



**Escola d'Enginyeria de Telecomunicació i
Aeroespacial de Castelldefels**

UNIVERSITAT POLITÈCNICA DE CATALUNYA

TREBALL FINAL DE GRAU

**TÍTOL: AIRCRAFT CONFLICT DETECTION USING DATA MINING AND
STATISTICAL TOOLS**

TITULACIÓ: Grau en enginyeria d'Aeronavegació

AUTOR: Marc Pérez i Font

DIRECTOR: Cristina Barrado Muxi

DATA: 5 d'Octubre de 2016

Títol: Aircraft conflict detection using data mining and statistical tools

Autor: Marc Pérez i Font

Director: Cristina Barrado Muxi

Data: 5 d'Octubre de 2016

Resum

Avui en dia i tenint en compte els indicadors i la previsió d'organismes com Eurocontrol qui afirma que de cara al 2030 el transit aeri a Europa es doblarà, es de vital importància trobar mètodes eficaços tals que permetin absorbir la demanda de capacitat a l'espai aeri europeu. D'aquí la importància de traçar un pla per tal de absorbir tot aquest flux de transit aeri previst com es cas del ATM MASTER PLAN englobat dins de SESAR (Single European Sky and Research).

Actualment el transit aeri és regulat gràcies a una infraestructura que sovint està desfasada ja que molts dels processos que es fan son rudimentaris i pesats, i tots son vigilats per persones (ATC's) que no deixen de ser elements propensos al error. Per tal de monitoritzar el transit aeri s'utilitzen eines de control com per exemple el RAMS PLUS o d'altres softwares equivalents els quals calculen de manera simultània a partir dels FP (Flight plans), de les posicions dels avions i diferents inputs d'aquests les trajectòries 4D per tal de saber si existeix o no algun moment en el que es vulnerin les distàncies mínimes de seguretat exigides per l'organisme competent.

L'objectiu d'aquest treball es testar una eina que permeti detectar conflictes a l'espai i testar fins quin es el rendiment que ofereix davant d'un software convencional de control. El treball es realitzarà íntegrament dins l'entorn de Python i s'utilitzaran la biblioteca de Scikitlearn per tal de fer aprendre el computador quins casos son conflictes reals i quins no ho son pas.

Partint de la hipòtesis de que quants mes casos aprèn la màquina, mes acurades son les prediccions en casos diferents, es realitzaran diferents proves per tal d'analitzar la millor configuració del sistema en cada cas. Així doncs es tracta de caracteritzar el sistema i veure el rendiment que aquest ofereix davant d'un sistema tradicional de detecció de conflictes.

Title: Aircraft conflict detection using data mining and statistical tools

Author: Marc Pérez i Font

Director: Cristina Barrado Muxi

Date: 5th of October of 2016

Overview

Nowadays and taking into account the different indicators and the forecast of the air traffic volume from Eurocontrol, it is accepted the idea that the traffic will be doubled by 2030, that is why it is so important to find new and efficient methods which allow the infrastructure absorb such a huge demand. Hence the importance of designs a plan in order to increase the actual capacity. This is the example of ATM MASTER PLAN included inside SESAR (Single European Sky and Research).

On these days, air traffic is regulated by a infrastructure that sometimes can reach its limit due its processes since some of them are rudimentary, tough and all of them are supervised by humans (ATC's) which are elements of the system that can induce an error because its nature. In order to supervise each flight some computational tools are used like RAMS PLUS or other equivalent software which all of them compute from the FP (Flight Plans) of the evolved aircrafts, its positions and other inputs the 4D trajectories in order to prevent the system from a hazard situation.

The main aim of this project is to test a different tool which lead to detect conflicts on the airspace and find which is its performance compared to a traditional 4D forecast trajectory software. It will be entirely developed on the environment of Python and the Scikitlearn library will be used in order to make the machine learn which is a conflict situation and which not

Starting from the idea that in Machine learning it is really important the number of learned situations in order to get a better accuracy in the predictions, different tests will be executed in order to find how good the forecast can be. Once the results are obtained, the system will be compared with a traditional controlling system.

Table of Contents

INTRODUCTION	7
CHAPTER 1. GENERAL OVERVIEW	9
1.1 HISTORY OF ATC'S AND ATMS	9
1.2 TYPES OF AIR TRAFFIC CONFLICT DETECTION	10
1.3 MACHINE LEARNING CONCEPT	13
1.4 INTRODUCTION TO PYTHON AND SCIKIT LEARN LIBRARY	14
1.5 NEST DATABASE	15
1.6 THE CONCEPT OF FREE ROUTE AIRSPACE	16
CHAPTER 2. MODELISATION OF THE PROBLEM	18
2.1 PROBLEM DEFINITION AND ASSUMPTIONS	18
2.2 MODEL HYPOTHESIS AND CONSTRUCTION	19
2.2.A RANDOM MODELISATION OF CONFLICT DETECTION:	19
2.2.B NEST:	22
CHAPTER 3. PERFORMANCE AND RESULTS OF THE MODEL	27
3.1 MODEL PERFORMANCE AND RESULTS	27
CHAPTER 4. CONCLUSIONS AND FURTHER CONSIDERATIONS	35
CHAPTER 5: FURTHER STUDIES AND INVESTIGATIONS	37
BIBLIOGRAPHY	38
ANNEXES	39
ANNEX 1: CODE OF SIMPLE FIGURES	39

INTRODUCTION

On these days, where you can buy something through internet, buy to a national company where the product has been assembled in a foreign country and all the small parts of it have been built on a third country, it is clear that “frenetic” is a perfect word to describe the world where we live. Regarding air traffic conflict detection, it is well known that aircraft conflict detection is one of the most important processes towards ensure the security requirements imposed by organisations such as Eurocontrol.

In order to meet them it is really important to develop tools that allow the ATMS (Air Traffic Management System) monitor the aircraft movement inside the airspace and how it interacts with the whole system. According from a report of Eurocontrol **¡Error! No se encuentra el origen de la referencia.**, on 2030 the 11% of the demand will not be accommodated. Hence, if it is taking into account IATA’s (IATA 2016) forecast for the years to come which ensures that by 2034, the total passengers set will be double to 7 billion and knowing that from 2003 to 2008 the total delay was increased to 60,7% whilst the air traffic was only increased by 19,9%. All these facts prove that there is the necessity to find new processes and ways to detect a conflict in the airspace in order to prevent the system from a hazardous situation by increasing the total capacity of the system.

In addition, considering the following figure, shows that when demand is increased and its capacity is exceeded, the overall delays are increased as an exponential relation. That is why it is necessary to build new and robust ways to enhance how the system works. This would lead to the ability of giving shelter to the forecasted demand and that all processes in this world will be on time.

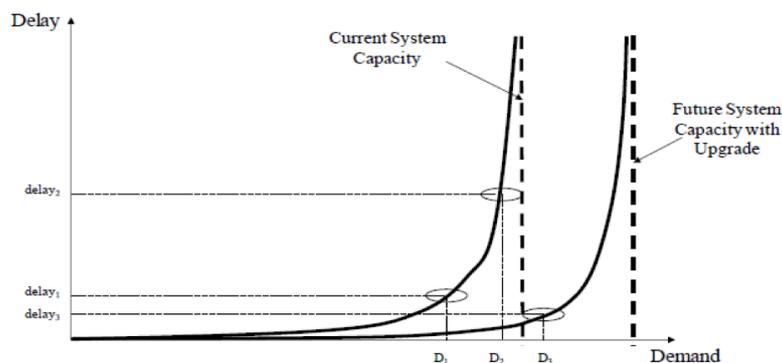


Figure 1 Delay vs. Demand, extracted from Ball et Al., 2010

Over the last years, the concept of the Machine learning has taken a huge importance in the industry, for example, many important worldwide companies like Amazon or Microsoft has implemented this technology to offer to their customers the ability to

perform a prediction of the demand of some products to plan the production. Another example is the ability to make a binary classification like a credit risk prediction, or the twitter sentiment analysis to reach the required critical mass of the possible consumers. It is known in computational technology as the process of making the machine to make a prediction of a scenario based on the experience and knowledge from many cases of the past.

Nowadays the trend of global aviation is to fly under IFR rules where navigation is done by airways or through waypoints depending on each free flight rules due the benefits of alleviating part of the workload from the ATC's due the decentralisation of the conflict resolution. Since the resolution of hazardous situations would be made mainly on the skies between the aircrafts, new methods to prevent this type of situations happen should be developed. That is why, if the historic amount of data is considered, a predictive model could be developed towards the conflict resolution as a preventive tool by finding conflictive patterns.

Due all this reasons, Machine Learning could be an interesting way to be applied in air traffic conflict detection in order to improve the actual system and enhance it to achieve better security requirements in order to locate the expected future demand on the years to come and prevent from hazardous situations.

CHAPTER 1. GENERAL OVERVIEW

1.1 History of ATC's and ATMS

From the beginning of aviation, the workload of air traffic controllers has suffered many changes according to the demand of the world. On the beginning, the aircrafts were controlled only by people on ground stations who predicted the trajectories of the planes by asking by radio to the flight dispatchers pilots' position, speed (horizontal) and course in order to predict if two aircrafts would enter in conflict during the minutes to come. If their predict that a conflict was going to happen, the ATC would give the necessary indications to the pilots to change or their flight level or their course. This was a really tough and inefficient way to control them since the radio only allowed one conversation and in one direction and it was necessary to be updating constantly the data in order to ensure security.

Due the WWI (World War I) the aviation suffered a huge development. During the mid 20's, after the first transcontinental air mail service was established, the first beacons were located along the routes of the air postal service. This fact allowed the pilots to perform visual flight during the night. One thing was fair clear, and is that some how of process must be implemented to prevent the aircraft from colliding one with each other without the necessity of flying under VFR.

After a few years, on 1930, the first radio-equipped control room was ever created and that fact led the air traffic flow management grow and expand outside the fence of the airport to the skies. Since radio communications offered a way to talk directly with pilots as equal as transmit signals to give a focal point to the airplanes, a huge investment was done since the benefits were worthy.

During that decade, the task of controlling the airplanes was still tough because controllers were using maps, blackboards and hand made calculations to predict the aircraft positions along time and ensure surveillance along their flight. Since the ATC's had no direct link to communicate with pilots, the controlling task was still really laborious.

Again, due WWII the demand of moving freight and soldiers made the implied countries to invest on research and develop new technologies making planes move faster, increasing their range and height of cruise. The radar was invented and adopted in radiolocation, and allowed the ATC's to permanently locate each aircraft along its flight and prevent them from a hazard situation.

It was on the sixties that computers appeared and were for the first time introduced as a tool of control since they took inputs from the flight plans, and the radar lectures, information was shown as dots in screen with information of speed, altitude and course. Many facilities were built in order to prevent air traffic congestions in critical zones as busies airways and the surroundings of the most important airports.

1.2 Types of Air traffic conflict detection

Nowadays there are many different types of Air traffic conflict detections tools. Many reports have been issued explaining rather than suggesting which could be the best, classify them by their common things and explaining which benefits offers each of them to overcome a conflict between two or more aircraft. Kuchar (Kuchar and Yang 2000) explains on his report the differentiation of them according to the following categorisations:

- **STATE PROPAGATION:** Since the model of conflict detection tries to predict where the aircrafts will be on the near future, depending on how each future state of the aircraft is computed there are three categorisations: *Nominal, Worst-Case or Probabilistic*.

What the Nominal does is to directly extrapolate from the data of the flight plan the future position of the aircraft, that does not consider any kind of uncertainty and says that the aircraft behaviour will always be the same.

On the other hand, what the worst-case considers is that the aircraft can perform any kind of maneuvers, so at the end what it is found is a potential collection of different kind of scenarios that could happen in the near future. If each of them were considered, many false alarms would be detected. That is why this kind of process is only considered in the short-term time.

In the probabilistic methods, what it is computed is which could be the most potential trajectories according to a Nominal trajectory, but adding some uncertainty in order to make as real as possible. This last method helps to set the best configuration between the ideal case and the worst case.

- **STATE DIMENSIONS:** Depending on which different dimensions are used, methods of CD&R (Conflict Detection & Resolution) are listed in if they are pure horizontal or vertical, or any of each of the 3 axes. The majority of the models consider the three axes since aircrafts can move either in any of the three directions. Even so, since in some of flight phases the aircraft movement is limited (theoretically) in one plane like the final approach for landing. For example TCAS (Traffic Conflict Avoidance System) takes a variety of different measurements, but just to estimate if a conflict exists on the horizontal plane. The same happens with GPWS (Ground Proximity Warning System), it is a system that detects if a conflict is about to happen when approaching to an airport for landing.

- **CONFLICT DETECTION:** Depending on if a model detects or not if a conflict exists there are classified in this way, if yes or neither. On the one hand, there are models where the definition of what is a conflict in terms of distances and parameters has been uploaded to the system, so once a situation of conflict exists, it alerts the user and says which are the two or more aircraft involved in this conflict.

On the other hand, there are models, which do not say if a conflict exists, or not. What they do is to give the necessary information so the user is the one

that performs this judgement and takes the necessary actions to prevent a possible hazardous situation.

- **CONFLICT RESOLUTION:** As explained before, there are many models that detect if a conflict is about to happen or not. Apart from detecting the conflict, there are some of them that propose a resolution action to solve the situation and prevent two aircraft from violating the minimum-security separation distances.

Normally, the resolution is a simple and standardised action to be performed once the alert is issued to the pilots/ATC. These prescribed actions are as easy as possible in order to not spend too much time by deciding what to do. Even so, these actions are not the optimal ones, for example, GPWS issues an alert of “Pull up” when a conflict exists between the aircraft and the ground. In the case of TCAS, explores which is the less aggressive climb/descent maneuver from a range of possible maneuvers and taking into account a cost function to be optimised. Optimisation is normally done with techniques such as Genetic Algorithms or Game theory.

- **RESOLUTION MANEUVERS:** Depending on the type of resolution maneuver that the pilot must perform can be categorised in the following lists:
 - *Turns:* When turns are performed on the horizontal plane.
 - *Vertical maneuvers:* When climbing or a descent is a proper solution.
 - *Speed Changes:* When the changing of speed is a possible solution.

It is important to consider that some systems perform a combination of any of the previous lists, but as mentioned before, even being more efficient solutions, they are more complex to be understood by pilots.

- **MULTIPLE CONFLICT:** Depending on how a model handles with more than two aircraft conflict, they can be divided on Pairwise, or Global Solution.
 - *Pairwise:* It means that each conflict is detected as a pair of aircraft. It is more easy to handle conflict since only two are aircraft are involved. Even so, one conflict resolution may develop another one, and iteration should be applied until a scenario where conflicts do not exist is found. That would make a really inefficient conflict and also it may not be able to solve a conflict in some specific scenarios.
 - *Global solution:* The system is not analysed pair by pair, it is considered as an entire system where more than two aircraft are present. Compared to Pairwise means that when a conflict resolution would imply another conflict, in the case of a global solution, a optimal decision would be made regarding all the different actors that are involved in the conflict.

As seen in the following figure, the two different approaches to multiple conflicts solving system are shown:

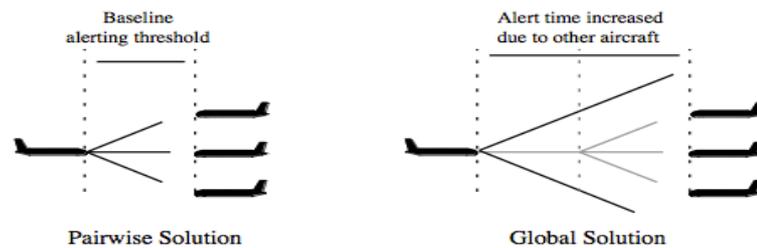


Figure 2 Pairwise and Global solution, extracted from Kuchar, 2000

As seen before, for Pairwise solution, TCAS would alert and advise a maneuver to be performed, but that would develop in another conflict since one aircraft has a plane above and under it. Iteration would be performed between pair of aircrafts until a global solution where conflict does not exist is found.

In another hand, what global solution does is to perform an advisory maneuver to be performed by considering all the aircrafts of an area. This implies that the time to conflict has increased since one or more involved aircraft in the predicted conflict should perform more dramatic maneuvers.

- **OTHER MODEL ELEMENT:** Apart from the different categorisation explained before, each model could be categorised depending on the metrics it uses to perform conflict detection. Another important point to take into account is how each model manages uncertainty in the predictions. Many models limit their time scope in order to prevent from high number of false alarms, as equal as the amount of uncertainty is induced to the system. Also the degree of coordination between aircraft for conflict resolution may be taken as a variable. It is more efficient to make two aircrafts cooperate to solve a problem rather than only one performing a solving action to prevent from a dangerous environment.

In conclusion, there is a huge list of different ways to detect the conflict and automate the ATMS. But, if considering the demand of capacity on the years to come, different approaches may be done to the conflict resolution in order to find efficient and safer ways to solve a possible scenario where a conflict exists. In the design phase security standards may be considered as equal as the amount of workload the ATC may have since the goal is to alleviate part of the stress they suffer, not to give them more work in order to increase the total capacity of the system. In this case, and depending on the degree of accuracy the model may have, it may be used on either of the three different phases of the CFMU (Control Flow Management Unit) which are the strategic, the pre-tactical and the tactical.

1.3 Machine learning concept

During the last decades, many research has been done on the field of develop machines capable to decide by their own depending on the inputs from autonomous cars to searching tools on Internet. Since the available historic data on aviation is huge, that collection could be used for “data mining” or what means, process raw data for a goal or an objective, in this study, detect conflicts in the airspace from historical data.

The main idea is that the computer is capable to create a model from raw data in order to make a prediction of future scenarios. There are different ways and parameters that can be chosen in order to create a model with better accuracy, or if it is not really important, a model which needs few cases to learn and ensure a required ratio of success.

Basically there are two different types of machine learning, which can be divided into supervised or unsupervised learning.

- **Supervised machine learning:** The machine learns from raw data that is pre-processed by the user by giving each scenario a category.
 - **Classification:** When each scenario has to be labelled
 - **Regression:** When the output is a continuous variable
- **Unsupervised machine learning:** The machine learns from raw data that does not have any label. Hence the main goal is to find clusters of scenarios according to its behaviour and find groups from patterns.

Inside each kind of machine learning there are many parameters that can be changed according the expected features that the user want to chose.

The key of machine learning resides on the correct election of those key variables that are relevant of the system. For example, if there were specie of flower that has two varieties according to the size of its petals, a good election of a feature of each case would be the dimensions (x, y) of each petal.

Nowadays, there are no models that use Machine Learning as a main tool to detect conflicts in airspace that is why investigations should be done to test if it could be considered as a good and safe way to detect conflicts in the airspace.

In conclusion, what it is clear is that if security must be ensured according to international safety standards, predictability must be tested and huge amount of historic data should be taking into account in order to meet the desired performance.

1.4 Introduction to PYTHON and SCIKIT learn library

Python was developed by Guido Van Rossum during the nineties decade. During that time, Van Rossum was working on a investigation centre on Holland, he used to work on a project of the creation of Amoeba, an OS where the programming language was focused on to be as easy as possible to be used by the programmers community. Due its facilities offered and since the existence of SCIKIT learn library which is a module developed for machine learning problems, Python will be chosen as the main programming environment.

As mentioned before, SCIKIT learn module gives the user the opportunity to develop projects of Machine Learning and test them. It is a really useful tool and can be used for any kind of Machine Learning problem. In the following flow chart it is shown the best way to chose the best model depending on each problem.

Since the purpose of this project is to decide which scenario is conflictive and which not, what will be predicted is labelled data since the user will previously decide which are conflictive and which not, and after that, the model will be the one to say if new scenarios are conflictive or not.

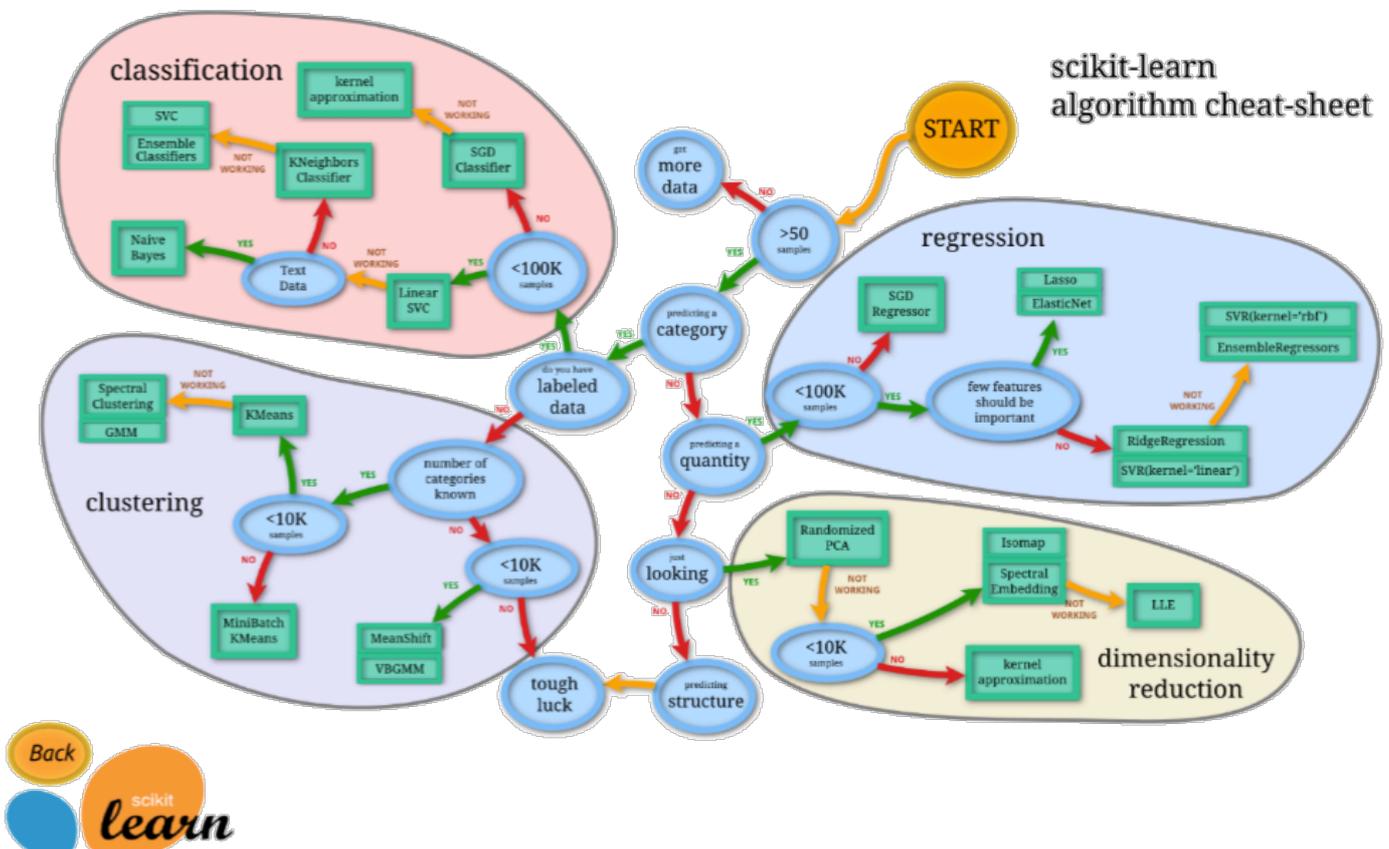


Figure 3 SCIKIT learn FLOW chart Extracted from (SCIKIT Learn 2016)

1.5 NEST Database

NEST is a tool developed by Eurocontrol that allows the user the visualisation of historic events from tons of available data of Europe airspace including plane routes, airspace configuration, or conflict existence during a selected day. It is really important to use it to nourish the prediction model since the idea of including historic data ensures predictability. It is also a good idea to be included since the continuity is given to it. Airways and sectorisation is implicitly included on the aircrafts position and the geometry of the sector. Even sectorisation may change according to a flexible airspace configuration, it may follow patterns according to peaks of demand on the busiest days and seasonally flights of the airliners.

It is a really useful tool for research due the huge amount of data available from the past. Also it is a really good capacity-planning tool due considering different available sector configuration according to the resources available and the demand.

It can be downloaded from EUROCONTROL webpage, and it is totally free if it is for research purposes or the educational environment. For these reasons, it will be used to have access to the big data from the past to ensure predictability and continuity to safety standards.

What is clear is that all the Europe airspace cannot be taken into account, that is why the scope of the project will be limited to a specific zone where testing of the model is interesting. That could be a zone with high air traffic density.

In the following image, a portion of the airspace of Europe is shown. Painted lines are only those flights that a given moment of a day crossed a common airspace before reaching their destination. In this case is the Benelux airspace zone. It is known as being part of the Maastricht upper Airspace Zone (MUAC), which is a really busy zone due its location.

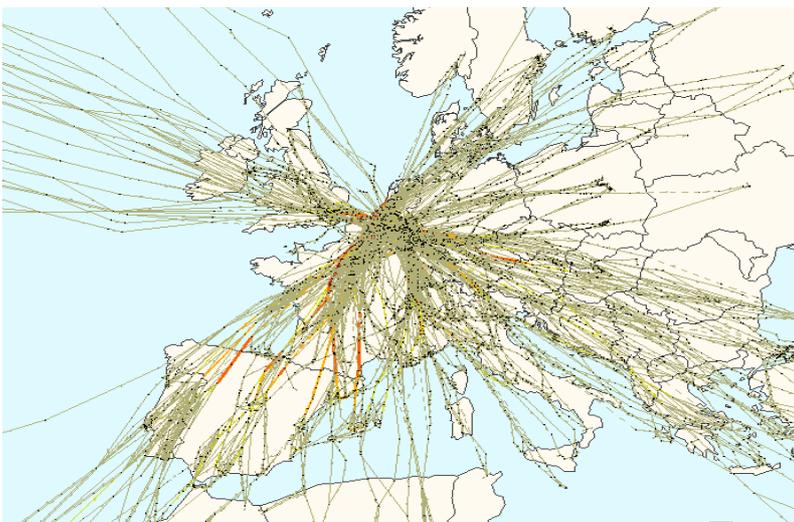


Figure 4 EBBUCTA ACC (Belgium) on 6th of September of 2015, 1716 flights crossed this airspace along that day, extracted from NEST database

1.6 The concept of Free Route Airspace

Since the airspace is the given one, and considering that more aircrafts will be flying through it, different airways and en-route flight rules should be implemented to give shelter to all this excess of demand on the future.

Nowadays air traffic is confined in some defined airways. This fact makes easier for ATC to perform their tasks of surveillance of the airplanes along their path and prevent them from possible conflicts. Since the forecast for the years to come is that demand will be the doubled for 2034, and considering the idea that ATC's will still exist by that year, some improvements may be done in order to alleviate their workload.

The concept of the free-route Airspace (FRA) was introduced on 2008 when Eurocontrol started to develop its implementation on Europe Airspace, the first time it was tested was on 2011. What the concept represents is that airliners can plan freely their desired route between an entry point and an exit point from a sector, always subject to considerations of the ATC's and depending on the airspace availability. This fact helps to reduce distances between origin and destination and as a result a saving on fuel, time and emissions emitted to the atmosphere. In addition, the number of conflicts on the airspace is decreased since aircrafts are more spread around the airspace.

On the other hand, they represent an increment to the workload of the ATC's due variability that each flight can have depending on the destination. This is result of spread the aircrafts and the difficult to detect an eventual conflict between two or more aircrafts.

Nowadays, FRA represents more than 25% of the total distance of European domestic flights, and theoretically, the amount of NM saved are about 7.5million NM which can be traduced to 45000 tons of fuel saved priced in 37 million euros since its implementation at the end of 2014.

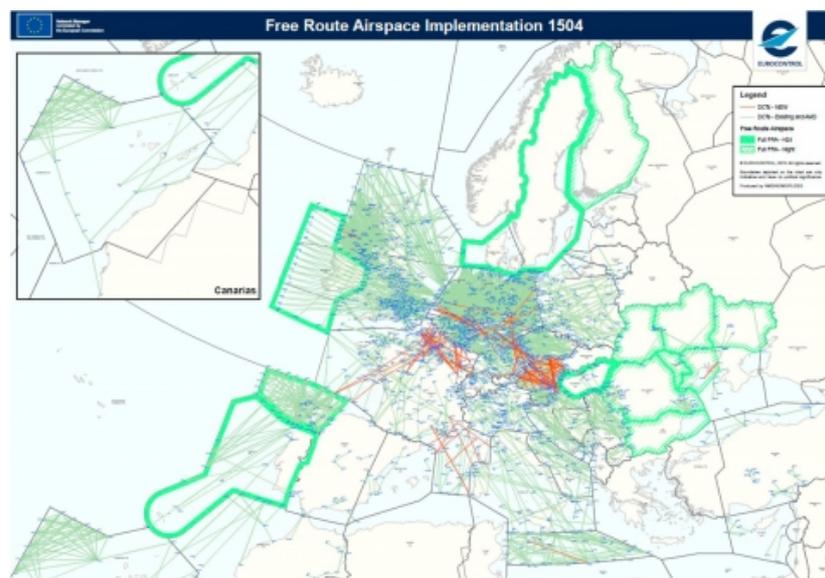


Figure 5 FRA implementation in Europe, extracted from Eurocontrol

Chapter 1. General Overview

That is why if the global trend is to adopt the FRA as the main way to follow a route, some tools may be developed to alleviate this increment of workload on ATC.

The implementation of this concept started on 2010 on the region of Maastricht Upper Airspace Centre (MUAC). It is known in air navigation one of the most busies zones in Europe due its location between London, Paris and Berlin cities.

At the beginning, it was implemented during he least busies hours of the day, which mean during the night that zone and time slots were the number of current flights was not really high. In the case of the MUAC, the implementation process has been the following one:

- 2008: FRA Definition concept and start of implementation
- 2011, March: During least busy hours of the day 00:00 to 06:00 CET
- 2011, June: Extension of night-time, from 00:00 to 08:00 CET
- End of 2011: Weekends, from Sat. 00:00 to Mon: 08:00 CET
- Early 2012: Real time simulations to prove that is feasible
- 2012: Day time on busy Fridays, from 12:00 CET to Mon: 08:00 CET
- 2013 and beyond: 24/7 operations.

FRA boundaries extend from FL 245 to the top of the managed airspace by controllers. Altitudes below FL245 are suitable to be monitorised by ATC's due the amount of traffic that is moving to upper areas to cruise altitudes or lower areas to perform the final approach to the airport.

Taking into account that the trend of air navigation is to fly under free flight navigation rules since the benefits that gives, the goal of Eurocontrol to reduce en-route distance along the years to reduce fuel emissions and also considering that flights in airlines are more or less constant, that is to say that they have more or less a predictive operation plan with some repetitions along every season, a model could be developed by cutting historic traffic and try to find similarities and patterns in time to predict repetitive conflicts on this kind of spaces. Once the interesting traffic has been cut, the rest will be processed with NEST in order to obtain when and where are the conflicts happening.

CHAPTER 2. MODELISATION OF THE PROBLEM

2.1 Problem definition and assumptions

The chosen type of machine learning will be supervised since what we want to predict is if a scenario is or not a conflictive situation. Since this is done by creating new scenarios and taking into account that will be the user the one who chooses the labels, the machine learning process will be done under Supervised Machine Learning rules. The following flow figure shows the typical process of supervised machine learning

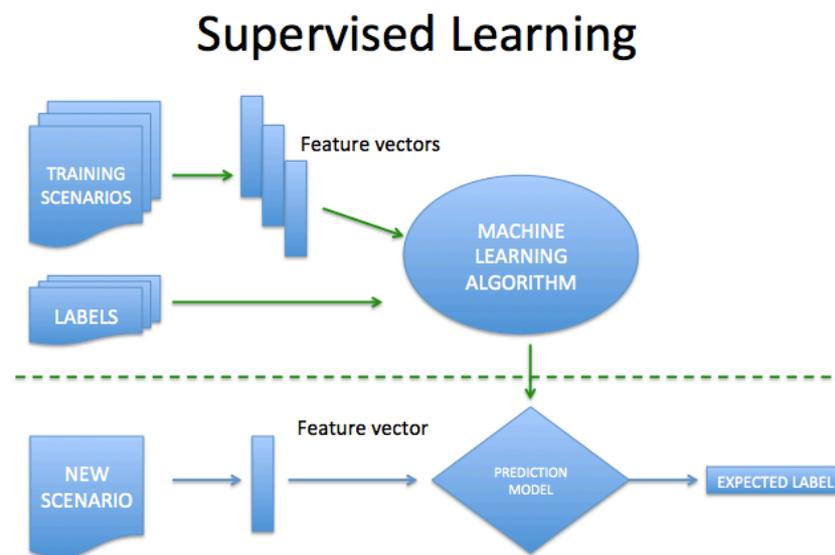


Figure 6 Machine Learning flow for unsupervised learning

As seen in the previous flow chart, in order to generate the model, the user must load some vectorized scenarios with assigned labels in order to allow the machine to approximate each scenario to a model by finding patterns between them.

Since the computational power available resources are the typical ones from a personal computer for this study and also considering that this will be a first review of applying this new technique to conflict detection, different assumptions and simplifications will be considered:

- Aircraft **movement is limited** to only a few directions
- Airspace sector will have a limited but **constant capacity** for a given time
- No vertical movement is considered, **only horizontal movement**
- Only detection on time is performed, **no forecast** in future
- Sectors have a **regular shape**

The algorithm will first learn by memorising which cases represent a conflict and which not in order to perform a prediction in new cases not seen before by the machine.

2.2 Model hypothesis and construction

In order to build the model, first of all it is necessary to find a way to make the airspace comprehensive for the machine. This means looking for the key variable depending on each method. In this project, two different ways to define the airspace and each flight will be tested. One will be done by assuming that the airspace is a square, and each flight is represented by a coloured dot. The other one will be represented as a line where each position variable (latitude, longitude and flight level) for each aircraft at different moments of the day will be added to the key variable line.

2.2.A RANDOM MODELISATION OF CONFLICT DETECTION:

As explained, it is important to find a good way to represent the airspace. Since it can be shown as an image and taking into account that machine learning was used before to detect hand-written numbers, aircrafts can be drawn as a points in the airspace, hence, try to execute a model that detects when two points are adjacent or not.

For example, in the following figures, hand-made numbers are extracted from the digits dataset, available on scikit library. The main idea was to predict the numbers written by persons and predict which number it is.

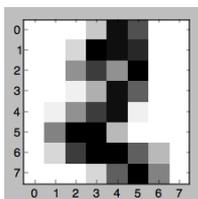


Figure 7 Digits dataset, number 2

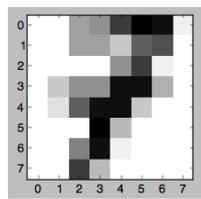


Figure 8 Digits dataset, number 7

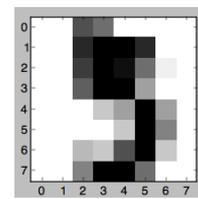


Figure 9 Digits dataset, number 5

As seen before, numbers are represented as matrix with size 8x8. They were obtained by digitalising random numbers written by persons. So, once many matrixes are generated, there are transformed in a unique line in order to allow the machine to learn since it does not accept matrixes. Apart from that, labels are assigned to each figure, in the previous cases:

- Figure 7: Label='2'
- Figure 8: Label='7'
- Figure 9: Label='5'

The concept explained before will be applied to the airspace generation, which means to find a way to represent as a matrix equally as the digits dataset.

Since sectors are not regular and in order to simplify the process, pseudo-random scenarios will be generated, and that means that the airspace will be represented as a grid with the same colour and the cells where the aircraft is located will be represented as a different coloured square, in values, no aircraft will be equal as a 0

Chapter 2. Modelisation of the problem

and aircraft will be represented as 1. Furthermore, conflict will occur if an airplane is adjacent with another one. The following matrixes represent these two very simple examples, one where conflict does not exist and the other one where the conflict exists.

- *NO CONFLICT*: $\begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \rightarrow \{001_100_001\}$

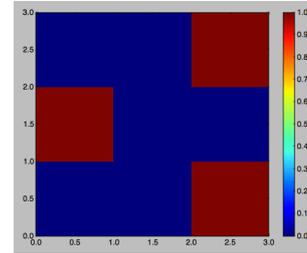


Figure 10 Example of 3x3 no conflict

- *CONFLICT*: $\begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \rightarrow \{011_000_001\}$

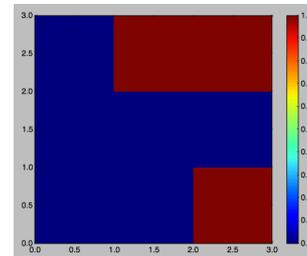


Figure 11 Example of 3x3 conflict

As seen before, since the library cannot understand matrixes, they must be linearized. Hence, each line is placed after the one that is before it.

In the following figures two different cases are represented, the first one is a grid, which represents a square airspace where 10 planes are placed pseudo randomly since it is a requirement for the first half of the cases to not be conflictive scenarios whilst the second half yes. The second figure represents the airspace in a hazardous situation since two or more aircraft are adjacent.

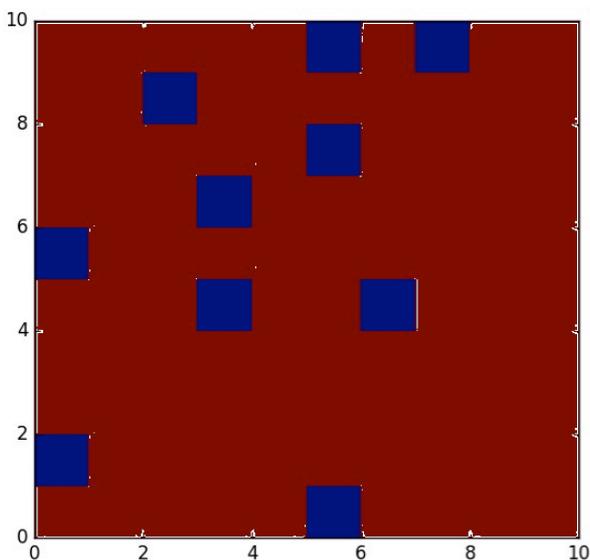


Figure 12 No conflictive situation for 10x10 matrix and with a capacity of 10 aircraft

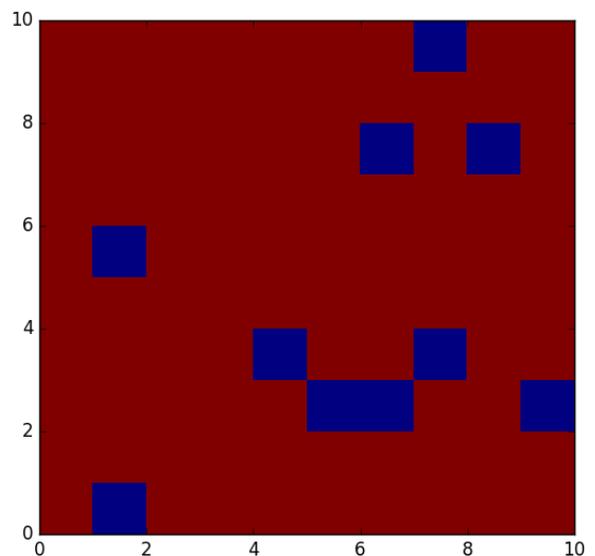


Figure 13 conflictive scenarios for 10x10 matrix with 10 aircraft capacity

Chapter 2. Modelisation of the problem

The learning process will be performed by the creation of a dataset of pseudo-random scenarios. Since it is interesting for the process of learning to have cases with conflictive and non-conflictive situations, during the process of creation of the dataset, half of the case population will be conflictive and the other half no. Once the set of learning cases has been created, new scenarios will be generated to test the accuracy of the model. In order to see how the size of the population dataset affects the model performance, tests will be executed by starting with a population of 1000 cases and will be incremented uniformly until a population of 35000 elements. For each case, the matrix dimension will be changed from 10x10 to 40x40 in order to see how the size affects the success ratio of the model.

Once the data-set has been created, the test cases will be created randomly, that means that before creating the case, a random value will be created, if it is confined between a range of values, it will be compulsory to be conflictive or not. The number of test cases will always be constant in relation with the learned cases. This can be translated by considering that from every 100 of learned cases, 2 random scenarios are generated to test how accurate the model is.

In conclusion, the main idea is to generate a huge dataset collection and once it is created, test the model with new cases to analyse how good is its performance and if any kind of consideration should be taken into account.

2.2.B NEST:

Back to NEST and its capabilities, there is a module that allows the user to cut the traffic and only select those flights that comply with a certain specifications imposed by it. Once the traffic is cut, it can be extracted and export it to a data file where each flight is shown by saving its route waypoints (latitude and longitude), the time of entry and exit on each route leg, and its course.

Since the time of entry and exit on each route leg are specified, the position¹ of each involved plane can be computed for every minute of the day by finding where the plane is located. It can be computed by finding which leg is on course for each flight of the day. Once the leg has been found, the time is converted to timestamp, and after that, as speed is considered constant along the leg, position can be found by the following simple calculus:

2

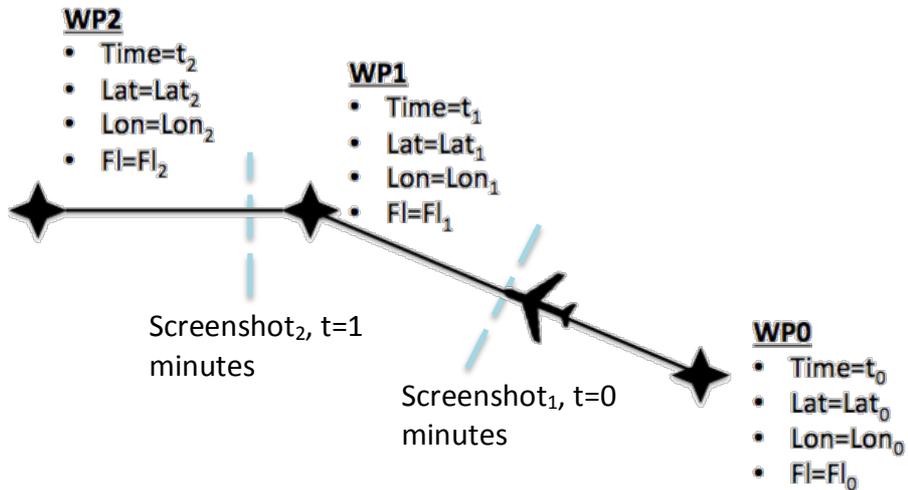


Figure 14 Screenshot generation

If the position has to be computed for the screenshot 1 for the flight number 1, the time of WPO is considered as equal as the time of WP1 since the time of the screenshot³ number 1 is between these two times.

The portion of elapsed time after the WPO is computed, hence, the intermediate latitude and longitude can be computed:

- $lat_{s1} = \frac{s_1 - t_0}{t_1 - t_0} \cdot (lat_1 - lat_0) + lat_0$
- $lon_{s1} = \frac{s_1 - t_0}{t_1 - t_0} \cdot (lon_1 - lon_0) + lon_0$

¹ A screenshot is a capture at a given minute of a day of all the airplanes. Identified

² WP is defined as a Waypoint of the each plane route, so at the end a flight route is defined by a succession of a numbered WP's.

Chapter 2. Modelisation of the problem

- $fl_{s1} = \frac{s_1 - t_0}{t_1 - t_0} \cdot (fl_1 - fl_0) + fl_0$

Once each position is computed for each evolved flight for each screenshot, the SO6 file extension, which is the one that the NEST generates with all the information of all the day, is extracted and sent to the conflict module. This will give the necessary information to decide which screenshot is conflictive and which not. At the end, each scenario will have an assigned label according to if it is conflictive or not. In this project, '0' will represent no conflict whilst '1' will represent conflict.

It is also important to consider that, if only a screenshot is considered for each minute of the day, resolution may be lost, here is an example:

Considering two aircrafts, one flies in the opposite direction to the other, the distance just before the next capture is set to be 5NM which is the minimum from security standards and the speed is set to 450kt, when the next screenshot is computed, the position would be the one as follows:

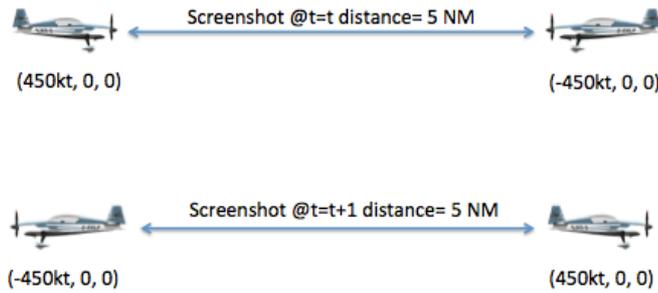


Figure 15 Diagram of a missed conflict due screenshot resolution

As explained before, the selected data will be chosen from the air traffic along a day, in this case, will be done by selecting the traffic of the 5th of September of 2015. Once it is done, the air traffic will be cut in order to only list those aircrafts that at a given moment passed through the Benelux airspace.

Since the selected way to make the machine learn if a scenario is conflictive or not, once the data is extracted different screenshots will be created, that is computing the exact position of each plane for each minute of the day, once the latitude and longitude are computed, will be represented in a line (as the key features) and for each line will be assigned with a label according to what the conflict module says, where 0 will be no conflict and 1 will be conflict

$$\begin{pmatrix} lat_{s1}^{f1} & lon_{s1}^{f1} & fl_{s1}^{f1} & \dots & lat_{s1}^{fn} & lon_{s1}^{fn} & fl_{s1}^{fn} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ lat_{sn}^{f1} & lon_{sn}^{f1} & fl_{sn}^{f1} & \dots & lat_{sn}^{fn} & lon_{sn}^{fn} & fl_{sn}^{fn} \end{pmatrix} = \begin{pmatrix} label_{s1} \\ label_{sn} \end{pmatrix}$$

Chapter 2. Modelisation of the problem

Where “ f ” represents the number of flight, and “ s ”, the number of screenshot. Considering that the number of screenshot is one per minute, that means that will be a total of 1440 screenshots per day, so there will be a maximum of 1440 labels.

It is also important to consider that flights don't last all the day, that means the first flight introduced in the matrix will not be the same, so that could affect the final performance of the model.

Also, the 4th of September and 6th of September will be used to make the machine learn, once it is done, 5th of September will be loaded and after generating the data file of each screenshot of the day, the forecast will be compared with NEST conflict data to see the performance.

Only a portion of the airspace is considered, in the following figure, the Belgium Airspace is shown:

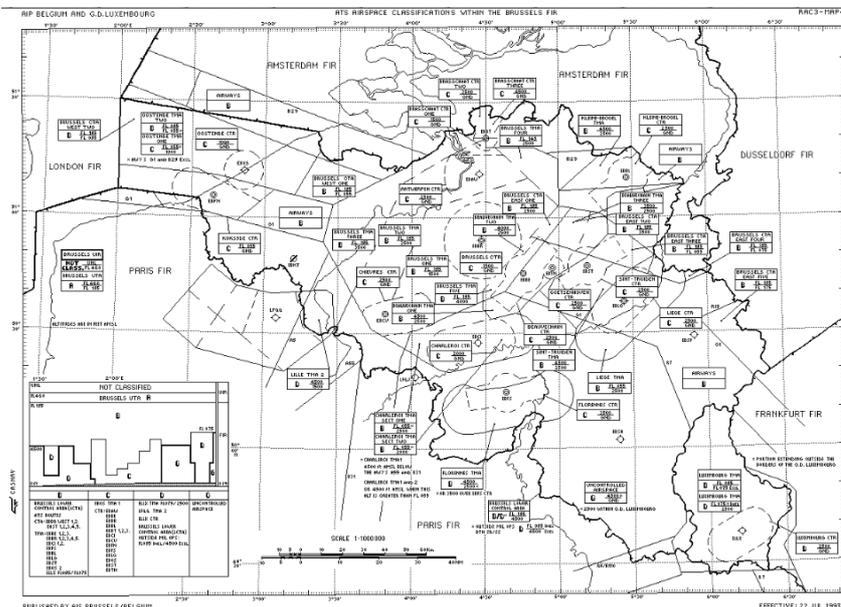


Figure 16 Belgium airspace, extracted from IVAO

Airspace will be simplified to a square, so the corners of the Belgium airspace are defined:

- Upper_left: 2.0 E, 51.6 N
- Upper_right: 6.5 E, 51.6 N
- Down_left: 2.0 E, 49.3 N
- Down_right: 6.5 E, 49.3 N

Here is an example of a screenshot list generated from the SO6 file:

- Line of control: Belgiumtrafficpunch
- Number of total screenshots: 1440
- Initial timestamp: 1441411200

Chapter 2. Modelisation of the problem

- Final timestamp: 1441497600
- Number of screenshot: newscreenshot: 131
- Number of evolved flights: 2
- Flight1 (n° flight, lat, lon, fl):
189845638.0,3051.75103531,247.612205857,110.0
- Flight2 (n° flight, lat, lon, fl):
189845312.0,3034.45778555,320.224290547,210.0

Once the screenshot “.txt” is generated, the conflict file is created from the SO6 file. In order to find where are the conflicts happening, a loop sequence will go step by step on each screenshot and find out if on the conflict file if there is anything wrong during that minute of the day, so at the end the feature vectors and the labels will be generated.

Here is an example of a screenshot painted with Python matplotlib library:

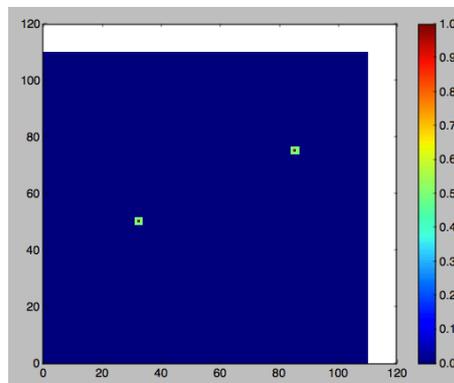


Figure 17 Screenshot n° 131

In this case only two flights are painted since there are only two flights involved during the screenshot number 131.

It is really important to consider that since the security standards must be ensured, that means to comply with a minimum distance of 5NM during en-route operations and up to 3NM during terminal operations. Since the available data corresponds to flights during route operations, only 5NM will be considered. Also it is important to take into account that the resolution of the square has to be chosen in order to ensure that visualisation can be performed.

In the example above, each flight is represented by a 3x3 pixel square. If a 3x3 square corresponds to a 5NM, and knowing that Belgium, which can be included in a square of 160X160NM, the amount of necessary pixels needed to represent this country are 90 units, in order to round it 100 units will be considered to represent the airspace.

Once the labels were loaded, and since the idea is to predict and not detect the conflict during the same moment they are happening just detect in advance to give the ATC the enough amount of time to consider which is the best resolution of the conflict. That is why, labels were modified and inside the 5 following minutes a conflict was about to happen, the label was changed from 1 which meant conflict to 2 which means that during the following 5 minutes a conflict is about to happen, see the table above to understand how labels are modified in order to predict conflicts.

Screenshot	Time to conflict	Labels (before):	Labels (Now):
131	5 min	0	0
132	4 min	0	2
133	3 min	0	2
134	2 min	0	2
135	1 min	0	2
136	0 min	1	1
137	Null	1	1
138	Null	0	0
139	Null	0	0

Table 1 Screenshot labelling with/without forecast

After loading each feature line and each label to the model and executing many tests, all with slightly differences, it demonstrated fair clear that it can not be used this model.

The results shown that or the system was always telling that conflict was happening when it was not true, or totally in the other way, it was always telling that no conflict was happening. The reasons of why conflicts were no detected may be the following ones:

1. Not enough **number of learning conflicts** scenario
2. **Forecast not useful**
3. **Data not consistent** due its position on feature vector
4. **Not enough data** to find patterns
5. **Many air traffic**, other zone may be considered

That is why, considering real data to test a predictive model can be really complicated, since in this case, only a few cases that were conflictive were taking into account, so the portion of number of conflictive screenshots compared to the number of non-conflictive screenshot was really slow. This fact explains why the ratio of learning cases should be divided at least uniformly or something similar.

CHAPTER 3. PERFORMANCE AND RESULTS OF THE MODEL

3.1 Model performance and results

First of all, the following figures correspond to the analysis of the ratio of hits from every situation. Conflict and non-conflictive scenario were generated for different sizes of population with the model of random Modelisation of scenarios to see its performance.

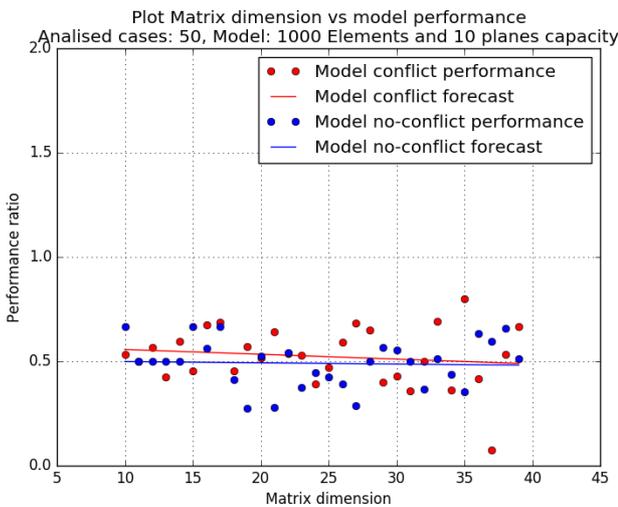


Figure 18 Model of 1000 elements

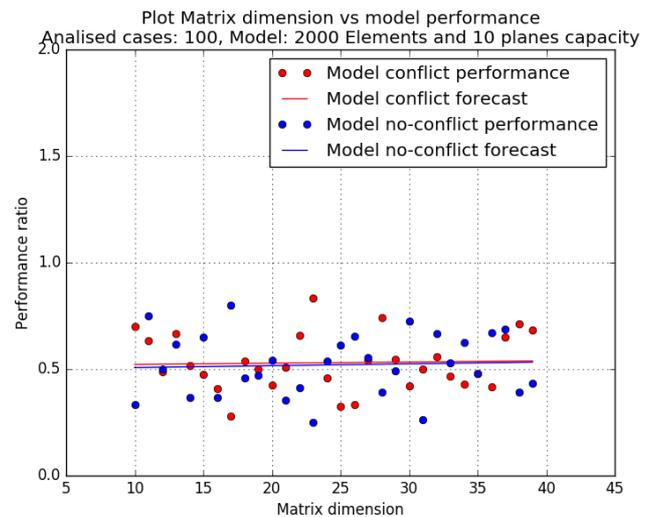


Figure 19 Model of 2000 population elements

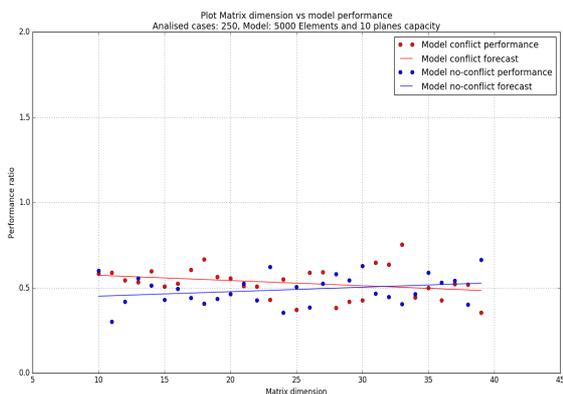


Figure 20 Model of 5000 population elements

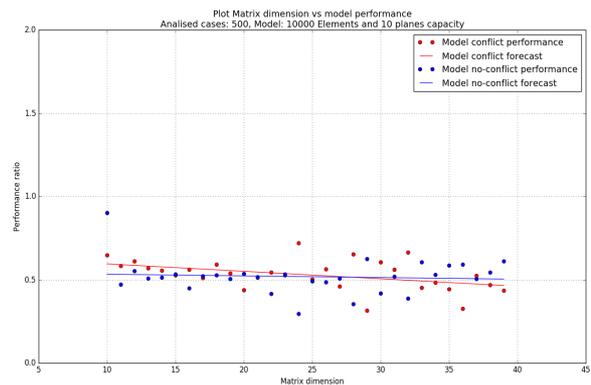


Figure 21 Model of 10000 population elements

As seen in this figures, the trend of the forecast for bigger matrix dimension is to decrease. It is really important to consider that even the trend the minimum squared line is more or less constant, the model accuracy decreases as the size of the dimension increases. Since the number of learned cases is always constant along the

Chapter 3. Performance and results of the model

algorithm execution, that means constant whilst the size of the matrix is increased, the number of possible combinations where planes can be allocated increases with as an exponential relation. Because of that, the amount of learning cases should be increased also as an exponential relation, but, as the user before the algorithm execution fixes the learning cases, a loss of accuracy occurs.

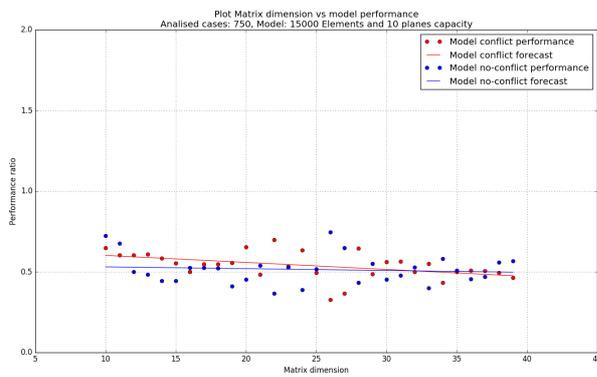


Figure 22 Model of 15000 population elements

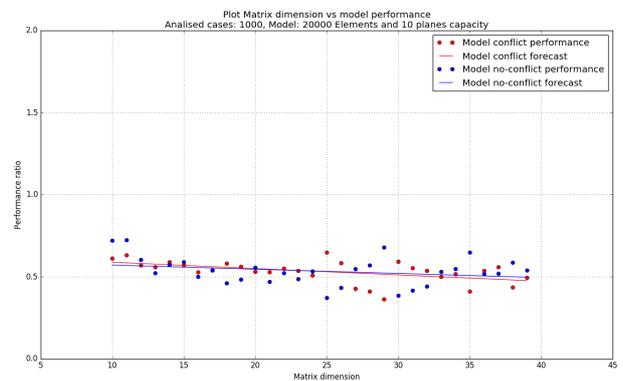


Figure 23 Model of 20000 population elements

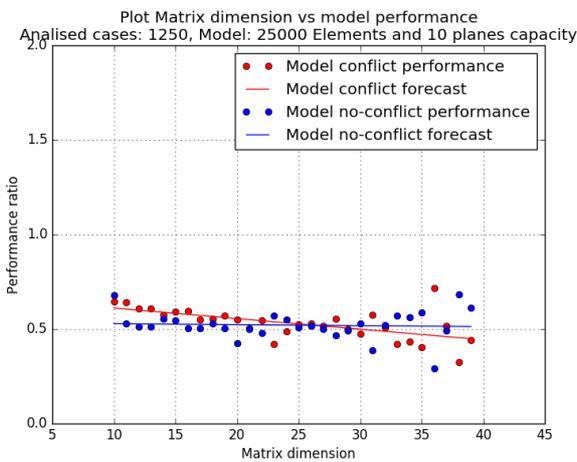


Figure 24 Model of 25000 population elements

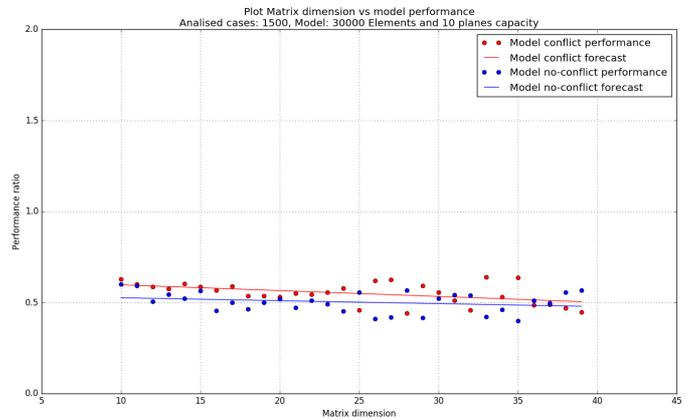


Figure 25 Model of 30000 population elements

Chapter 3. Performance and results of the model

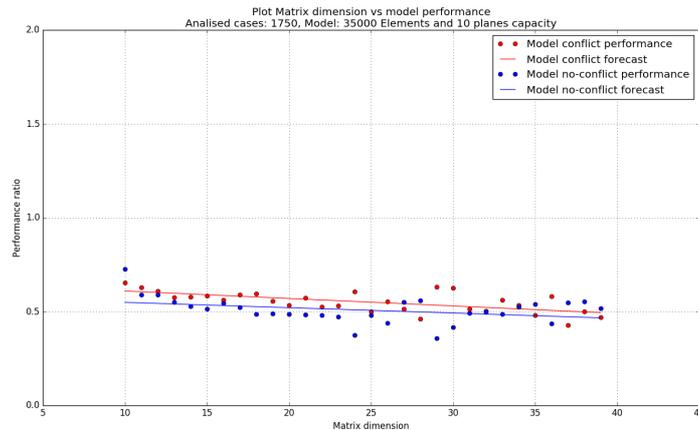


Figure 26 Model of 35000 population elements

If the different permutations of placing the aircrafts on the different available positions are computed for each size of the matrix, the following plot is obtained:

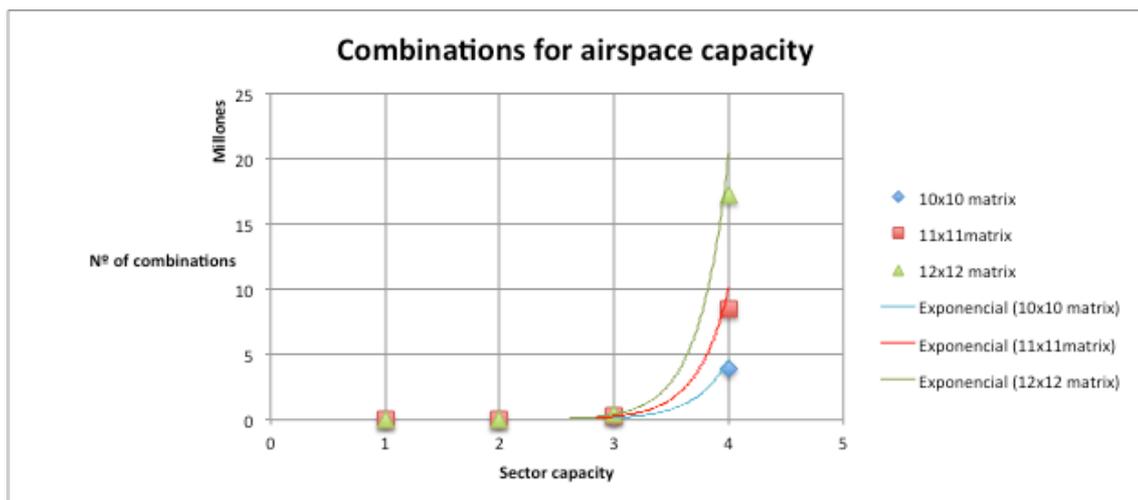


Figure 27 Number of combinations for different matrix sizes and different sector capacities

As seen, increasing the matrix size is traduced to increasing the number of different combinations where the planes can be. Since the model is generating pseudo-random scenarios without any kind of rule, the number of possibilities of placing the planes is really huge. That explains why the model for small sizes has a better performance, whilst for big sizes; the model does not work properly because it needs more learning situations to predict new cases.

Chapter 3. Performance and results of the model

As seen, the general trend for the model prediction is to be more or less constant but decreasing with matrix dimension, that could be due the amount of probabilities of placing ones along a matrix, as the size is incremented there are many cases that the machine could not have seen before.

In this type of situations and since the field of study, time is everything since it is important to see how fast the model can be created, as equal as valorate if it is a good solution. That is why, the time of execution for each type of situation will be considered and once it has been computed, the following plot shows how time increases with more population elements.

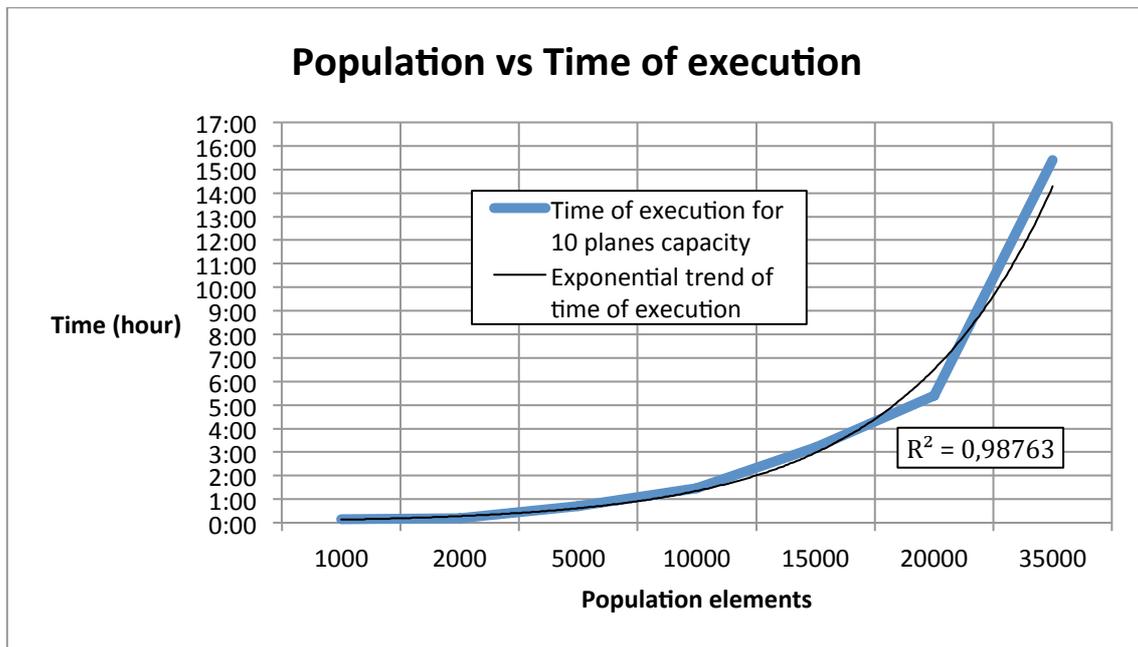


Figure 28 Execution time for different population elements

As seen before, the time of execution for bigger dimension elements grows with an exponential relation. This happens since the way the program computes everything, increases in this way.

Elements	Elapsed time
1000	0:09:00
2000	0:12:00
5000	0:43:00
10000	1:28:00
15000	3:11:59
20000	5:24:00
35000	15:23:59

Table 2 Execution times

Chapter 3. Performance and results of the model

Also it is important to consider that the time of process can differ between each case time and its value compared with the exponential relation trend. This happens since the processes that the computer had in progress by the time the python script was executed may be different, increasing or decreasing the workload, and making the program run faster or slower.

The obtained plots show that generating pseudo-random scenarios without any kind of rule, make the model tough and with a grade of predictability that cannot be accepted according to safety standards in aviation, if airways are loaded to the model, that could help the system to be simplified, and, as a result, obtain a higher match ratio.

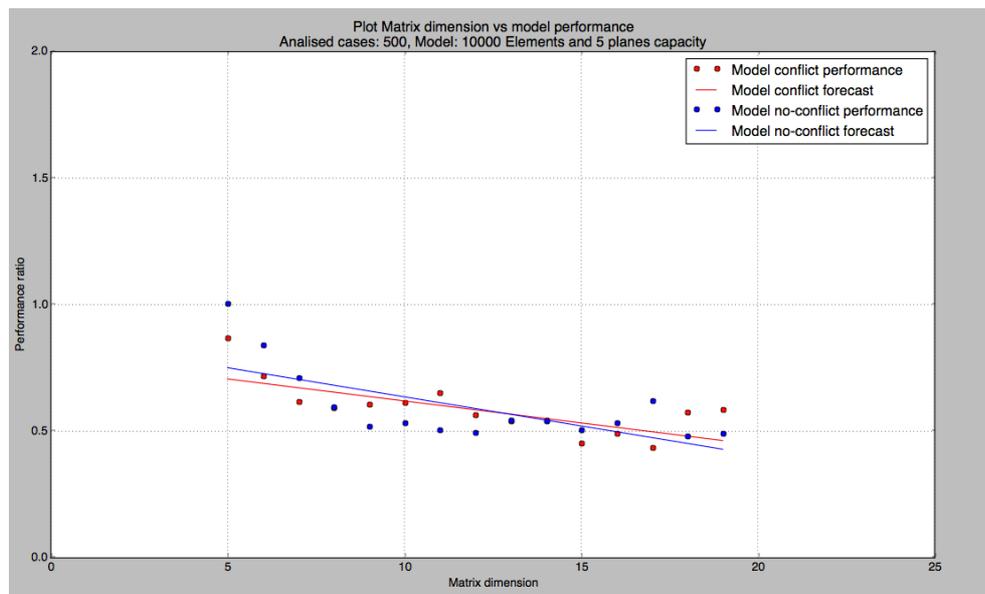


Figure 29 Model of 10000 elements, 5-airplane capacity

In the previous plot, lower dimension of matrix was considered as equal as lower airspace capacity, now lets imagine that on the matrix, the place where the aircrafts will be placed now it's going to be limited to a few positions. That helps to imagine that flights can only be confined inside some airways or some positions. To ensure and give more reality to the project, imaginary airways will be considered, and only some flights according to the sector capacity will be generated and placed somewhere along the airways.

The cases will be the following ones:

- **CASE 1:** Simple intersection with 1 node
- **CASE 2:** Intersection not perpendicular with 1 node
- **CASE 3:** Displaced intersection with airway crossing, 3 nodes
- **CASE 4:** Double horizontal intersection and airway not perpendicular, 5 nodes

Chapter 3. Performance and results of the model

The simple examples will be the ones as follows:

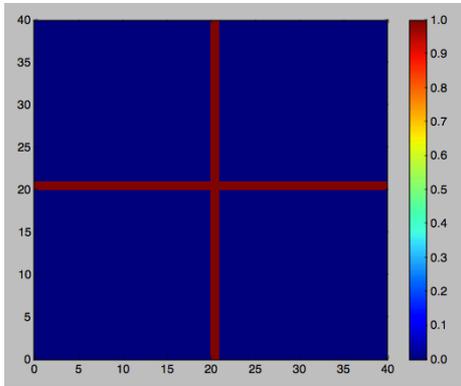


Figure 30 Airway configuration 1 with 1 node

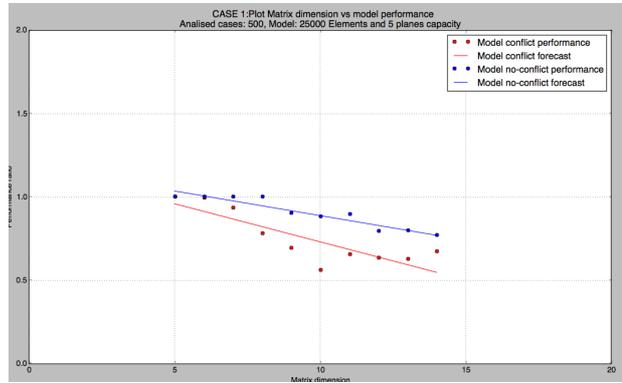


Figure 31 Model performance for 1 airway nodes

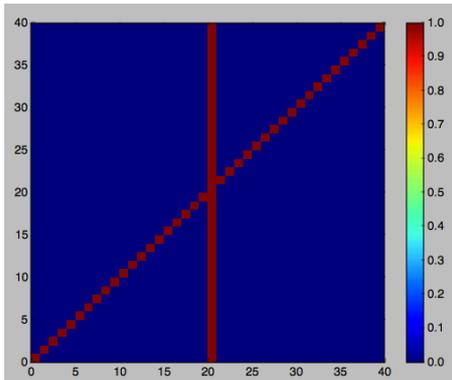


Figure 32 Airway configuration 2 with 1 node

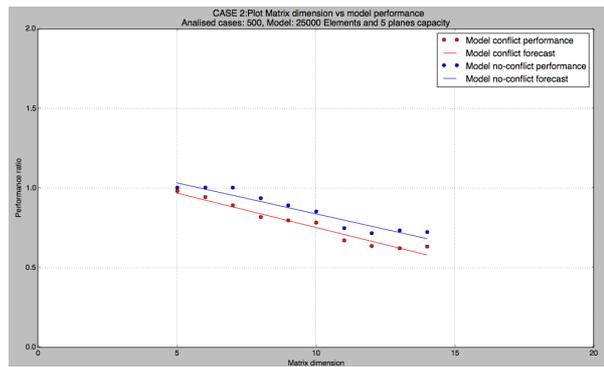


Figure 33 Model performance for 1 airway node, 5 aircraft capacity

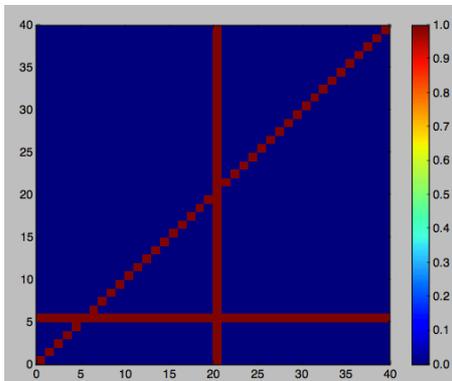


Figure 34 Airway configurations with 3 nodes

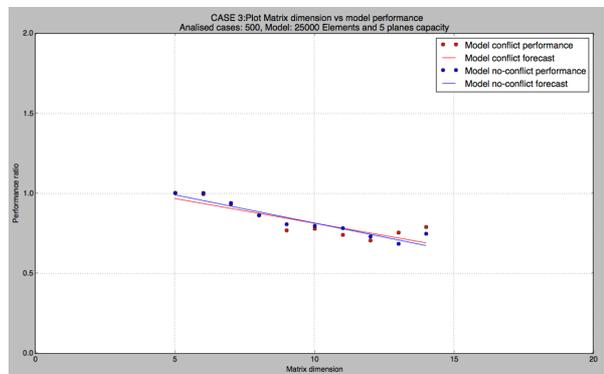


Figure 35 Model performance for 3-airway node, 5 aircraft capacity

Chapter 3. Performance and results of the model

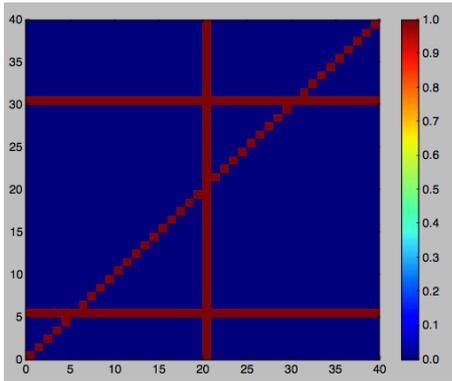


Figure 36 Airway configuration with 5 nodes

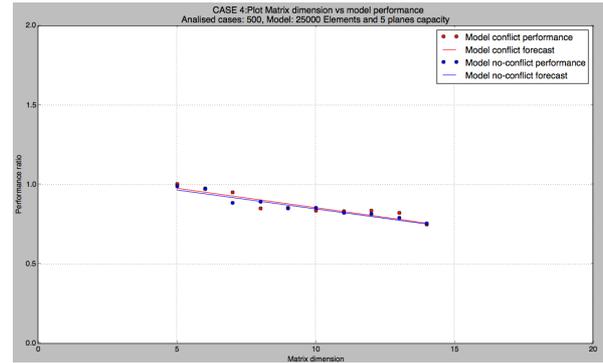


Figure 37 Model performance for 5-airway node, 5 aircraft capacity

As seen, the behaviour in all cases is more or less the same, that is to say that, for small sizes, the algorithm works really well whilst if the size of the matrix is increased, the system loses predictability and accuracy.

That is explained by the same explanation as mentioned before, small sizes means less possible combination where placing the flights, whilst increasing the matrix, is traduced on a increase in the airways length, hence the workload of the ATC's since they have to monitor the flights during more time. That is why, better accuracy values of the ratio of success for conflictive and non-conflictive situations are obtained for small sizes whilst they are decreased proportionally equal for any number of nodes with the increase of the matrix.

In addition, the performance of the model may be improved by pre-processing the data mapping. This could be performed by finding which is the centre of mass of the set of aircrafts. Deduced by their actual position and considering that all of them are punctual mass of unit 1, the centre is found.

This pre-processing procedure could help to normalize all cases and as a result improving system accuracy since lower number of necessary cases would be needed to get the same accuracy compared with the procedure without the pre-processing part of the image.

Chapter 3. Performance and results of the model

On the other hand, NEST data was included inside the study to see if real cases could be taken into account to make predictions from real life. As explained, the Benelux upper airspace is considered since its configuration and proximity to busy airports like London-Heathrow, Paris Charles de Gaulle and Berlin-Tegel.

Traffic passing through that airspace zone was considered, and with the *“.SO6”* file from NEST, and after loading the constrains of which sector will be considered (in this case EEBUCTA), a new *“.SO6”* file is generated as equal as a *“.conf”* file where each conflict happening inside that sector is specified.

The method to create the model has been the one explained in previous chapters, where, for each minute, the position (latitude, longitude and flight level) has been computed. After that, for each capture of each minute, every relevant data regarding the position of the planes was placed on a unique line as a feature vector. Later, each feature vector was assigned with a label, obtained with the *“.conf”* file where each conflict is specified as equal as the initial time and the final time.

The following tests were performed:

- A. Loading 5th of September
- B. Loading 4th, 5th and 6th of September

A.

This test was performed by loading a part of the day as a learning process, and the other part of the day was considered as being the part of the test process. Results shown huge ratios of performance since the model reached levels of a 100% of hits regarding only no conflictive cases. This fact did not give us any kind of conclusion. This happens since some conflicts may be needed during the part of the test towards being able to see if it works or not. When loading a conflictive case inside the test area, this means repeating one case in the model and after executing the test, each repeated conflict was labelled as a conflictive, keeping a 100% hit ratio.

Even so, this 100% of hit ratio was not true since new cases not seen before the machine would be needed in order to see if the model predicts conflictive situations.

B.

In order to test if the model works, 2 days were loaded and one totally different to test if at the end it is working or not. After some tests, and trying to change things such as the order of planes, or the limit of capacity of each line of features, we can conclude that any conflict was detected, at least new ones. What happened is that the model always was telling that there was not any kind of conflict, thus, all no conflict situations were labelled with the correct one. This makes us understand that model does not work.

CHAPTER 4. CONCLUSIONS AND FURTHER CONSIDERATIONS

After all the different tests and different ways to generate scenarios in order to test how good applying Machine Learning to air traffic conflict detection can be to enhance the actual system the following conclusions and statements are proved:

First of all, Machine Learning has demonstrated to be a really good way to automate many things in real life, from predictions of a product demand to twitter sentiment analysis. In this world where knowledge is everything, using the big data concept and taking profit from it, which refers to data mining, could ensure that on the years to come, airspace security is maintained by enhancing the actual system.

Secondly, even having made many assumptions as considering airspace regular, or speed constant along the route leg between waypoints, this tool should be considered of being applied in some how, not only in air traffic conflict detection, but also in different aspects of aviation like preventive maintenance to save money or the movement not only on airborne operations but also on platform movements.

On the one hand, airspace conflict detection is something really hard to do since security standards are really rigorous and restrictive, that is why, and taking into account the results obtained for free-flight matrixes, it is demonstrated that the system does not work as good as it should for vast extensions of airspace. At least, the system shown that if airplanes are placed randomly, the number of combinations where aircrafts could be its huge, hence the system needs an infinite number of inputs to generate the model correctly.

In contrast, this could be solved by choosing the best cases to take into account since the flight operations plan of an airliner is more or less repetitive, and that could help to detect conflictive patterns and predict when is going to happen another one.

On the other hand, the system works quite well for small sizes of matrixes since the number of available combinations is not as big as before. This fact demonstrates that Machine Learning is something that could be applied to airspace conflict detection but only in a special zones like hotspots where intersection of important airways are.

In addition, what is clear is that the model should be updated constantly from historic data since the operation of an airliner changes seasonally, that is the reason why if the model has only taking into account the last two weeks of operation during summer,

Chapter 4. Performance and results of the model

predictions may not be good as it should when winter approaches due the seasonality of the airliner, or maybe is not the case and it's a company flag which the number of flights remains more or less constant along the year.

In conclusion, airspace conflict detection is a process that is really difficult to perform, at least by using only predictive models, since they are not as accurate as geometrical methods are. Even so, this does not mean that it cannot be used to enhance part of the process of detecting conflicts. It may not be used exclusively to detect airborne conflicts but also it could be used on ground operations where the weather makes difficult to place where the planes on the platform are, or in places that are really repetitive due its configuration, or conflicts on queues to take off.

It is important to consider that the key of Machine Learning is the process of choosing which are the best key features that define each situation just to make more understandable the problem to the machine.

As mentioned before, the trend of air navigation is to fly under free-flight rules due benefits it would bring like the reduction of the cost of fuel and as a result the decrease on the emitted pollutants, that is why it is so important to develop new and efficient tools to be prepared for the years to come.

CHAPTER 5: FURTHER STUDIES AND INVESTIGATIONS

Due its complexity, this project was focused on define which could be the basis to start a study process to determine how Machine Learning could be used on air traffic conflict detection to improve the actual ATMS that exists today to ensure that for 2030 air traffic capacity would be the necessary one to harbour the traffic demand.

As seen, the topic is vast and extensive and that is why, if a system has to be developed, there are many phases of implementation, from the basis, to the state-of art.

In this project, many time was spent on try to make the model functional only by taking into account real data obtained from NEST, but, as the obtained results were not the ones expected and since the amount of available time as equal as the amount of available computational power was limited, all the project was redesigned towards obtain something functional which could be as real as possible. That is why, random scenarios had been created in contrast of getting the information directly from NEST and the same happened with how the airspace was represented by the system. Since the same learning process had been used before on hand-written detection for determine the numbers, the same idea was applied to represent the airspace as an image with placing 0 where there was no airplane and 1 where it was.

It is fair clear that this is just the introduction to something that could be applied on the near future to enhance the actual system, even so, if the line of investigation has to be continued many considerations and recommendations should be taken into account to plan the best way to develop the tool:

- Start from the very basics
- Limit the study range
- Explore different scenarios, not only enrout

To develop this project, first of all, the main idea should be tested with data, which covers a full range of different possible cases where the model must deal with it.

After designing the basis of the study, analysis should be made with real data, that is to say to get data from NEST and find out if the conflicts obtained with one method match with the other ones.

BIBLIOGRAPHY

BIBLIOGRAPHY

- [1] AENA. *ENAIRES (Navegación aérea)*. www.enaire.es (último acceso: 15 de 10 de 2016).
- [2] Alam, Sameer, Kamran Shafi, Hussein A. Abbass, y Michael Barlow. *An ensemble approach for conflict detection in Free Flight by data mining*. Canberra: Elsevier, 2007.
- [3] Ball, M. *Total delay impact study: A comprehensive Assessment of the Cost and Impacts of Flight Delay in the United States*. Nextor, 2010.
- [4] EUROCONTROL. *Challenges of growth 2008*. Statistics and Forecast Service, EUROCONTROL, Eurocontrol STATFOR, 2008.
- [5] Garreta, Raul, y Guillermo Moncecchi. *Learning sci-kit learn: Machine learning in Python*. Birmingham: Packt Publishing Ltd., 2013.
- [6] Hackeling, Gavin. *Mastering machine learning with Sci-kit learn*. Birmingham: Packt Publishing Ltd., 2014.
- [7] Honore, Carl. *La lentitud como método*. RBA, 2013.
- [8] IATA. *International Air Transport Association*. 2016. www.iata.org (último acceso: 6 de 10 de 2016).
- [9] IVAO. *IVAO*. www.IVAO.com (último acceso: 23 de 10 de 2016).
- [10] Kuchar, James K., y Lee C. Yang. «A Review of Conflict Detection and Resolution Modeling Methods.» *Transactions on Intelligent Transportation Systems*, Massachusetts Institute of Technology, Cambridge, 2000.
- [11] NATCA. *National Air Traffic Controllers Associations*. 2016. www.natca.org (último acceso: 6 de 10 de 2016).
- [12] Prandini, Maria, John Lygeros, Arnab Nilim, y Shankar Sastry. *Randomized algorithms for probabilistic aircraft conflict detection*. Berkeley, CA: University of California, 1999.
- [13] SCIKIT Learn. *SCIKIT Learn*. 2016. <http://scikit-learn.org> (último acceso: 19 de 10 de 2016).
- [14] SKYBRARY. *Skybrary*. www.skybrary.com (último acceso: 15 de 10 de 2016).

ANNEXES

Annex 1: Code of simple figures

```
import numpy as np
import random
from sklearn import svm
__author__ = 'marcperezfont'
import statsmodels.api as sm
import matplotlib.pyplot as plt
from time import gmtime, strftime
from scipy.optimize import curve_fit
clf = svm.SVC()
#Aquest script generarem figures aleatòries per tal de comprobar la efectivitat del sistema.
#Generarem 1000 situacions aleatòries i assignarem conflicte o no en funció de la posició dels 1's

#Analitzare per a diferents mesures de matriu per veure la tendència de tasa d'encerts i la tasa de falsos positius
print(strftime("%Y-%m-%d %H:%M:%S", gmtime()))

lim_inf=3
lim_sup=7
#Genero els vectors on incloure la tasa d'encerts i de falsos positius
tasa_hits=np.zeros((1,lim_sup-lim_inf))
tasa_hitsnoconflict=np.zeros((1,lim_sup-lim_inf))
coordenadesy=np.zeros((1,lim_sup-lim_inf))
point=0
#data_image=np.zeros(lim_inf,lim_inf)
#Recorro desde el limit inferior fins el limit superior, anire canviant la dimensio de x i de y
for s in range(lim_inf,lim_sup):
    print(s)
    number_of_solutions=100000#Solucions que es pasaran al model per fer el predict
    tasa_conflictes_noconflictes=0.5#Percentatge de solucions amb/sense conflicte
    nombre_no_conflictes=int(number_of_solutions*tasa_conflictes_noconflictes)#Nombre de solucions que no son conflictes
    #nombre_conflictes=int(number_of_solutions*tasa_conflictes_noconflictes)#Nombre de solucions que son conflicte
    percentage=0.95#Percentatge de aprenentatge/predict
    counter=int(number_of_solutions)
    dimensionx=s#Dimensió de matriu x
    dimensiony=s#Dimensio de matriu y
    number_flights=2#Capacitat del sector
    flight_data=np.zeros((int(number_of_solutions*(1+(1-percentage))),dimensionx*dimensiony))#Inicialitzo el vector de matrius on guardare els casos
    flight_data_solution=np.zeros(int(number_of_solutions*(1+(1-percentage))))#Inicialitzo el vector de solucions si es o no conflicte
    i=0
    while (i<(int(number_of_solutions*(1+(1-percentage))))):#Fins que no esta ple no paro de generar situacions (tant aprenentatge com predict)
        #print(i)
        #Dades de situació inicial
        flights_per_sector=number_flights#Poso de nou tots els avions que poden haver en un sector
        number_of_zeros=(dimensiony*dimensionx)-flights_per_sector#Calculo el nombre de 0 per a cada matriu
        case=np.zeros((dimensionx,dimensiony))#Ara tinc la situació de ningun vol, afegim de manera aleatoria vols a la situació
```

```

for j in range(0,dimensionx):
    for k in range(0,dimensiony):
        #Tenint en compte els vols maxims per sectors he de buscar un metode que posicioni els 1 de
        alguna manera logica
        #En aquest cas simplement calculo una distribució uniforme i vaig col·locant els 1 fins que ja no
        hi caben mes avions al sector
        probability_of_zero=number_of_zeros/(number_of_zeros+flights_per_sector)
        probability_of_one=flights_per_sector/(number_of_zeros+flights_per_sector)
        pointer=random.uniform(0,1)
        if pointer<=probability_of_one:
            #Estem en el cas de que es un 0
            case[j,k]=1
            flights_per_sector=flights_per_sector-1
        else:
            number_of_zeros=number_of_zeros-1
            #case[j,k]=0
        #Ara ja tinc pintada una situació, la guardare en el vector de solucions sempre i quan hagi omplert
        primer els casos amb conflicte i despres els que no
        conflicte=0#En principi no hi ha conflicte, busco si existeix
        for j in range(0,dimensionx-1):
            elements_x0=case[:,j]#Agafo per columnes
            elements_x1=case[:,j+1]#Agafo els elements de la següent columna
            sum_of_columns=elements_x0+elements_x1
            for k in range(0,len(sum_of_columns)-1):
                total=sum_of_columns[k]+sum_of_columns[k+1]#Miro la suma del primer element de
                columna amb el següent per veure si suma mes que 1
                if(total>1):
                    #Estem en el cas en que hi ha conflicte
                    conflicte=1

        #Tinc la suma dels elements
if(i<number_of_solutions):
    if((i<nombre_no_conflictes)&(conflicte==0)):#Casos d'aprenentatge que no tenen conflicte
        flight_data[i]=np.resize(case,(1,dimensionx*dimensiony))#Ho paso tot en una fila
        #Parte coloritos Adri
        i=i+1

    else:
        if((i>=nombre_no_conflictes)&(conflicte==1)):#Casos que tenen conflicte
            flight_data[i]=np.resize(case,(1,dimensionx*dimensiony))#Ho paso tot en una fila
            flight_data_solution[i]=1
            i=i+1

        #case=case*0.75
        #for j in range(0,dimensionx):
        #    for k in range(0,dimensiony):
        #        if(case[j,k]==0):
        #            case[j,k]=1
        #print(case)
        #airsp=plt.pcolor(case)
        #plt.colorbar(airsp)
        #plt.show()

else:#Aquests seran les solucions que mirare en el predict
    #Generare de manera aleatoria solucions amb i sense conflicte
    existeix_conflicte=random.randint(0,1)#Aleatori, o 0(No conflicte) o 1(Conflicte)
    if(existeix_conflicte==conflicte):#El següent es tracta de que sigui no conflicte
        flight_data_solution[i]=conflicte
        flight_data[i]=np.resize(case,(1,dimensionx*dimensiony))#Ho paso tot en una fila

```

```

i=i+1#Nomes incremento si ha coincidit que l'aleatori coincideix amb el cas generat

print('Executing model')
from sklearn import svm
clf = svm.SVC()
#This script loads the statistics for each screenshot

#-----Sckit learn-----

clf.fit(flight_data[:number_of_solutions], flight_data_solution[:number_of_solutions])
predicted = clf.predict(flight_data[number_of_solutions:])
casos_analitzats=number_of_solutions*(1-percentage)
encerts=0
conflictcs=0
hit=0
missed=0
falsos=0
no_conflictcs_hit=0
no_conflictcs=0
for p in predicted:

    if (p==flight_data_solution[counter]):#Si es el mateix cas, conto un encert
        encerts=encerts+1#Encert tant 0 com 1
        #CAS 1: Conflictc real i s'ha predit el conflictc
        if(flight_data_solution[counter]==1):
            hit=hit+1#Cas en que el sistema ha encertat un conflictc
            conflictcs=conflictcs+1#Conflictc real
            #CAS 2: No conflictc real i s'ha predit correctament
        else:
            no_conflictcs=no_conflictcs+1#Afegeixo un no conflictc ja que no es '1'
            no_conflictcs_hit=no_conflictcs_hit+1#Afegeixo un encert com a no conflictc
        else:#Estic en els casos en que no s'han encertat
            #CAS 3: S'ha predit conflictc pero en realitat no ho es
            if(p==1):
                falsos=falsos+1#Casos en que el sistema no coincideix amb el real
                no_conflictcs=no_conflictcs+1#Afegeixo un no conflictc ja que en realitat no ho es
            #CAS 4: No s'ha predit conflictc pero en realitat ho es
            else:
                conflictcs=conflictcs+1#
                missed=missed+1#Sumo els no encerts
        counter=counter+1

tasa_hits[0,point]=hit/conflictcs
tasa_hitsnoconflict[0,point]=no_conflictcs_hit/no_conflictcs
coordenadesy[0,point]=s
point=point+1

#Ara ja tinc les estadistiques totals
print('Coordenades y')
print(coordenadesy[0])
print('tasa encerts conflictius')
print(tasa_hits[0])
print('tasa encerts no conflictc')
print(tasa_hitsnoconflict[0])

#Tasa hits
y1=tasa_hits[0]
y2=tasa_hitsnoconflict[0]
x=(coordenadesy[0])

```

```

z1 = np.polyfit(x, y1, 1)
z2 = np.polyfit(x, y2, 1)
f1 = np.poly1d(z1)
f2 = np.poly1d(z2)

# calculate new x's and y's
x_new_hits_conflict = np.linspace(x[0], x[-1], 50)
#x_new_hits_noconflict = np.linspace(x[0], x[-1], 50)

y_new_hits_conflict = f1(x_new_hits_conflict)
y_new_hits_noconflict = f2(x_new_hits_conflict)

#Averiguo l'error vertical dels punts amb la recta que correspon al forecast, d'aquesta manera pintare
les barres d'error per a cada cas.
#print(z1)
print(f1)

plt.plot(x,y1,'o',label='Model conflict performance',color='red')
plt.plot(x_new_hits_conflict, y_new_hits_conflict,label='Model conflict forecast ', color='red')
plt.plot(x,y2,'o',label='Model no-conflict performance',color='blue')
plt.plot(x_new_hits_conflict,y_new_hits_noconflict,label='Model no-conflict forecast', color='blue')

plt.xlim(lim_inf-5, lim_sup+5)
plt.ylim(0,2)
plt.title(u'Plot Matrix dimension vs model performance\n'
          u'Analised cases: %d, '
          u'Model: %d Elements and %d planes capacity'%((int(number_of_solutions*(1-
percentage))),number_of_solutions,number_flights))
plt.xlabel(u'Matrix dimension')
plt.ylabel(u'Performance ratio')
plt.legend()
plt.grid()
print(strftime("%Y-%m-%d %H:%M:%S", gmtime()))
plt.show()

#Pintem les estadistiques:
print('Estadistiques')
print('Casos analitzats:')
print(int(number_of_solutions*(1-percentage)))
print('Total percentatge encerts en conflictes:')#Nombre d'encerts tant 0 com 1 del total de casos
print(hit/conflictes)
print('Total percentatge encerts en no conflicte')
print(no_conflictes_hit/no_conflictes)
print(strftime("%Y-%m-%d %H:%M:%S", gmtime()))

```