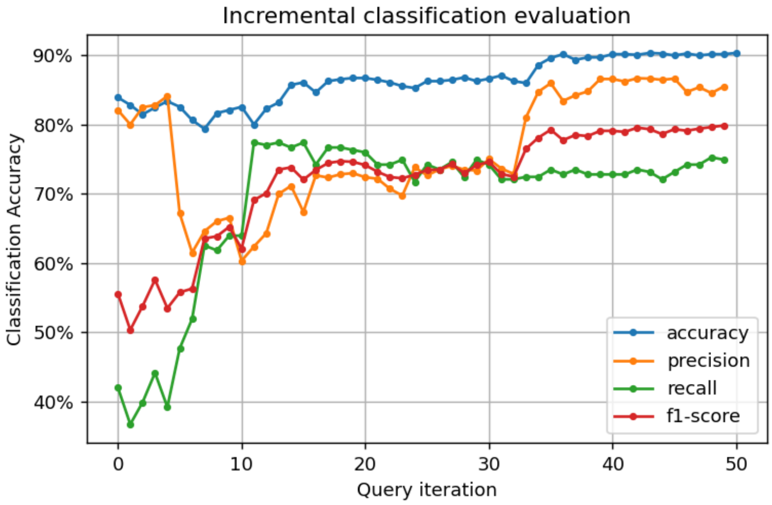
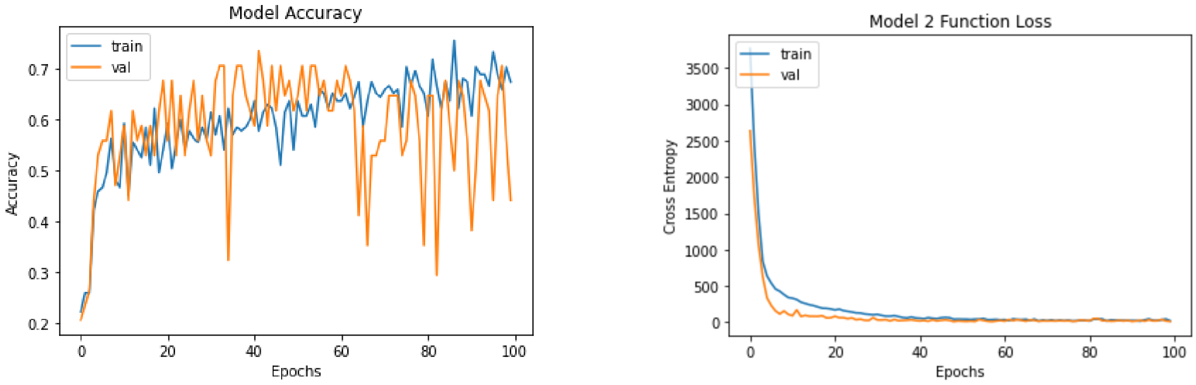


Figure 4.4: Performance criteria results as querying takes place for Active Learning for Model 1

4.2 Model 2

For Model 2, there is less quantity of data since only the accidents that produced fatalities are considered, which means for an algorithm such as MLP, there are limitations with quantity of data that error function for first epochs is very high (figure 4.5(b)). Despite the good shape of the loss function, the loss decay of the training data set is severe, which makes it difficult to judge the behavior of both curves as the epochs progress, regarding overfit or underfit of the model. With the accuracy function, there can be more understanding on the evolution of MLP with more training (figure 4.5(a))

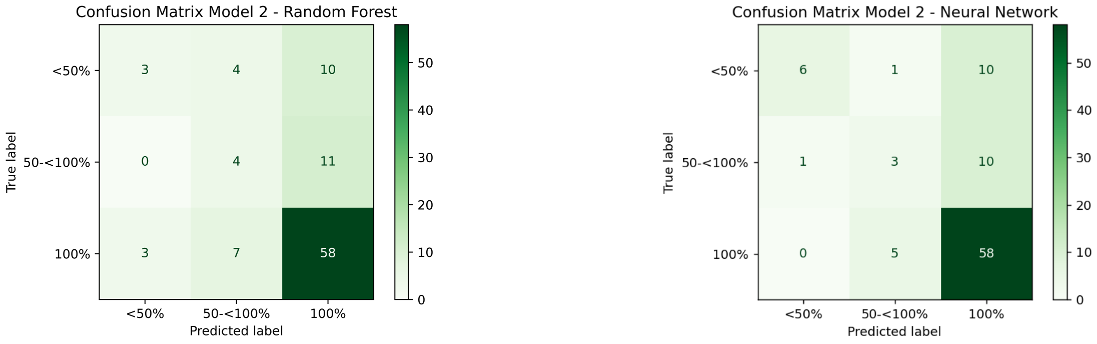


(a) Accuracy

(b) Loss

Figure 4.5: Accuracy (a) and function loss (b) for Model 2 with MLP algorithm

The change in accuracy of the validation split of data is sharp with every epoch, which can indicate either an underfitted model or a small sample of data, however the training curve is consistent with a regular accuracy curve, so it was considered the data structure to be well fitted. The model was also applied using the RF algorithm and the confusion matrices were obtained. The algorithms work in different approaches: for the RF algorithm, the predictions made favor the dominant class (100% Fatality), whereas the MLP diversifies its predictions, with many wrong ones located in the row of the



(a) Random Forest

(b) Neural Network

Figure 4.6: Confusion Matrix for Model 2 with RF algorithm (a) and MLP algorithm(b)

dominant class. Due to the scarcity of data, there will be significant variations of value in performance criteria, particularly in classes ”Below 50%” and ”50%-<100%”. If it is also considered several false predictions of those classes and the precision-recall and ROC curves will have poor shape.

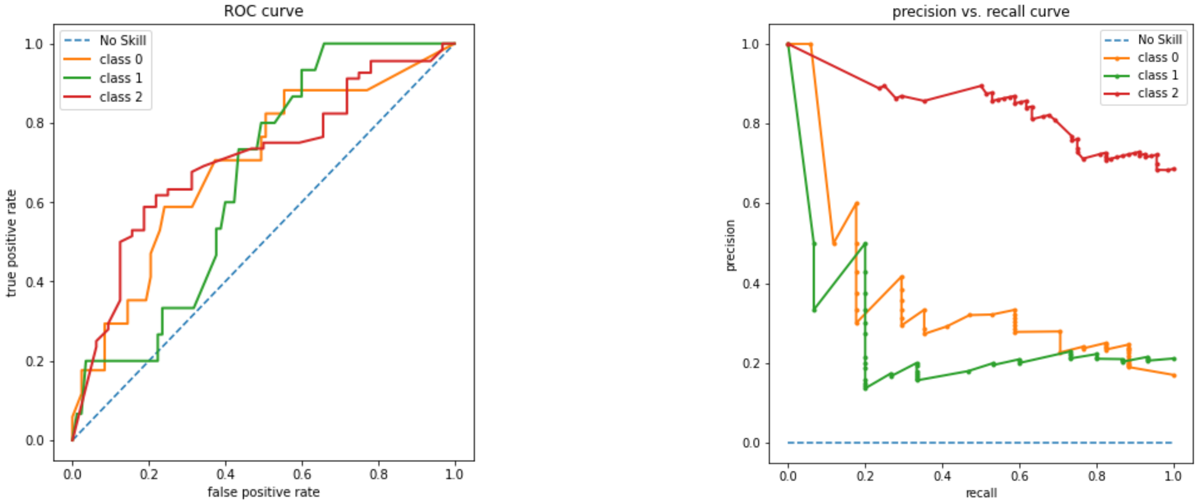
Type of Learning			Precision	Recall	F1-score
Multilayer Perceptron	Class	Below 50%	0.32	0.63	0.42
		50%-<100%	0.25	0.32	0.28
		100%	0.75	0.51	0.61
	Averages	Macro	0.44	0.49	0.44
		Weighted	0.59	0.50	0.52
Random Forest	Class	Below 50%	0.43	0.18	0.25
		50%-<100%	0.23	0.21	0.22
		100%	0.70	0.83	0.76
	Averages	Macro	0.45	0.41	0.41
		Weighted	0.58	0.62	0.59

Table 4.2: Classification report for Model 2

For Model 2 it was also used the semi-supervised learning algorithm Active Learning (AL), with the same sampling strategy. The query selection was the entropy one, as described in 2.3 because of the labels skewed distribution towards one class. The labelled data selected was 1% of the total data available, with the rest being used as pooling data. After each query, the algorithm was tested using the dataset (figure 4.8). Because of the overall number of data points available and the fact that a entropy sampling is used to select the queries, it is expected a decrease of accuracy compared to Model 1 and also some outlier performance results that produced either a sharp increase or decrease of percentage. Nevertheless, the overall precision results continues to be higher than recall and all performance measures continue to rise, which means the algorithm is capable of training adequately with the pool data.

After the last query, the confusion matrix can be computed. With the constant retraining of data where new testing data is added for the AL algorithm, the prediction can be improved even though the overwhelming amount of data predicted is towards the predominant class (figure 4.9).

Despite having a entropy approach of selecting queries, there are more predictions made for the



(a) ROC Curve

(b) Precision-Recall

Figure 4.7: Multi-class ROC curve for Model 2 (a) and Precision-Recall curve algorithm (b)

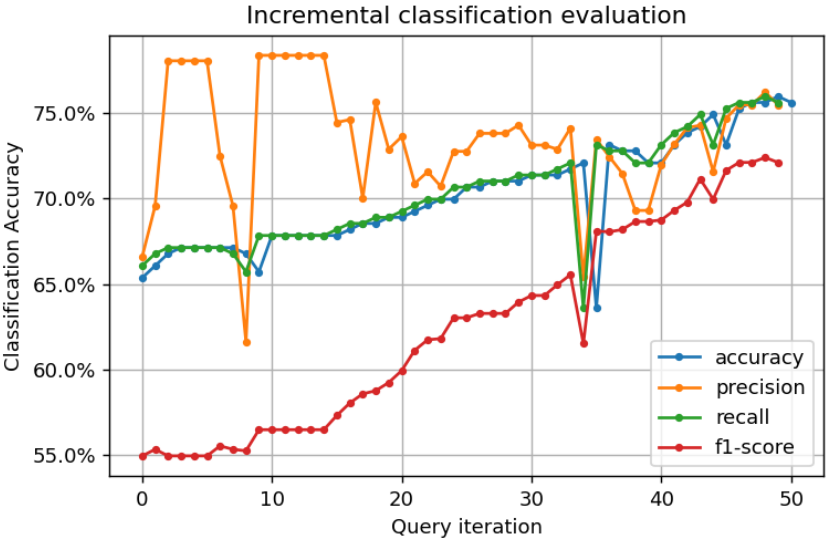


Figure 4.8: Performance of Active Learning with pool-based sampling for Model 2

lower classes ”Below 50%” and ”50%-< 100%” with the RF algorithm than the AL one, so even with the entropy sampling for querying selected, the RF algorithm still has a better exploration capacity than AL. Notwithstanding, the AL is capable to perform better than the RF when it comes to predicted correctly labels, as seen with table 4.3 where AL has better performance across all criteria, meaning it can predicted correctly more often the non-prevailing labels.

4.3 Model Optimization

For both models, a hyperparameter tuning of the algorithms parameters was conducted for optimization purposes. The data was divided into 20% for testing, 10% as validation data and 70% for training. The artificial NN is structured with three or four layers (counting both input and output layer). The op-

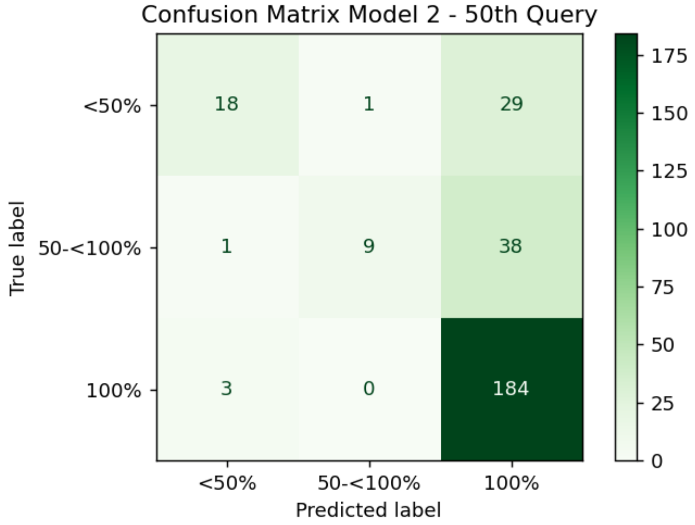


Figure 4.9: Confusion matrix of Model 2 after 50 queries performed

Type of Learning	**Classes**	Precision	Recall	Macro f1-score
Random Forest	Below 50%	0.43	0.18	0.41
	50% -<50%	0.21	0.21	
Active Learning	Below 50%	0.75	0.38	0.72
	50% -<50%	0.75	0.31	

Table 4.3: Comparison between RF and AL for predicting sparse labels for Model 2

timization of its structure for an MLP is achieved using the keras-tuner version 1.0.1. Then, a grid of parameters was selected for the MLP:

- Learning Rate: 0.1, 0.01, 0.001, 0.0001, 0.00001
- Activation Function: relu, tanh, sigmoid;
- Number of neurons for the first hidden neuron: Between 5 and 50 with intervals of 2;
- Existence of second hidden layer and if so, the number of neurons

There is the possibility of adding more parameters or even optimize their selection but the convergence of results would be significantly more difficult. The optimization goal was selected to maximize the accuracy. The hyperparameter runs were made using two different algorithms: Adam and mini batch gradient descent. For each attempt, 400 trails were made and every one of them was executed at least two times to reduce the variance with random initial weights to improve the accuracy of search. Plus, the first 50 trails were random to allow enough trials for a Gaussian distribution. With that, considering the goal of maximizing the MLP's accuracy, the two results of the best performing models for each optimization algorithm are displayed in table 4.4.

Interestingly enough, the best performing structure is a single hidden layer with 23 neurons applied, in which for both optimizers, all three activation functions are valid. However, the learning rate required

Optimizer	Learning Rate	Activation Function	Nº neurons 1st layer	Nº neurons 2nd layer
Adam	0.1	sigmoid	23	-
	0.1	sigmoid	23	-
RMSProp	0.00001	relu	49	-
	0.01	tanh	49	-

Table 4.4: Hyperparameter tuning MLP Model 1

for the Adam optimizer was higher than the more flexible RMSProp. The adaptive algorithm though still needed some fixed parameters for this analysis:

- Momentum = 0.0
- ρ = 0.9

whereas the Adam optimizer only requires the learning rate as a parameter. For comparison of hyperparameter tuning results with the ones of Model 1, there is a caveat: for a binary classification, the keras metrics interface, where precision and recall are determined, only consider the positive class. Accuracy, on the other hand, is obviously determined by looking at the whole prediction results. So table 4.5 represents this comparison, where it is required the row from the class ”Fatality” on table 4.1 for MLP to represent the original model. It is notable and surprising to notice the results of this tuning are worse than the original solution proposed. Taking into account that the positive class is the minority, it is very notable the decline of recall. Despite the number of correct predictions is maintained, there is more tendency for the tuning to predict the majority class.

Type of tuning	Precision	Recall	Accuracy
Original Model	0.75	0.59	0.83
Adam Optimizer	0.73	0.47	0.82
RMSProp Optimizer	0.74	0.38	0.81

Table 4.5: Hyperparameter tuning of MLP results for Model 1

For Model 2, the same type of analysis was performed, and with the same goal, which is to maximize the score of accuracy. As seen with table 4.6, the best two performers were considered. The most remarkable aspect of this analysis is that, in three out of four examples, the best tuning with a lower data set required two layers with a considerable amount of neurons. This type of structure usually favors

Optimizer	Learning Rate	Activation Function	Nº neurons 1st layer	Nº neurons 2nd layer
Adam	0.1	sigmoid	33	40
	0.1	sigmoid	49	50
RMSProp	0.1	sigmoid	25	34
	0.1	sigmoid	5	0

Table 4.6: Hyperparameter for Multilayer Perceptron for Model 2

larger amounts of data, and so further analysis and comparison with other tuning functions and the original model is needed in order to get a broader view of the validity of this model.

Concerning the RF algorithm, a random search and a Bayesian Optimization were applied using the scikit-learn library for Python and the scikit-optimize respectively, an extension that is a sequential model-based optimization where it is possible to perform the Bayesian Optimization [39]. A grid of parameters was built in order to perform the tuning:

- Number of estimators: between 400 and 4000, with steps of 400
- The maximum depth of the tree: None, 1-5 nodes
- The minimum number of samples
- The minimum number of samples to be at a leaf node
- Which features to deal with when splitting a node: 'auto','sqrt','log2'
- Whether or not bootstrap samples are used for building estimators

Both tuning functions (BayesSearchCV and RandomizedSearchCV) are used within the scikit-learn library scope, where the training data has to be fitted with the function so it can be trained. Unlike the grid search where all parameters are tested, both functions work within a search space that its built beforehand, as seen above, but there are a number of iterations (or trials like in the case of keras tuner) to be used within the confines of that search space. Those number of trials are repeated for a cross validation of data being used, where in the case of binary and multiclass classifiers, a stratified fold is the best strategy to build it. After the training, it is possible to extract information about the best hyperparameter: its score, parameters. It is also possible to specify with each cross-validation split of data.

Hyperparameter Tuning	Nº trees	Nº Leaf	Nº Split	Bootstrap	Max features	Max Depth
Bayesian Optimization	800	4	5	False	sqrt	-
Random Search	3600	1	10	False	sqrt	-

Table 4.7: Hyperparameter with Random Forest Model 1

The notable part is the convergence about certain parameters both in the Bayesian Optimization and Random Search functions, such as the absence of bootstrap samples, which might indicate there is enough data points to build the estimator, the absence of a maximum depth of trees, and therefore building trees can be more flexible and there is also convergence about how the maximum number of features to be used is determined. There is though differences on how the tree should be built in terms of minimum number of samples in a leaf node and for splitting a node through the branch of the tree. The comparison between the hyperparameter tuning functions and the proposed model are shown in table 4.8. Interestingly, the hyperparameter tuning was not able to improve with the state space provided and the iterations made for 5 cross-validators of data, with worse results across all relevant metrics. The notable changes are a decrease of recall that decreases the f1-score and a lower accuracy. This means that the FNs increase, which are costlier than FPs and correct predictions decrease, so this reduction

Model 1	Type	Precision	Recall	f1-score	Accuracy
Original Model	Macro	0.88	0.86	0.87	0.90
	Weighted	0.89	0.90	0.89	
Bayesian Optimization	Macro	0.88	0.84	0.86	0.89
	Weighted	0.89	0.89	0.89	
Random Search	Macro	0.88	0.84	0.86	0.89
	Weighted	0.89	0.89	0.89	

Table 4.8: Results for Model 1 with hyperparameter tuning for Random Forest

on these parameters is even more consequential. However, the changes are subtle therefore indicating the original model that was designed maximized the capabilities of the data set and the overall results of all performance measures are still between 85-90% which validates the model created.

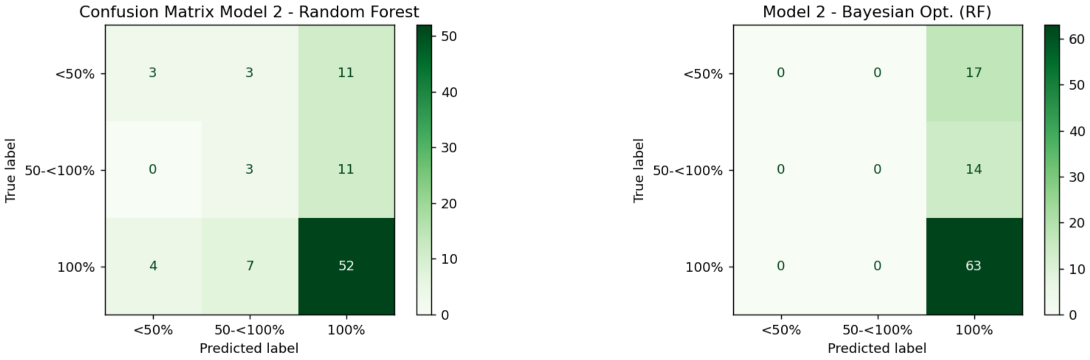
For Model 2, the classification metrics of keras-tuner will separate the false positives from the false negatives regardless whether or not a multi-label classification is being considered. By comparison with the hyperparameter tuning of the RF, the scikit-learn allows for a more complete evaluation of the performance measures after the tuning. And since both use bayesian optimization as a function for tuning, combining with a lack of data, there is similarity between the two results as presented in table 4.9.

Type of Learning	**Type of Tuning**	**Accuracy**	**Precision**	**Recall**
Original Model	None (RF)	0.62	0.71	0.83
	None (MLP)	0.71	0.74	0.92
Random Forest	Bayesian Optimization	0.66	0.67	0.98
	Random Search	0.67	0.72	0.81
Multilayer Perceptron	Adam Optimizer	0.20	0.91	0.87
	RMSProp Optimizer	0.19	0.90	0.83

Table 4.9: Performance results for the class ”100% Fatality” with the hyperparameter tuning

What happens is that the bayesian optimization of the RF will have more of an exploitative approach than an exploratory one, therefore the predictions will be overwhelmingly towards the dominant class ”100% Fatality” because for the accuracy metric, since that class is more represented, it is a safer option to predict solely on that class (figure 4.10(a)). Whereas the Random Search, because of its more stochastic approach in selecting the parameters for tuning, may select from the search space a solution from an exploratory parameter, and with that, all three classes are at least somewhat selected (figure 4.10(a))

From keras-tuner, it is very difficult to have a complete understanding of multi-label classification when all metrics such as precision or recall only calculates as a binary operation between FPs, FNs and TPs regardless of class. From results in table 4.9, the tuning for MLP produced low accuracy, high precision and high recall results with both optimizers. In other words, the tuning is failing to predict correctly any sort of labels, and when confronting the fact that, from the original model, accuracy is 60%,



(a) Random Search

(b) Bayesian Optimization

Figure 4.10: Confusion matrix for Model 2 after hyperparameter tuning with two functions

it is assumed that the tuning for MLP was a failure.

Chapter 5

Conclusions

In this dissertation, a modelling system for the aviation industry based with data from incident and accident reports, was proposed. The main objective in most systems of this industry area is to improve the safety standards of a highly regulated industry, where despite the low risk of failure, the consequences can produce devastating results. From the ASN database, it was possible to join existing information about contributory causes preceding the occurrence and its result for the aircraft, with more information about the phase of flight, the damaged sustained and the mortality.

Since there is more work done regarding the role of human factors in safety procedures for aviation, a correlation was made with those contributory causes and human factors by applying the HFACS taxonomy and its extension, HFACS-ME for maintenance related failures. With this correlation, a labelled database tailored for machine learning was created for prospective models, with the aim of having the ability to predict fatal accidents and the degree of those fatalities, when compared to the total number of people who were present during the flight. After those models were created, three algorithms were proposed based upon previous work with the ASN database: two supervised learning algorithms, Random Forest and Neural Network, and a semi-supervised learning algorithm called Active Learning. Supervised Learning techniques have been implemented for other high-risk industries with success, and semi-supervised learning has been emerging as an alternative, when labelling data can prove costly and only lower amounts of data are available. The parameters selected was based upon the intricacies of both models, and with the intent of developing learning abilities capable of being replicated with new data. The parameters of the learning algorithms were carefully constructed in order to build a structure tailored for each model that fits the features available in the database and encourage learning capable of being replicated with new data instead of memorizing the dataset.

With the use of all techniques, it was proven that both models were capable to learn from the data given, especially for Model 2, where there was a less quantity of data. For Model 1, RF was capable of producing better results than MLP across all metrics, especially when predicting correctly a fatal accident instead of non-fatal. The semi-supervised learning was also tested for this model, but there were no relevant improvements when comparing with the other supervised learning methods. For Model 2, with imbalanced classes and lower class, the RF and MLP struggled with predicting those classes however

RF did perform better. With AL though, after a run of querying, the algorithm was able to improve significantly from the supervised learning methods, proving that with a lower amount of labelled and unlabelled data, there is a capability of training models with quality.

Finally, functions were computed to improve the algorithms performance for each model, with no improvements for Model 1 either with RF or MLP, and ineffective for Model 2 where the performance results decrease dramatically and the dataset is not capable for this sort of operation.

5.1 Future Work

A first step to build upon the work performed with this dissertation is to extend the database to incorporate more data points, for analysis such as the one made for Model 2, where the supervised learning techniques struggled with some of training. For example, adding more 10 years of data might be beneficial in twofold: with a more robust database, the findings for some models can be confirmed and new tendencies can be explored.

Another aspect is regarding the HFACS taxonomy and the process of labelling. There has been work done by implementing a network of human factors with the use of taxonomy, and with the assistance of methods such as fuzzy logic, it could be possible to question several experts, for instance academics, engineers or maintenance specialists, to build a heterogeneous consensus between what type of human factors are related to contributory causes of accident [32]. On top of those findings, for biases purposes when building networks such as Multilayer Perceptron, it is possible to build comparisons between human factors to further boost the robustness of a model, or even to get more insight and interpretability from the algorithm and improve the model's performance [51, 55].

Bibliography

[1] ICAO. *Safety Management Manual*. Montreal, third edition, 2013. DOC 9859 AN/474.

[2] ICAO. *Annex 19 to the Convention on International Civil Aviation*, chapter 1, page 2. International Civil Avaition Organization, 2nd edition, July 2016.

[3] L. F. Santos and R. Melicio. Stress, pressure and fatigue on aircraft maintenance personal. *International Review of Aerospace Engineering*, 2019.

[4] D. A. Wiegmann and S. A. Shappell. *A Human Error Approach to Aviation Accident Analysis*. Ashgate Publishing Ltd., 2003.

[5] E. Mazareanu. Passenger air traffic each year, May 2021. URL `https://www.statista.com/statistics/564717/airline-industry-passenger-traffic-globally/`.

[6] N. G. Dias, L. F. Santos, and R. Melicio. Aircraft maintenance professionals: Stress, pressure and fatigue. In *MATEC Web of Conferences*, volume 304, page 06001. EDP Sciences, 2019.

[7] A. Hobbs, K. B. Avers, and J. J. Hiles. Fatigue risk management in aviation maintenance: current best practices and potential future countermeasures. Technical report, National Aeronautics and Space Administration Moffett Field CA Ames Research, 2011.

[8] H. Rashid, C. Place, and G. R. Braithwaite. Eradicating root causes of aviation maintenance errors: introducing the AMMP. *Cognition, technology & work*, 16(1):71–90, 2014.

[9] T. Madeira, R. Melício, D. Valério, and L. Santos. Machine learning and natural language processing for prediction of human factors in aviation incident reports. *Aerospace*, 8(2):47, 2021.

[10] FAA. FAA Definitions. `http://www.faa-aircraft-certification.com/faa-definitions.html`, 2007. accessed: 29.06.2020.

[11] H. Rashid, C. Place, and G. Braithwaite. Helicopter maintenance error analysis: Beyond the third order of the HFACS-ME. *International Journal of Industrial Ergonomics*, 40(6):636–647, 2010.

[12] EASA. Part-145. In *Continuing Airworthiness Requirements*. European Aviation Safety Agency, October 2008.

[13] M. Ward, N. McDonald, R. Morrison, D. Gaynor, and T. Nugent. A performance improvement case study in aircraft maintenance and its implications for hazard identification. *Ergonomics*, 53(2):247–267, 2010.

[14] S. Watson. On risks and acceptability. *Journal of the society for radiological protection*, 1(4):21, 1981.

[15] EASA. ICAO Annex 19. In *Safety management, International Standards and Recommended Practices*. European Aviation Safety Agency, July 2013.

[16] ICAO. *Annex 13 to the Convention on International Civil Aviation*, chapter 1, pages 1–3. International Civil Avaition Organization, 11th edition, July 2016.

[17] B. A. Turner, N. Pidgeon, D. Blockley, and B. Toft. Safety culture: its importance in future risk management. In *Position paper for the second World Bank workshop on safety control and risk management, Karlstad, Sweden*, pages 6–9, 1989.

[18] N. F. Pidgeon. Safety culture and risk management in organizations. *Journal of cross-cultural psychology*, 22(1):129–140, 1991.

[19] J. Reason. *Human error*. Cambridge University Press, 1990.

[20] S. A. Shappell and D. A. Wiegmann. The human factors analysis and classification system–HFACS. *Embry-Riddle*, 2000.

[21] M. De Ambroggi and P. Trucco. Modelling and assessment of dependent performance shaping factors through analytic network process. *Reliability Engineering & System Safety*, 96(7):849–860, 2011.

[22] T. L. Saaty, L. G. Vargas, et al. *Decision making with the analytic network process*, volume 282. Springer, 2006.

[23] S. A. Shappell and D. A. Wiegmann. US naval aviation mishaps, 1977-92: differences between single-and dual-piloted aircraft. *Aviation, space, and environmental medicine*, 67(1):65–69, 1996.

[24] S. A. Shappell and D. A. Wiegmann. A human error approach to accident investigation: The taxonomy of unsafe operations. *The International Journal of Aviation Psychology*, 7(4):269–291, 1997.

[25] E. Edwards. Introductory overview. In *Human factors in aviation*, pages 3–25. Elsevier, 1988.

[26] E. Edwards. Man and machine: Systems for safety. In *Proceedings of British Airline Pilots Association Technical Symposium*, pages 21–36, London, 1972. British Airline Pilots Association.

[27] D. A. Wiegmann and S. A. Shappell. A human error analysis of commercial aviation accidents using the human factors analysis and classification system (HFACS). Technical report, Office of Aviation Medicine, FAA, February 2001.

[28] S. Reinach and A. Viale. Application of a human error framework to conduct train accident/incident investigations. *Accident Analysis & Prevention*, 38(2):396–406, 2006.

[29] J. Schmidt, D. Schmorrow, and M. Hardee. A preliminary human factors analysis of naval aviation maintenance related mishap. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 44, July 2000.

[30] E. Yazgan. Development taxonomy of human risk factors for corporate sustainability in aviation sector. *Aircraft Engineering and Aerospace Technology*, 2018.

[31] Y.-H. Chang and Y.-C. Wang. Significant human risk factors in aircraft maintenance technicians. *Safety science*, 48(1):54–62, 2010.

[32] E. Zarei, M. Yazdi, R. Abbassi, and F. Khan. A hybrid model for human factor analysis in process accidents: Fbn-hfacs. *Journal of loss prevention in the process industries*, 57:142–155, 2019.

[33] A. Rostamabadi, M. Jahangiri, E. Zarei, M. Kamalinia, S. Banaee, and M. R. Samaei. A novel fuzzy bayesian network-hfacs (fbn-hfacs) model for analyzing human and organization factors (hofs) in process accidents. *Process Safety and Environmental Protection*, 132:59–72, 2019.

[34] F. Li, C.-H. Chen, P. Zheng, S. Feng, G. Xu, and L. P. Khoo. An explorative context-aware machine learning approach to reducing human fatigue risk of traffic control operators. *Safety science*, 125: 104655, 2020.

[35] L.-Z. Xiao, D.-X. Sun, G.-P. Xing, and Y. Huang. Risk assessment model of error in aviation maintenance based on integrated neural networks. In *2011 International Conference on Quality, Reliability, Risk, Maintenance, and Safety Engineering*, pages 633–636. IEEE, 2011.

[36] W. Qiao, Y. Liu, X. Ma, and Y. Liu. A methodology to evaluate human factors contributed to maritime accident by mapping fuzzy FT into ANN based on HFACS. *Ocean Engineering*, 197:106892, 2020.

[37] L. Yin, H. Wang, W. Fan, L. Kou, T. Lin, and Y. Xiao. Incorporate active learning to semi-supervised industrial fault classification. *Journal of Process Control*, 78:88–97, 2019.

[38] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[39] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[40] D. T. Larose and C. D. Larose. *Discovering knowledge in data: an introduction to data mining*, chapter 8–9, pages 165–208. John Wiley & Sons, 2nd edition, 2014.

[41] A. Hemmati-Sarapardeh, A. Larestani, M. Nait Amar, and S. Hajirezaie. Intelligent models. In *Applications of Artificial Intelligence Techniques in the Petroleum Industry*, chapter 2, pages 23–50. Gulf Professional Publishing, 2020.

[42] S. Haykin. *Neural networks: a comprehensive foundation*, chapter 1,4, pages 23–,178–278. Prentice hall, 1999.

[43] F. Chollet et al. Keras. `https://keras.io`, 2015.

[44] M. Castelli, L. Vanneschi, and Álvaro Rubio Largo. Supervised learning: Classification. In S. Ranganathan, M. Gribskov, K. Nakai, and C. Schönbach, editors, *Encyclopedia of Bioinformatics and Computational Biology*, pages 342–349. Academic Press, Oxford, 2019.

[45] B. Settles. Active learning literature survey. *Science*, 10(3):237–304, 1995.

[46] J. Wu, X.-Y. Chen, H. Zhang, L.-D. Xiong, H. Lei, and S.-H. Deng. Hyperparameter optimization for machine learning models based on bayesian optimization. *Journal of Electronic Science and Technology*, 17(1):26–40, 2019.

[47] H. Ranter and F. I. Lujan. Aviation Safety Network, 2016. URL `https://aviation-safety.net/about/`. (accessed: 03.02.2021).

[48] E. Organisation for Safety of Air Navigation. SKYbrary, 2017. URL `https://www.skybrary.aero/index.php/Main_Page#operational-issues`. (accessed: 04.02.2021).

[49] B. C. Love. Comparing supervised and unsupervised category learning. *Psychonomic bulletin & review*, 9(4):829–835, 2002.

[50] R. Sathya, A. Abraham, et al. Comparison of supervised and unsupervised learning algorithms for pattern classification. *International Journal of Advanced Research in Artificial Intelligence*, 2(2): 34–38, 2013.

[51] O. Csiszár, G. Csiszár, and J. Dombi. Interpretable neural networks based on continuous-valued logic and multicriteria decision operators. *Knowledge-Based Systems*, 199:105972, 2020.

[52] G. Montavon, W. Samek, and K.-R. Müller. Methods for interpreting and understanding deep neural networks. *Digital Signal Processing*, 73:1–15, 2018.

[53] A. Fernández, S. García, M. Galar, R. C. Prati, B. Krawczyk, and F. Herrera. *Learning from imbalanced data sets*, volume 11. Springer, 2018.

[54] T. Danka and P. Horvath. modAL: A modular active learning framework for Python. 2018. URL `https://github.com/cosmic-cortex/modAL`. available on arXiv at `https://arxiv.org/abs/1805.00979`.

[55] C. Kahraman, T. Ertay, and G. Büyüközkan. A fuzzy optimization model for QFD planning process using analytic network approach. *European journal of operational research*, 171(2):390–411, 2006.

Appendix A

A.1 Categorization of Contributory Causes using HFACS and HFACS-ME

Contributory Causes	Taxonomy HFACS and HFACS-ME (3rd order)		
ATC & Navigation			
ATC & Navigation - Altimeter setting	Physical Environment		
ATC & Navigation - Language/communication problems	Skill-Based Error	Crew Resource Management	
ATC & Navigation - Preamture descent	Skill-Based Error	Physical Environment	Physical/Mental Limitations
ATC & Navigation - VFR flight in IMC	Skill-Based Error	Crew Resource Management	
Airplane - Airframe			
Airplane - Airframe - Cargo door	Skill-Based Error		
Airplane - Airframe - Passenger door	Skill	Inadequate Design	Supervisory Violation
Airplane - Airframe - Wing			
Airplane - Engines	Routine Violation	Inadequate Design	Skill
Airplane - Engines - APU			
Airplane - Engines - Fire			
Airplane - Engines - Fuel contamination	Routine	Inadequate Supervision	Inadequate Documentation

Table A.1 continued from previous page

Contributory Causes	Taxonomy HFACS and HFACS-ME (3rd order)		
Airplane - Engines - Fuel exhaustion	Routine	Inadequate Supervision	Inadequate Documentation
Airplane - Engines - Fuel starvation	Skill-Based Error	Personal Readiness	
Airplane - Engines - Loss/opening of engine cowling	Skill-Based Error	Infraction	Inadequate Design
Airplane - Engines - Prop/turbine blade separation	Routine	Inaccessible	Inadequate Design
Airplane - Engines - Reverse thrust/prop ground fine pitch	Exceptional Violation	Technological Environment	
Airplane - Engines - Shutdown of wrong engine	Exceptional Violation	Physical Environment	
Airplane - Flight control surfaces - Elevator	Inadequate Design	Uncorrected Problem	Supervisory Violation
Airplane - Flight control surfaces - Flaps			
Airplane - Flight control surfaces - Horizontal stabilizer	Inadequate Design	Uncorrected Problem	Supervisory Violation
Airplane - Instruments	Perceptual Error	Technological Environment	
AIrplane - Pressurization	Exceptional Violation	Physical/Mental Limitations	
Airplane - Systems - Hydraulics	Routine	Inadequate Documentation	Operational Process
Airplane - Undercarriage - Brakes	Routine Violation	Technological Environment	Inadequate Supervision
Airplane - Undercarriage - Gear-up landing	Perceptual Error	Physical Environment	Dated/Uncertified Equipment
Airplane - Undercarriage - Landing gear collapse	Routine	Inappropriate Operations	Rule
Airplane - Undercarriage - Premature retraction on take-off	Skill-Based Error		

Table A.1 continued from previous page

Contributory Causes	Taxonomy HFACS and HFACS-ME (3rd order)		
Cargo			
Cargo - CofG	Personal Readinees	Inadequate Supervision	
Cargo - Overloaded	Personal Readinees	Inadequate Supervision	
Cargo - Shift	Personal Readinees	Inadequate Supervision	
Collision - Aircraft - In flight	Exceptional Violation	Physical/Mental Limitations	Plan Inappropriate Operation
Collision - Aircraft - On Ground (platform)	Infraction		
Collision - Aircraft - On Ground (runway incursion)	Infraction	Attention	
Collision - Object			
Collision - Object - Airport equipment	Decision Error	Adverse Mental State	Adverse Physiological State
Collision - Object - Approach, rwy lights			
Collision - Object - Bird			
Collision - Object - Houses, buildings			
Collision - Object - Mast/pole wires	Decision Error	Personal Readiness	
Collision - Object - Person, animal			
Collision - Object - Trees			
Collision - Object - Vehicle	Perceptual Error	Physical/Mental Limitations	Plan Inappropriate Operation
Collision - Object - Wall dyke			
External factors - FOD	Routine	Training	Inappropriate Operations
External factors - Wake vortex			
External factors - Wind, hail			
Fire			

Table A.1 continued from previous page

Contributory Causes	Taxonomy HFACS and HFACS-ME (3rd order)		
Fire - Fire			
Fire - Fire resulting from tire failure	Lighting	Decision Error	
Fire - Fire during refueling	Skill	Qualification	
Fire - Hangar, ground fire			
Fire - Inflight			
Fire - Lithium battery thermal event			
Flightcrew - Alcohol, drug usage	Exceptional Violation	Personal Readiness	
Flightcrew - Disorientation, situational awareness	Perceptual Error	Physical/Mental Limitations	
Flightcrew - Distraction in cockpit	Routine Violation	Crew Resource Management	Operational Process
Flightcrew - Incapacitation	Advrese Physiological State	Physical Environment	
Flightcrew - Insufficient rest/fatigue	Adverse Mental State		
Flightcrew - Non adherence to procedures	Routine Violation	Supervisory Violation	
Flightcrew - Un(der)qualified	Fail to Correct Known Problem	Resource Management	
Landing/takeoff - Landing - Bounced	Skill-Based Error	Personal Readiness	
Landing/takeoff - Landing - Fast	Skill-Based Error	Personal Readiness	
Landing/takeoff - Landing - Heavy	Skill-Based Error	Personal Readiness	
Landing/takeoff - Landing - Late, far down rwy	Skill-Based Error	Personal Readiness	
Landing/takeoff - Landing - Unstabilized approach	Perceptual Error	Physical Environment	Adverse Mental State
Landing/takeoff - Landing - Wrong runway/taxiway			

Table A.1 continued from previous page

Contributory Causes	Taxonomy HFACS and HFACS-ME (3rd order)		
Landing/takeoff - Tailstrike	Decision Error	Physical Environment	
Landing/takeoff - Takeoff - Aborted	Exceptional Violation		
Landing/takeoff - Takeoff - Locked rudders/ailerons/gustlock	Technological Environment	Uncorrected Problem	Supervisory Violation
Landing/takeoff - Takeoff - Wrong runway (runway confusion)	Perceptual Error	Crew Resource Management	Inadequate Supervision
Landing/takeoff - Takeoff - Wrong takeoff conf (flaps/trim)	Decision Error	Crew Resource Management	
Maintenance			
Maintenance - Engine (deficient, inspections, etc)	Inadequate Supervision	Inadequate Design	
Maintenance - Failure to follow AD and SB's	Infraction		
Maintenance - Substandard practices (general)	Routine	Inadequate Design	
Maintenance - Wrong installation of parts	Inappropriate Operations	Training	Inadequate Supervision
Security - Suicide	Exceptional Violation	Adverse Mental State	Fail to Correct Known Problem
Unknown - Cause Undetermined			
Unknown - Missing			
Weather	Physical Environment		
Weather - Icing	Physical Environment		
Weather - Lightningstrike	Physical Environment		

Table A.1 continued from previous page

Contributory Causes	**Taxonomy HFACS and HFACS-ME (3rd order)**		
Weather - Thunderstorm	Physical Environment		
Weather - Turbulence	Physical Environment		
Weather - Visibility - low	Physical Environment		
Weather - Windshear/downdraft	Physical Environment		

Table A.1: Categorization of all contributory causes from ASN database into 3rd order HFACS/HFACS-ME taxonomy

A.2 Distribution of 3rd Ordered HFACS and HFACS-ME

3rd Order HFACS/HFACS-ME	Encoding (fig 3.2)	Frequency
Adverse Mental State	1	73
Adverse Physiological State	2	19
Crew Resource Management	3	11
Dated/Uncertififed Equipment	4	67
Decision Error	5	62
Exceptional Violation	6	20
Fail to Correct Known Problem	7	12
Inaccessible	8	2
Inadequate Design	9	42
Inadequate Documentation	10	36
Inadequate Supervision	11	63
Inappropriate Operations	12	76
Infraction	13	1
Lighting	14	1
Operational Process	15	12
Perceptual Error	16	131
Personal Readiness	17	134
Physical Environment	18	216
Physical/Mental Limitations	19	18
Plan Inappropriate Operation	20	1
Resource Management	21	9
Routine	22	114
Routine Violation	23	39
Rule	24	72
Skill	25	29
Skill-Based Error	26	104
Supervisory Violation	27	15
Technological Environment	28	19
Training	29	4
Uncorrected Problem	30	8

Table A.2: The HFACS and HFACS-ME manually labeled from the contributory causes of the database, with their frequency on the database.