Conflict Detection Module

Deliverable ID:	**D3.2**
Dissemination Level:	**PU**
Project Acronym:	**AISA**
Grant:	**892618**
Call:	**H2020-SESAR-2019-2**
Topic:	**SESAR-ER4-01-2019**
Consortium Coordinator:	**FTTS**
Edition date:	**10 February 2021**
Edition:	**00.01.00**
Template Edition:	**02.00.02**

Founding Members

EUROPEAN UNION EUROCONTROL

SESAR JOINT UNDERTAKING

Authoring & Approval

Authors of the document

Name/Beneficiary	Position/Title	Date
Javier A. Pérez Castán/UPM	Assistant Professor	17 February 2021

Reviewers internal to the project

Name/Beneficiary	Position/Title	Date
Luis Pérez Sanz/UPM	Lecturer	23 February 2021
Lars L. Schmidt/TU Braunschweig	Research Engineer	24 March 2021
Tomislav Radišić/FTTS	Assistant Professor	25 March 2021

Approved for submission to the SJU By - Representatives of beneficiaries involved in the project

Name/Beneficiary	Position/Title	Date
Javier A. Pérez Castán/UPM	Assistant Professor	30 March 2021
Tomislav Radišić/FTTS	Assistant Professor	30 March 2021
Lars L. Schmidt/TU Braunschweig	Research Engineer	30 March 2021

Rejected By - Representatives of beneficiaries involved in the project

Name/Beneficiary	Position/Title	Date

Document History

Edition	Date	Status	Author	Justification
00.00.01	17/02/2021	Draft	Javier A. Pérez Castán	New document
00.00.01	26/03/2021	Draft	Javier A. Pérez Castán	Initial draft
00.01.0	30/03/2021	Final Draft	Javier A. Pérez Castán	Final draft

Copyright Statement

AISA

AI SITUATIONAL AWARENESS FOUNDATION FOR ADVANCING AUTOMATION

This deliverable is part of a project that has received funding from the SESAR Joint Undertaking under grant agreement No 892618 under European Union's Horizon 2020 research and innovation programme.

Abstract

The Conflict Detection module document is a deliverable in the work package 3 "Development of Machine Learning (ML) modules" of the AISA project. This document is similar to other work package deliverables and develops a conflict detection module based on ML techniques for the AISA project.

This work deals with monitoring tasks focusing on situational awareness. This approach applies ML techniques to perform predictions about separation infringements and safety metrics associated with the intrinsic characteristics of the separation between an aircraft pair. Herein, this work focuses on the concept of Situation of Interest (SI). One SI is when an aircraft pair is expected to intersect with a horizontal separation lower than a pre-defined separation and infringe the vertical separation minima. The safety metrics are the Minimum Distance, the distance and the time to reach the Minimum Distance for each aircraft pair. Moreover, it has been developed two approaches with similar roles of the Air Traffic Controller's (ATCO) team. The Static mode focuses on planner ATCO. This mode predicts SI and their safety metrics when an aircraft pierces into the airspace with the aircraft located within the airspace. The Dynamic mode focuses on tactical ATCO. This mode predicts SI and their safety metrics throughout the aircraft's evolution within the airspace, and it receives the 4DT prediction of the aircraft within the airspace.

ADS-B trajectories have been extracted from The OpenSky Network and constituted the basis for the ML database. This database of ADS-B trajectories has also been used as 4DT predictions for the Dynamic mode. Current ADS-B trajectories do not provide SI or conflicts. Then, it has been necessary to build a customised database based on temporarily modified trajectories that constituted enough SI.

15 ML algorithms for classification and 17 ML algorithms for regression have been assessed. Results confirm ensemble methods are the most suitable for classification and regression algorithms. Random forest is the best ML model for classifying aircraft pairs as SI and their probability. Extra trees and Extreme Gradient Boosting were the best models for predicting numerical values of the safety metrics. Therefore, independent ML models perform predictions about conflict detection based on ADS-B data.

Purpose

This deliverable is based on the work package (WP) 3 Task 3.2 Development of the AISA project's conflict detection module. The primary goals of this deliverable are:

1. To define the operational concept to detect separation infringements with ML algorithms based on a data-driven approach for different ATC roles.
2. To develop a methodology for this module's whole process, from the extraction of data from a database to conflict detection.
3. To analyse which of the potential state-of-the-arts ML algorithms is more suitable for conflict detection based on performance metrics.
4. To provide input data required and output information to be integrated into WP4.

Intended Audience

There are two main groups of the intended audience:

- Experts from the related fields,
- The AISA consortium.

The development of conflict detection module via AI SA deliverable (AISA D.3.2) is important for the consortium as:

- In the framework of WP3, it develops one of the ML modules for the AISA project.
- The document will provide direct input to the other technical work packages (WP3, WP4, WP5) and the related deliverables, by providing the conflict detection module developed based on ML techniques.

The document is also useful for external stakeholders, especially the following ones:

- Air Traffic Management (ATM) system developers who would like to understand how AI, and particularly ML techniques, can be integrated into ATM,
- ATM experts conducting related research,

General automation and AI experts would like to see the possible use of AI in a new domain.

Associated documentation

The document is linked to several AISA and ATM documents; here, only the most relevant ones are listed:

- AISA D2.1: Concept of Operations for AI Situational Awareness System.
- AISA D2.2: Requirements for automation of monitoring tasks via AI SA.
- AISA D3.2: Conflict detection module.
- AISA D3.3: Air traffic complexity module.

Terminology

The following table lists the abbreviations used in this document.

Abbreviation	Description
ADS-B	Automatic Dependent Surveillance-Broadcast
AI	Artificial Intelligence
AISA	Artificial Intelligence Situational Awareness
ANSP	Air Navigation Service Provider
ATC	Air Traffic Control
ATCOs	Air Traffic Controllers
ATM	Air Traffic Management
AUC	Area Under the Curve
CD	Conflict Detection
CPA	Closest Point of Approach
CRM	Collision Risk Model
DistoMinDis	Distance to reach the Minimum Distance
FDP	Flight Data Processing
FL	Flight Level
ft	feet
GS	Ground Speed
kts	knots
MAE	Mean Average Error
min	minutes
MinDis	Minimum Distance
ML	Machine Learning
NM	Nautical Miles
RFE	Recursive Feature Elimination
RMSE	Root Mean Squared Error

RMSLE	Root Mean Squared Logarithmic Error
SCU	Sector Control Unit
TimetoMinDis	Time to reach the Minimum Distance
SI	Situation of Interest
WP	Work Package
4DT	4 Dimensional Trajectory

Table of Contents

Abstract 4
Purpose 5
Intended Audience 5
Associated documentation 5
Terminology 6
1 Introduction 13
1.1 Literature review about conflict detection 13
1.2 Literature review of ML techniques applied in aviation 15
2 Framework for CD based on ML techniques 19
2.1 Problem approach 19
2.2 Conflict detection principles 20
2.3 Towards a data-driven approach based on ML 23
2.4 Operational approach for ATC roles 26
3 Switzerland airspace and OpenSky as database 33
3.1 Airspace description 33
3.2 ADS-B data from OpenSky: Database analysis 35
4 Feature engineering 38
4.1 Data cleaning & clustering 39
4.2 Addition of new variables 40
4.3 Generation of customised aircraft pairs 42
4.4 SI and safety metrics analysis 47
4.5 Final database 50
4.6 Problems identified during the database building 52
5 ML approach 53
5.1 Generalization problem: underfitting and overfitting 53
5.2 Unbalanced classification problems 54
5.3 Preparation of the dataset to ML algorithms 55
5.4 ML experiments 57
6 ML Results 59
6.1 ML predictor for the Static Mode 59

6.2 ML predictor for the Dynamic Mode 64
7 Conclusions 70
8 Bibliography 72
Appendix A ADS-B data 77
A.1 ADS-B inputs 77
A.2 Output Labels 79
A.3 Labels as features in the Dynamic mode 79
Appendix B ML experiment for Static mode 80
Appendix C ML experiment for Dynamic mode 84

List of Tables

Table 1 Summary of the literature review focusing on ML techniques applied to aviation 18

Table 2 Matrix of risk levels depending on ML predictions. 31

Table 3 Statistical values of the raw database. 36

Table 4 Statistical data of LSAZM567 airspace for the first week of AIRAC Jun 2019..... 37

Table 5 Aircraft models identified in the LSAZM567 database. 40

Table 6 Features addition by aircraft pair generation. 45

Table 7 Results of one-week simulations for Static mode. 48

Table 8 Results of the simulations for the Dynamic mode. 50

Table 9 Number of samples of each experiment for the Static mode. 59

Table 10 Metrics for the different optimised ML classification models for the Static mode. 60

Table 11 Results of the finalised models for the Static mode..... 61

Table 12 Metrics for the finalised ML regression models for the Static mode. 62

Table 13 Number of samples of each experiment for the Dynamic mode. 64

Table 14 Metrics for the different optimised ML classification models for the Dynamic mode. 65

Table 15 Results of the finalised models for the Dynamic mode..... 66

Table 16 Metrics for the finalised ML regression models for the Dynamic mode. 67

Table 17 Description of ADS-B variables [53]. 78

Table 18 Description of relative features. 78

Table 19 Description of output labels. 79

Table 20 Metrics of ML algorithms applied to Pure model for the Static mode..... 81

Table 21 Comparison of the model performance with feature selection of Pure Model for the Static mode. 82

Table 22 Hyperparameters selected for the Grid search of the Pure model for the Static mode..... 83

Table 23 Results of the different models of Pure model for the Static mode. 83

Table 24 Metrics of ML algorithms applied to Pure model for MinDis. 84

Table 25 Results for the validation set of Pure Model for the Dynamic mode..... 85

Table 26 Metrics for the optimised Pure model for the dynamic mode. 87

List of Figures

Figure 1 Description of the ML approach for conflict detection. 19

Figure 2 Combinatorial problem for conflict detection between aircraft pairs. 21

Figure 3 Representation of the training set. 24

Figure 4 Adaptability of the training set to the database available. 24

Figure 5 Representation of Static mode. 27

Figure 6 Representation of Dynamic mode. 28

Figure 7 Representation of 4DT prediction based on ADS-B trajectories. 29

Figure 8 Representation of loss in data quality from the resampled trajectory. 29

Figure 9 Example of the impact on separation by resampled trajectories. 30

Figure 10 Conceptual schema of trajectory modification for Dynamic mode. 30

Figure 11 Information flow for different ML predictions. 32

Figure 12 Switzerland en-route chart upper airspace. 33

Figure 13 Geographical location of LSAZM567 [52]. 34

Figure 14 Data representation of LSAZM567 airspace: left) altitude, medium) groundspeed and right) vertical rate. 36

Figure 15 Flow of feature engineering process. 38

Figure 16 Aircraft-pair generation for Static mode. 43

Figure 17 Aircraft-pair generation for Dynamic mode. 44

Figure 18 Data pre-processing process from ADS-B to ML model for the Static mode. 46

Figure 19 Data pre-processing process from ADS-B to ML model for the Dynamic mode. 46

Figure 20 Example of safety metrics results 48

Figure 21 Example of results of the function *conflict_detection*(). 49

Figure 22 Histograms of safety metrics for the Static database. 50

Figure 23 Histograms of safety metrics for the Dynamic database. 51

Figure 24 Comparison of safety metrics for ADS-B and resample database for Dynamic mode 51

Figure 25 Cross-validation structure. 54

Figure 26 ML process for experiments. 58

Figure 27 Example of predictions of the Hybrid model for the Static Mode 64

Figure 28 Example of predictions of the Hybrid model for the Dynamic Mode 69

Figure 29 Distribution of SIs by sets of Pure model for the Static mode. 80

Figure 30 Left) feature importance and right) RFE function of Pure model for the Static mode. 82

Figure 31 Predicted vs Measured values of the left) MinDis, medium) DistoMinDis and right) TimetoMinDis for the non-optimised algorithm of the Pure model. 85

Figure 32 Feature importance of Pure model for MinDis. 86

Figure 33 Feature importance of left) DistoMinDis, and right) TimetoMinDis for the non-optimised algorithm of the Pure model. 86

1 Introduction

AISA project proposes building a foundation for automation by developing an intelligent situationally-aware system instead of automating isolated individual tasks. This system will at first be able to automate some of the monitoring tasks because machines cannot currently reach the same level of awareness as humans. Still, as the development progresses, it will be able to take over more complex tasks. The goal of WP3 is to build different ML modules to perform predictions regarding situational awareness tasks, particularly in monitoring tasks. This WP develops three independent modules focusing on trajectory prediction, conflict detection and airspace complexity analysis. Therefore, the future AISA system provides situational awareness support to ATC based on the information provided by the ML modules.

This deliverable focuses on task 3.2 about conflict detection. The main goal is to determine whether the ML techniques could be used for conflict detection purposes in en-route airspace, to identify the conditions to perform that prediction and to adapt the ML model as input to the Knowledge Graph of the AISA system. This work's output should be the ML predictors and how to integrate them into the Knowledge Graph of WP4.

The three modules of WP3 are developed independently, although it does not mean they could not be integrated in the future. Task 3.1 aims to provide Four-Dimension Trajectories (4DT) predictions based on ML techniques. It is expected to improve the prediction of the trajectories regarding flight plans or pre-tactical trajectories. One of the potential integrations will be that these ML 4DT predictions will be used as input to improve the conflict detection process. Task 3.3 is about the analysis of the airspace complexity.

Similarly, it could improve its process by considering the conflict prediction output as input for the air traffic complexity determination. However, these tasks are developed in parallel because of the short period available to develop the ML models. One of the potential further works will be about integrating these modules' outputs as inputs of the other modules.

1.1 Literature review about conflict detection

Conflict detection is a crucial safety aspect because of the necessity to keep a sufficient airspace safety level. Conflict is a barrier prior to the collision of two aircraft in the airspace. Typically, a conflict occurs when two aircraft are in the condition to infringe the airspace's separation minima. Before moving forward, there are several concepts about conflict and collision that must be defined in advance.

Aircraft collision means aircraft is crashing with the ground or aloft with another aircraft [1]. International Civil Aviation Organization (ICAO) defines conflict as any situation involving aircraft and hazards in which the applicable separation minima may be compromised [2]. ATM community is evolving to use the term called potential conflict as those trajectories for which the future position of 2 or more aircraft might fall below specified minima (not necessary the separation minima). In addition, a situation of interest (SI) is an aircraft pair that the ATC must pay attention because their trajectories are expected to cross below specified separation (similar to potential conflict). Hence, there are similar conflict terms to bear in mind in this work.

The study and research of conflict and collision have evolved throughout the years. Since 1960, several authors assessed and proposed different ways to study aircraft collision-risk. Reich [3], [4] pioneered the collision-risk model (CRM) between aircraft in en-route airspace. This work was paramount because it settled the collision risk basics based on random flight errors (positioning and velocity). Later on, ICAO developed a manual to estimate separation minima values in different airspace [5], [6]. For a more in-depth review of CRM, the work presented by Netjasov and Janic is recommended [7] because it reviewed the evolution of different CRM.

Furthermore, conflict detection has evolved throughout the years to detect separation infringements. Regarding conflict risk, Netjasov developed a simple model to assess the conflict probability in en-route airspace. The work focused on different time frames: strategic [8], pre-tactical [9] and current-day planning [10]. Other authors researched conflict-risk models to quantify the airspace's safety levels [11]–[14]. Most of these works focused on the strategical or pre-tactical environment. Nonetheless, the ultimate aim is to provide an ATC tool for conflict detection to facilitate Air Traffic Controllers' labour (ATCOs).

ATC manages the air traffic within its responsibility and must act to avoid any potential conflict between them. Presently, ATCOs can evaluate conflicts by analysing the predicted trajectories in the Sector Control Unit (SCU). The SCU shows the aircraft's relative position and can estimate whether a conflict can occur based on their expertise about previous air traffic situations. The SCU evaluates the trajectories to facilitate the conflict-search task. It identifies potential separation infringements between an aircraft pair based on their current position and predicted trajectory. Typically, the way it is calculated is transparent for the ATCO, and it could be by using different mathematical techniques such as static models, worst-scenario or probabilistic case. In the case a conflict is detected by the ground server, it notifies this situation to ATCOs. Then, the ATCO analyses the problem and solves the upcoming separation minima infringement by acting in advance.

Conflict detection is a problem tackled from different points of view over the years. [15] is a paper that analyses and summarises different models for conflict and collision in air traffic. The evolution of the main approaches can be divided into three areas:

- Static case. This is the most straightforward approach and the worst scenario because it does not consider a stochastic evolution of the trajectories. The current and future status of the trajectories is fixed [16]–[20]. The static case encompasses every situation that can generate a separation infringement between aircraft pairs. The static case is the worst-case study because it does not distinguish between false alerts and overestimate separation infringements by trying to tackle all situations.
- Dynamic case. This case is the stochastic evolution of the static case by considering the current status of the aircraft is not fixed and evolves throughout the trajectory [13], [21]–[26]. The dynamic case performs a probabilistic analysis to consider only separation infringements with a very high probability. Typically, it encompasses statistically 95% of all situations and discards the rest. Typically, it reduces false alarms but increases missed alerts.
- Probe case. This research tries to identify the conditions that underlie multiple and diverse conflict situations, aiming to determine which of them can constitute a separation infringement [23], [27], [28]. The probe case is the most complex problem because there is no one-size-fits-all approach. The goal is to adapt the model to identify every situation by reducing false and missed alerts.

This work follows the probe-case research line using ML techniques to predict potential separation infringements. The goal is to harness state-of-the-art ML algorithms to learn the patterns that identify SI between aircraft pairs in a database. These patterns or conditions, underlying in a situation of interest, present intrinsic features (entry point, velocity, altitude, etc.) that can advance the existence of an SI within the airspace.

1.2 Literature review of ML techniques applied in aviation

ML is one of the most promising technologies based on computer algorithms for data processing and learning. EASA defines AI and ML as follows [29]:

1. Artificial intelligence (AI) is a branch of computer science that aims to create intelligent machines. It has become an essential part of the technology industry. AI can be narrow, handling just one particular task, or strong, meaning a machine with the ability to apply intelligence to any problem.
2. Within AI, ML is a core part of AI. It uses data to train algorithms and give computer systems the ability to "learn" (i.e. progressively improve performance on a specific task) with data, without being explicitly programmed. ML techniques are increasing their importance for data processing and data learning from huge databases. Currently, there are large databases where lots of information can be used by different data-driven methods.

Therefore, ML is a data-driven approach that can be applied to multiple topics with different techniques. The pillar is the capacity of information extraction and learning by the machine from a database without being explicitly programmed for one specific task [30].

ML algorithms can be split into three types of algorithms [31]:

3. Supervised learning: the algorithms train with human supervision because the outcomes, denoted as "labels", are provided to the algorithm associated with the database's features. The algorithm creates a model to predict the labels from the features.
4. Non-supervised learning: the algorithms train without human supervision because they have to learn from the features without knowing the outcomes. These algorithms focus mainly on clustering applications.
5. Semi-supervised learning: this is the newest learning because it is a mix of the types mentioned above. This type of algorithms learns from the features that contain some of the labels, but others not.

Aviation is an area where there is a massive quantity of information that ML techniques can analyse. ML techniques have been broadly applied to three topics in aviation (atmospheric models, airspace performance metrics and trajectory prediction) and, to a lesser extent, airport operations, runway occupancy and conflict detection & resolution. In [32], the authors proposed a similar classification, although focusing on different four areas: national airspace performance metrics, aviation safety, conflict detection and resolution and decision-making by ATC. The categories are not important because there are as many as authors. Hegde and Rokseth [33] provides an interesting review of the application of ML techniques in different areas, among which is aviation. They specifically focus on the applications of risk assessments with ML, identifying associated literature to other engineering areas.

Aiming to show organised the information extracted from the literature review, Table 1 summarises the different ML techniques applied to aviation grouped by different aviation areas. It seeks to identify which ML model was used, although it does not evaluate which model or application is the best.

Article	ML application goal	ML algorithms
	Airspace performance metrics	
[34]	Traffic flow prediction method based on Deep Learning. It considers the spatial and temporal correlations inherently. A stacked auto-encoder model is used to learn generic traffic flow features.	Neural Networks Stacked Auto-Encoder
[35]	It evaluates Deep Learning algorithms to predict flight delays. It detects patterns in air traffic delays.	Recurrent Neural Networks Long Short-Term Memory networks
[36]	DART is a SESAR project that demonstrates how ML methods help in improving trajectory predictions. It also seeks for improving the prediction of demand-capacity imbalances in airspace use.	Hidden Markov Models Reinforcement Learning
	It evaluates the feature engineering problem to predict aircraft landing time in Extended TMA with ML models. 4 sets of features present a significant impact on predictions, and three ML techniques are compared.	Gradient Boosting Machine Random Forest Extra-Trees
	Trajectory prediction	
[37]	Predict an aircraft trajectory in the vertical plane. The method depends on a small number of starting features. Two prediction methods based on the operation of real trajectories or not.	Neural networks
[38]	An ML approach to Trajectory prediction for sequencing and merging traffic in Arrival Manager scenarios are evaluated using real aircraft trajectory and meteorological data. A stepwise regression method is used to determine the inputs and functions of the prediction model's information.	Generalised Linear Models Stepwise Regression

[39]	ML to improve the aircraft climb prediction for ground-based applications. This paper predicted the mass based on ML techniques and compared it with the BADA model.	Multiple Linear Regression on the k variables selected by a forward-selection MLR-FSk Ridge regression Ridge with Principal component regression A single-layer neural network Stochastic Gradient Boosting Tree algorithm
[40]	ML to improve the aircraft climb prediction for ground-based applications. This paper predicted the climbing speed-profile based on ML techniques and compared it with the BADA model.	Gradient Tree Boosting
[36]	DART is a SESAR project that demonstrates how ML methods help in improving trajectory predictions. It also seeks for improving the prediction of demand-capacity imbalances in airspace use.	Hidden Markov Models Reinforcement Learning (Ind-Colab-RL, Ed-Colab-RL and Ag-Colab-RL)
[41]	ML predicts the operational factors required for trajectory prediction focusing on the climbing stage. It also analyses the impact of operational factors on the climbing trajectory.	Gradient Tree Boosting
[42]	It introduces a hybrid model to address the short-term TP in Terminal Manoeuvring Area (TMA) by applying machine learning methods. ML model is trained to predict the Estimated Time of Arrival (ETA).	Clustering-based pre-processing Multi-Cells Neural Network Principal Component Analysis Nested Cross-validation Multiple Linear Regression
[43]	It focuses on the application of ML algorithms and NN models to runway recognition trajectory classification study.	Most ML algorithms
	Conflict detection and resolution	

[44]	An ensemble approach for CD in free flight is proposed. Different CD techniques are evaluated. Data mining techniques are used to identify patterns where the CD algorithms do not correctly identify a conflict.	supervised Classifier System Linear Classifier System - genetics-based ML
[45]	It proposes the development of artificial intelligence capable of resolving conflicts. The conflict resolution manoeuvres consider the presence of traffic and environmental uncertainties. The ML does not need prior knowledge from expected actions.	Reinforcement learning: Deep Deterministic Policy Gradient
[46]	It introduces a conflict detection framework with ML methods. It aims to improve the Closest Point of Approach (CPA) prediction accuracy based on time to separation minima infringement with real trajectory data.	Multiple Linear Regression Support Vector Machine Feed-Forward Neural Networks K-Nearest Neighbours Gradient Boosting Random Forest
[28]	It introduces a conflict-detection framework with ML for 3D CPA prediction in a look-ahead time of fewer than 20 minutes. ML models predict the time, horizontal and vertical distance of CPA based on real trajectory data.	Feed-Forward Neural Networks K-Nearest Neighbours Gradient Boosting Random Forest

Table 1 Summary of the literature review focusing on ML techniques applied to aviation

To sum up, the most extended field related to ML until now could be the trajectory prediction. There is more research in airspace performance metrics, although they are not included herein for the sake of clarity. There are no references about ML techniques applied to atmospheric conditions because they are out of the problem's scope to tackle herein. The authors recommend the work [32] that describes several previous research on this topic. Regarding conflict detection, two articles tackled different approaches based on data mining techniques or ML. The solution for the lack of separation infringements was to simulate [45] or modify trajectories extracted from radar [46]. Notably, [47] was interesting because it analysed different ML techniques to calculate the closest point of approach and the minimum separation between aircraft pairs. Finally, it can be concluded that little research deals with applying ML techniques for conflict detection.

2 Framework for CD based on ML techniques

This section details the framework developed in task 3.2, aiming to apply ML techniques for conflict detection. One of this work's goals is to analyse the viability of using ML models to predict separation infringements with or without having 4DT predictions as inputs. Firstly, the underlying problem and the methodology used for ML techniques implementation is described. Secondly, the conflict detection principles and the safety metrics that characterise separation infringement predictions are explained. This data-driven approach focuses on the implementation of ML techniques, and then, it is detailed the operational concept for this novel approach. Lastly, this data-driven approach tackles two operational models based on ATC roles.

2.1 Problem approach

As new technologies appear, it is required to evaluate their viability in present problems. This work deals with introducing new technology (ML algorithms) to a present problem (conflict detection). The problem does not vary compared with the one described in the introduction. However, the new approach applies new data-driven approaches to perform separation infringement predictions during tactical air traffic management. Figure 1 shows a flow diagram of the problem approach.

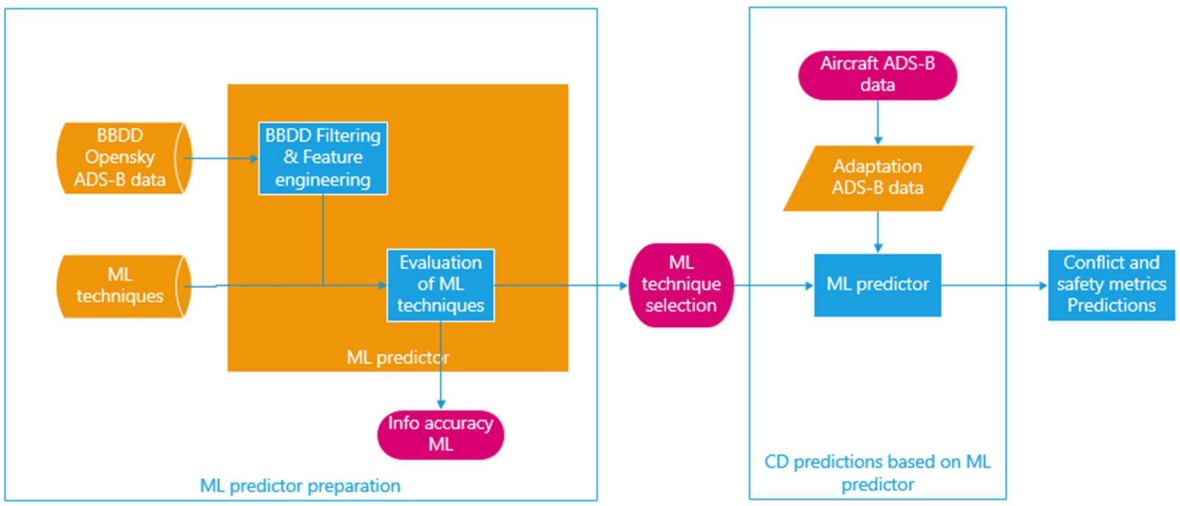

Figure 1 Description of the ML approach for conflict detection.

The application of data-driven approaches presents different methodologies and necessities compared with traditional model-driven methods. The main requirement is to have a database to extract the information to train the ML model, which can be the most complicated part. This work considers a database based on Automatic Dependent Surveillance-Broadcast (ADS-B) trajectories. Further research should evaluate this methodology's compatibility with other data sources, e.g., by using radar data instead of ADS-B. However, ADS-B trajectories cannot be implemented immediately because the previous filtering and adjustment of the ML algorithms' trajectories are required.

Currently, there are several state-of-the-art ML techniques, and it is expected to evaluate different of them. Considering several of them will permit identifying the best among them, although other ML techniques out-of-the scope of this work could improve the results. The results of this process are to provide a trained ML model that could perform predictions. Once the ML model is trained, it can be used to perform predictions. However, this application demands a similar ADS-B pre-processing before using them for predictions.

2.2 Conflict detection principles

This work deals with monitoring tasks focusing on situational awareness. With this aim, this work uses the concept of Situation of Interest (SI). One SI is when an aircraft pair is expected to intersect with a horizontal separation lower than a pre-defined separation. This pre-defined separation is specified by the ANSP and is larger than the current separation minima ($S_{min} = 5$ Nautical Miles - NM - and $H_{min} = 1000$ feet - ft). It is more convenient to talk about SI, a similar concept to potential conflict because ML's predictions present unknown uncertainties. These unknown uncertainties are difficult to quantify and make it difficult to identify separation infringements when the goal is to provide information about situational awareness. Therefore, these aircraft pairs can be classified into two groups:

- SI: aircraft pairs that cross with separation minima lower than specific separation. Herein, the pre-defined separations are horizontally 10 NM and vertically 1000 ft.
- No Si: aircraft pairs that cross without infringing pre-defined separations.

A conflict can then be considered a reduced SI group, i.e., every conflict means SI, but not every SI implies conflict.

This document uses the terms detection and prediction interchangeably. The reason is this work try to predict a separation infringement within the airspace. At one specific location or with some look-ahead time, it performs a prediction of separation evolution to know whether an aircraft pair will infringe specific separations. Generically, this situation encompasses all definitions described in section 1.1: conflict, potential conflict and situation of interest. Besides, this document uses the term conflict detection to analyse the evolution of the separation between an aircraft pair. However, the results and implications are related to SI. Therefore, it is important to note these potential ambiguities in the terminology that could mislead interpretations.

Moreover, this work evaluates a set of safety metrics for each aircraft pair. Safety metrics are defined as those obtained for each aircraft pair about their intersection's operational characteristics. One SI can be characterised based on different metrics such as time or distance to the minimum separation or the conflict probability. This information is crucial for the system and ATCOs to take in advance actions to avoid separation infringements.

2.2.1 Generation of SI based on ADS-B simulations

The first hindrance to the statistical analysis of SI is the lack of real situations. The primary responsibility of the ATC service is to avoid these type of situations. When the different safety barriers fail, a separation infringement can occur from strategical to tactical level [23]. Hopefully, the separation

infringements arise in rare situations and by specific circumstances [47]. Conversely to these rare situations, ML techniques need an extensive database to learn the patterns that underlie SI situations. Therefore, the first necessity is to generate a database with a sufficient number of SI that could be used to train the ML algorithms.

Up to now, different methods have been used to simulate conflicts or SI. It has been used top-down models where the conflicts' characteristics were defined in advance, and the simulations represented those situations [48]. Other methods performed simulations adding uncertainty to different flight variables (wind, velocity, weight, etc.) [12], [17], [21], [27]. Herein, the author's approach modifies the entry time of real trajectories randomly (from ADS-B data) to pierce the airspace in the same time period. Therefore, SI situations can be simulated based on real trajectories about 'what-if' situations. This approach provided positive results in previous research, e.g., [46] performs an experiment where all aircraft pierces the French airspace at the same time.

One limitation of this approach is the weather, and operational conditions vary throughout days and hours. A second problem to generate aircraft pairs is the high computational workload. This issue is a combinatorial problem depending on the number of aircraft involved, e.g., a set of n aircraft generates $O(n^2)$ aircraft pairs see Figure 2.

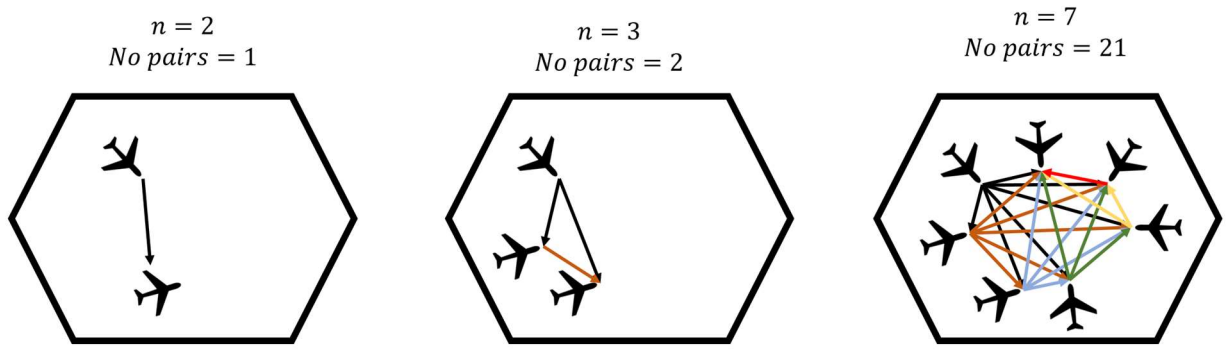

Figure 2 Combinatorial problem for conflict detection between aircraft pairs.

The solution applied to both problems is the same: to cluster the trajectories in specific time periods. The cluster of trajectories based on time periods allows considering the same weather and operational conditions for the aircraft pairs and reducing the computational workload by considering only a specific set of trajectories.

2.2.2 Safety metrics

Safety metrics provide information about the characteristics of a SI or the values acquired by specific variables. This information is crucial because it identifies the SI and includes information about the probability and severity. It has been considered the following ones.

- Minimum Distance($MinDis$): It is the minimum separation reached by an aircraft pair (a_i, a_j). This variable considers the horizontal separation (s) and vertical separation (Δh) between aircraft pair and provide information about the severity of the SI.

$$MinDis = s(a_i, a_j) = \min(s(a_i, a_j)\ if\ \Delta h < H_{min}\ \&\ s(a_i, a_j) < s_{SI}$$

$$otherwise\ MinDis = s(s(a_i, a_j) = \min(s(a_i, a_j)) + \quad \textbf{(1)}$$

$$\Delta h(s(a_i, a_j) = \min((a_i, a_j))S_{min}/H_{min}$$

- Situation of interest (SI): Aircraft pair are denoted of interest (SI) when they cross with a vertical separation lower than H_{min} and horizontal separation $s_{hor} < s_{SI}$. Specific separation for SI in this work is $s_{SI} = 10\ NM$. However, each Air Navigation Service Provider (ANSP) can define this control variable. Therefore, an SI is defined as:

$$if\ \Delta h(a_i, a_j) < H_{min}\ and\ s < s_{SI}\ for\ the\ same\ t\ \rightarrow SI = 1, otherwise\ SI = 0 \quad \textbf{(2)}$$

- Distance to Minimum Distance ($DistoMinDis$): it provides the distance required to reach the MinDis. This value depends on the timestamp considered (t_0).

$$DistoMinDis = (Long, Lat)(s(a_i, a_j) = MinDis) - (Long, Lat)(t_0) \quad \textbf{(3)}$$

- Time to Minimum Distance ($TimetoMinDis$): it provides the time (t) required to reach the MinDis. This value depends on the timestamp considered.

$$TimetoMinDis = t(s(a_i, a_j) = MinDis) - t_0 \quad \textbf{(4)}$$

- Probability of prediction (P_{pred}): Several ML algorithms for classification problems provide the probability of their predictions. This is the ML algorithm's probability of classifying as SI (1) or No SI (0). This value must be considered as a confidence level of the prediction.

Throughout the project development, other safety metrics have been studied as distance and time to conflict. However, the formulation of these safety metrics and the results obtained were noted right and were discarded. These or other safety metrics should be studied in further works.

2.3 Towards a data-driven approach based on ML

This work's primary goal is to develop a methodology or process to perform predictions about separation infringements using ML techniques. Later on, these predictions would be integrated into the AISA system. One requirement to apply ML techniques is to have a database from which the ML algorithm could learn the underlying patterns to perform predictions in other situations. This database is composed of a set of historical situations based on aircraft pair trajectories. The main problems to build a database are the data source and the 4DT predictions that will affect the model constitution and performances.

Currently, the main data sources available are DDR2 provided by Eurocontrol [49], ADS-B data provided by The OpenSky Network, and radar data offered by ANSPs. The information provided by radar surveillance is different from ADS-B data, and the data format and the datalink used. Based on historical data, there are three types of available trajectories: flight plans (strategic goal), pre-tactical trajectories (pre-tactical goal) and flown trajectories (tactical goal). Every kind of trajectories provides different information with different level of accuracy.

Another issue is about the source to obtain the 4DT prediction. ANSPs and aircraft can provide 4DT predictions performed by their systems. However, the accuracy of the predictions differs from the prediction source. Both systems predict 4DT based on the available information. The available data can be the flight plan, a 4DT prediction performed by the system, or a prediction obtained from the aircraft. Nonetheless, the worst situation would be when the on-ground system does not have any information about future aircraft intent.

Herein, the approach followed seeks to be a one-size-fits-all methodology suitable for different data sources and 4DT predictions, although this work only deals with ADS-B data. It provides a whole process that could be adaptable to an additional data source. However, one limitation of this work is the compatibility of different data sources and how to combine them with other servers. This issue is one of the future problems that the digitalisation process will face [50], [51].

Therefore, it is essential to note that the goal of applying ML techniques of this work is to provide predictions focusing on separation infringements but not in trajectory prediction. This means the ML process does not perform the trajectory's prediction and then analyses the separation infringement that could occur. The ML process predicts the separation infringements based on the available information. This approach is novel because it employs ML algorithms to learn the operational factors that lead to SIs.

The first step is to train an ML predictor that could perform SI predictions for aircraft pairs. To this end, a database must be used to train the ML models. The number of aircraft pairs considered in the database is crucial because the greater their number, the greater the prediction's ability. Besides, the features of the database provide information crucial to the database. ML models demand a database with different operational characteristics between an aircraft pair to perform the prediction. Conversely, to increase the number of samples means higher computational times. The goal is not to provide the best prediction but to prove that this technology can predict SIs.

This work considers four labels or targets: the existence of SI or not, probability of SI, minimum separation to reach (MinDis), distance up to get the minimum distance (DistoMinDis) and time up to minimum distance (TimetoMinDis). According to the type of labels, there are two different problems to solve with ML. Identifying whether a conflict or SI will happen is a classification problem. The conflict

probability is related to the significance level of the classification prediction. On the other hand, the regression problem makes numerical predictions, as is the safety metrics. Different ML models must be used independently for each target. It is important to note that both ML predictors must be trained in the same database. Otherwise, inconsistencies can arise between them.

Typically, the database is split into two sets: training and testing set. The training set is provided to the ML model to learn the relations and generate a mathematical model to make predictions. The testing set is used to validate the model trained and know how it will work for new instances. ML model must learn from the historical information (training set) the operational conditions to predict a SI. The approach evaluates the features for every aircraft pair at each timestamp. The features relate the operational characteristics with the occurrence of a separation infringement in the future. These features (such as initial separation, relative velocity, vertical speed, etc.) are detailed in section A.1. Figure 3 shows a draft of the training-set distribution in features and labels.

Figure 3 Representation of the training set.

One of the future research lines of this methodology will be to analyse the compatibility with different data sources, i.e., the information source of the trajectories can come from different ways: flight plans, 4DT predictions or Flight Data Processing (FDP) data. Figure 4 shows a scheme of the application of this potential compatibility.

Figure 4 Adaptability of the training set to the database available.

However, ML projects present several problems associated with data consistency, algorithms and their optimisation. The identified issues with data consistency are:

- Lack of data. ML algorithm needs lots of data to learn the patterns that underlie the problem. The larger the database, the better metrics obtained. Most complex ML models can demand millions of samples as well as features that constitute each sample. However, the cost of using millions of samples is the computational cost in terms of time and memory required.
- Non-representative data. This problem arises when the data acquired is non-representative of the situation to analyse. The training data do not correctly represent the situation to predict, e.g., it could be the case in which the model is trained for one airspace and later on employed in different airspace.
- Outliers. Outliers are data points that differ significantly from the observations. In the case the number of outliers is high, the ML model will not predict the situations correctly.
- Imbalance. Unbalanced problems are those in which the classes of the dataset are not equally represented. This is a typical problem that appears in this work, and it is detailed in section 5.2.

On the other hand, ML algorithms tend to over or underfitting with the training set, i.e., they can perform perfect predictions in the training set and provide bad performances on new samples. This is a generalisation problem that can be solved by different techniques, as is explained in section 5.1.

2.4 Operational approach for ATC roles

This work develops two modes for each ATC role. They have been considered because their finality and information requirements are different. This approach evaluates the application of ML techniques to different databases to identify which of them provides better results.

The Static mode focuses on planner ATCO. This mode predicts SI and their safety metrics when an aircraft pierces into the airspace. The predictions are performed with the aircraft located within the airspace. Aircraft beyond the airspace are not considered. This prediction is performed once and is not updated. It does not receive as input the 4DT predictions of the different aircraft. Finally, the Static mode provides information about SI between the aircraft that pierces into the airspace with other aircraft within the airspace, a similar function that the planner ATCO currently does.

The Dynamic mode focuses on tactical ATCO. This mode predicts SI and their safety metrics throughout the evolution of the aircraft within the airspace. The predictions can be performed dynamically and not once as the Static mode. Then, it considers aircraft within the airspace, and that will pierce in the same time period. Another difference is the Dynamic mode receives a 4DT prediction of the aircraft within the airspace. The SI and safety metrics predictions of the Dynamic mode evolves with the trajectory evolution. Finally, the Dynamic mode provides updated predictions throughout the evolution of the aircraft within the airspace.

Therefore, both modes respond to the viability of using ML techniques for separation infringement predictions. Besides, the Dynamic mode confirms whether ML models improve the outcomes of the 4DT predictions using those 4DT predictions as input data.

2.4.1 Static mode

The Static mode focuses on the planner controller and presents the following characteristics:

- It performs a prediction when one aircraft pierces into the airspace. The prediction is performed once and is not updated. This fix snapshot provides information with the rest of the aircraft located within the airspace. However, this condition could be modified to perform predictions 1, 2 and 3 minutes or 2, 5, and 10 NM before the aircraft pierces into the airspace. The ANSP should define this requirement.
- It only evaluates separation infringements with aircraft within the airspace.
- The system does not receive information about 4DT predictions. The only information available to perform the prediction is the state vector of the aircraft.
- It focuses on the planner controller role, providing information about the potential aircraft pairs of interest. With this information, the planner controller will take the ATC actions more convenient.

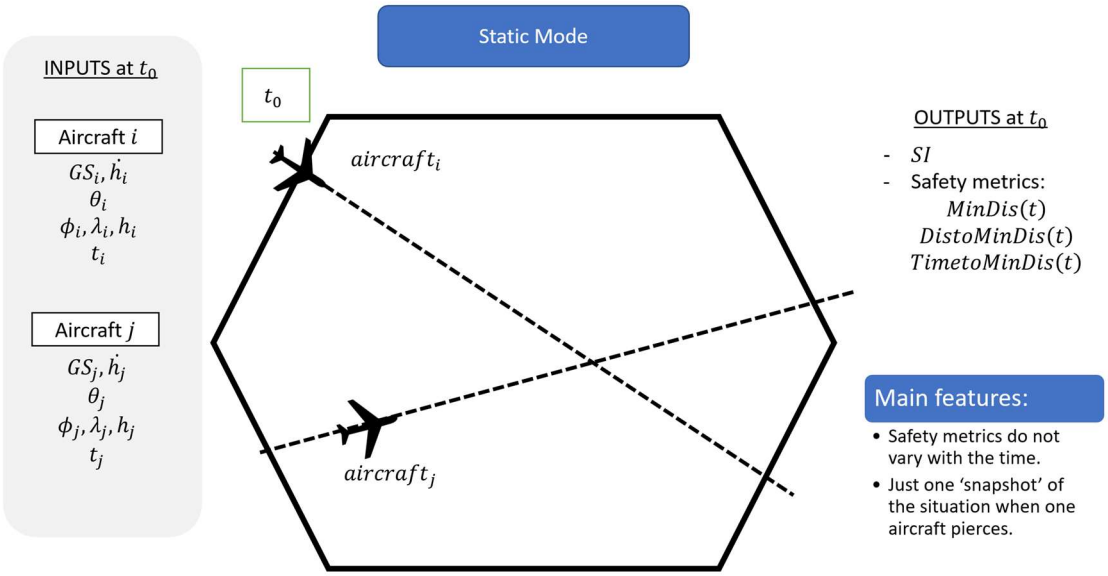

Figure 5 Representation of Static mode.

Figure 5 represents the operational concept of the Static mode. When aircraft i pierces the airspace the system seeks for SI with the aircraft within the airspace. The information available at this instant is the state vector based on ADS-B data. ML predictor must provide predictions about SI and their safety metrics.

2.4.2 Dynamic mode

The Dynamic mode focuses on the tactical controller and presents the following characteristics:

- It performs predictions throughout the aircraft's evolution within the airspace, and the prediction varies with the time throughout the flight trajectory. In this way, the system has update information each c minutes.
- It considers aircraft within the sector and aircraft that are in the proximities of the airspace. The requirement is the aircraft will pierce the airspace in the following m minutes. Both c and m values should be defined by ANSPs.
- The system provides a 4DT prediction for each aircraft and is used as input for the ML model.
- It focuses on the tactical controller's role providing continuous surveillance of the aircraft within the airspace. With this information, the tactical controller will take the ATC actions more convenient.

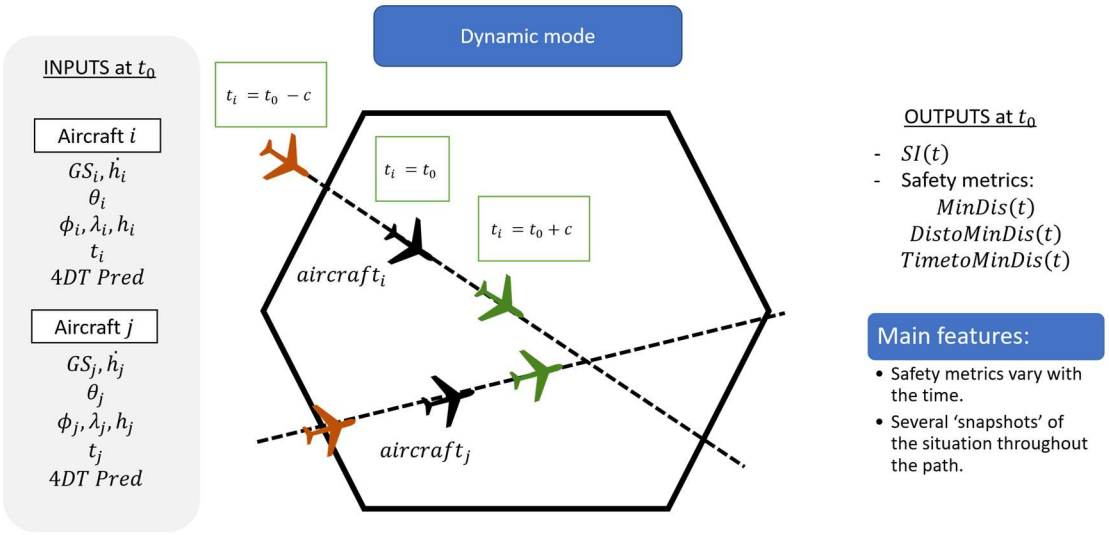

Figure 6 Representation of Dynamic mode.

Figure 6 represents the operational concept of the Dynamic mode. The aircraft i pierces into the airspace, and the system performs SI prediction with the rest of the aircraft within the airspace and aircraft that will pierce into the airspace in m minutes. The available information is the aircraft's state-vector based on the ADS-B data and a 4DT trajectory prediction of both aircraft. With this information, the ML predictor performs SI and safety metrics prediction. The ML predictor calculates the prediction of the SI and safety metrics throughout the airspace trajectories' evolution.

The main problem of the Dynamic mode is to get 4DT predictions of the aircraft. Typically, the SCU or the aircraft (Extended Projected Profile ADS-B) provide 4DT predictions. In this work, it is not possible to obtain this type of predictions. Based on OpenSky's ADS-B data, a whole dataset of 4D trajectories that flew within the airspace previously is available. It is assumed the 4DT prediction can be a similar trajectory stored in the ADS-B database. Therefore, the 4DT prediction will be considered based on stored 4DT trajectories from ADS-B data. The introduction of different 4DT predictions based on SCU or aircraft information should be one topic to research in further works.

It is required to define several operational requirements to ensure the similarity between the 4DT prediction and a trajectory stored in the ADS-B database:

- The first filtering selects one trajectory with the same callsign. Typically, aircraft repeat their trajectories throughout time.
- The second filtering considers operational restrictions considering velocity, heading and location of the entry point.
    - The velocity difference must be minor than 10 knots.
    - The heading difference must be minor than 2º.
    - The location difference of the entry points must be minor than 5 NM.
- In case there is more than one trajectory that fulfils previous restrictions, the trajectory with a minor heading difference will be selected.
- If none of the same callsign's trajectories fulfils the above restrictions, the algorithm extends the search to other callsigns.

Figure 7 shows an example of the above process. It represents the selection of 4DT prediction based on ADS-B trajectories that can be the most similar when one aircraft pierces the airspace.

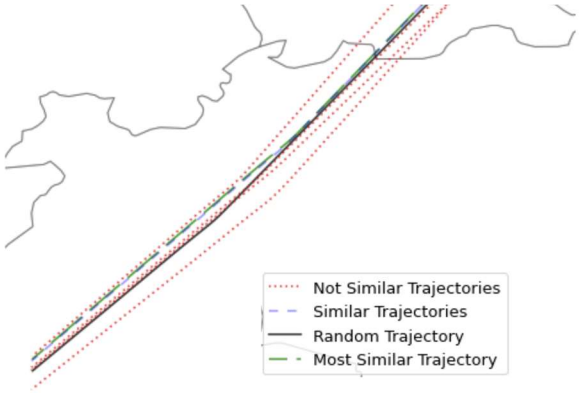

Figure 7 Representation of 4DT prediction based on ADS-B trajectories.

Once the 4DT prediction is selected from the ADS-B database, two modifications have been made to ease the system's interoperability in this work: resample and time deviation.

The first modification is a trajectory resample depending on the frequency ϵ of timestamps considered. This value can vary from seconds to minutes, and the ANSP can define it. The resample's goal is to reduce the computational workload and the final database size because of the high number of ADS-B state vectors. Typically, ADS-B state vectors provide information every second. Therefore, the 4DT prediction is a resampled trajectory of the ADS-B stored data, and it does not have the same quality. It is not an issue to resample a trajectory going on a straight line, but if the aircraft is turning or doing a manoeuvre, it won't be reflected correctly, as Figure 8 shows.

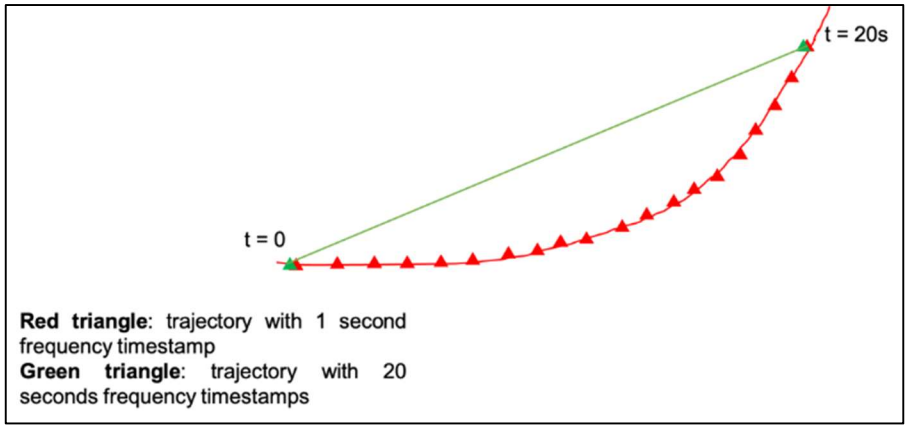

Figure 8 Representation of loss in data quality from the resampled trajectory.

The resample is also critical for detecting if two aircraft have a conflict because of the minimum distance an aircraft pair can reach. Typically, the resampled trajectory is composed of fewer data points. The minimum distance that could be reached will be higher than the minimum distance reached by the full ADS-B trajectory. Therefore, the goal is to improve the separation prediction performed by the 4DT prediction (resampled) by implementing ML techniques. This approach is different from the Static mode because no 4DT prediction is provided to the ML algorithm. Figure 9 shows an example of the variations in the separation evolution due to the resampled trajectories.

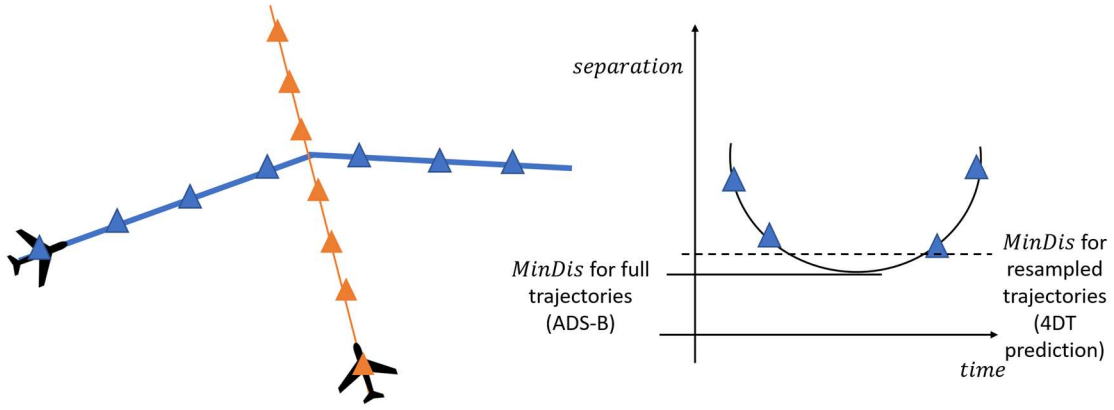

Figure 9 Example of the impact on separation by resampled trajectories.

The second modification is about the time deviation of the 4DT prediction. This 4DT prediction must adapt its entry time to the real trajectory, which the ML will perform the prediction. However, it is necessary to introduce some uncertainty between the real trajectory's entry time and the 4DT prediction. Then, the second modification is about a time deviation (τ) of the entry time of the resampled trajectory (4DT prediction). The larger this value, the larger the error between the real trajectory and the 4DT prediction. Herein, the uncertainty considered is a random distribution of ± 20 seconds. This is a value selected by the authors, and further work should study its validity.

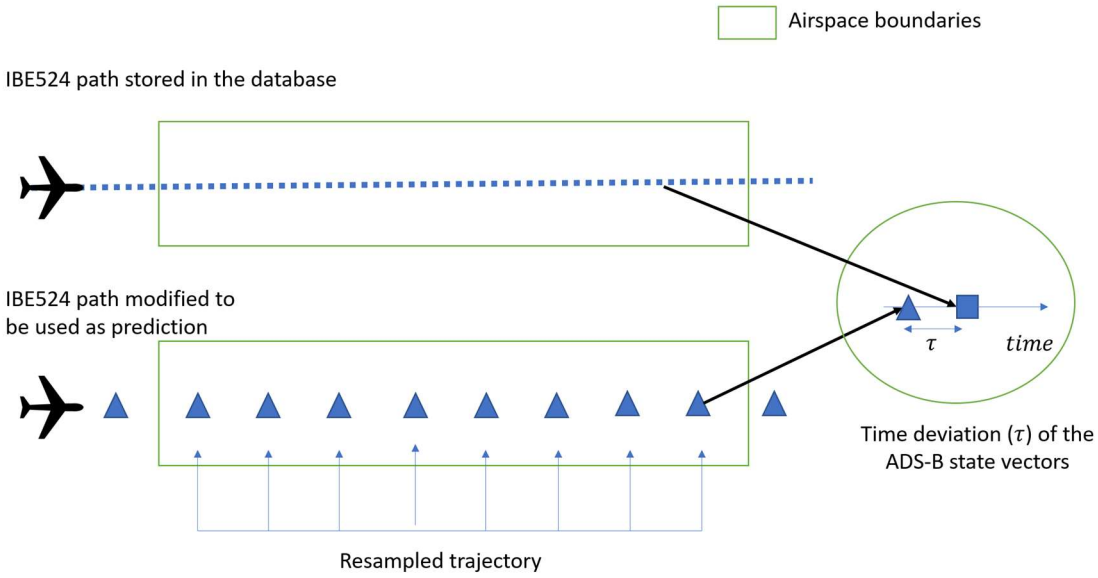

Figure 10 Conceptual schema of trajectory modification for Dynamic mode.

Figure 10 represents the modifications performed to the ADS-B trajectory stored in the database and used as 4DT prediction. The triangles represent the resample trajectory, reducing the number of samples of the 4DT prediction to reduce the computational workload. In addition, these triangles are deviated temporarily to introduce uncertainty to the entry points.

2.4.3 ML predictions and implications to ATC

There are two types of problems to solve with ML algorithms: classification and regression. One issue of classification is the problem of clustering between two classes (binary) or multiclass. The ML algorithms must learn which class a sample belongs to, based on its features. The regression problem aims to provide a numerical prediction of the target. In this work, the application of both approaches are the following ones:

- Classification. ML algorithms provide predictions about which aircraft pair are SI and which not. It is required to have a database that contains multiple situations of aircraft pairs. Once the model is trained with diverse samples, the ML algorithm can predict whether an aircraft pair labelled as SI or not. Besides, some algorithms provide information about these predictions' confidence level, i.e., the prediction's probability. However, classification algorithms can provide erroneous predictions:
    - Missed alerts: aircraft pairs that constitute an SI but the ML predictor erroneously classify as no SI. This is the worst situation because the ML predictor does not inform about aircraft pair that could infringe the separation in the future.
    - False alerts: aircraft pairs that do not constitute an SI but the ML predictor erroneously classify as SI. This is also bad performance of the ML predictor because it will increase the ATCO's workload.

  Different metrics of the ML algorithms can evaluate both rates. Although it is more important to reduce the missed alerts than the false alerts, this is something that the ANSP should balance previously. This work will try to minimise both missed and false alerts levelly.
- Regression. ML regression algorithms provide numerical predictions of the labels selected. These labels refer to the safety metrics considered herein. However, this type of algorithms does not give information about the level of confidence in the predictions. The only information about the effectiveness of the ML algorithms is about overall metrics.

This work will develop ML classification and independent ML regression predictors for the different safety metrics. The classification predictor provides information about whether an aircraft pair is an SI and its probability. The regression predictor provides information about safety metrics. To consider both predictor's type is a strong point of this approach because a double check is performed between the classification and the regression predictor of MinDis.

However, there is a problem when the SI prediction and the MinDis prediction do not match, i.e., the prediction of classification is No SI, but the regressor predicts a MinDis of 3 NM or vice versa. The solution is to avoid these misleading situations by including a prediction risk level. There are several combinations between the predictions that can have different risk levels. Table 2 shows the different types that can arise.

Regression	Classification	
	No SI (>10 NM)	SI (<10 NM)
[10;>10) NM	Level 0 - No inform ATC	Level 1 - Inform ATC
[0;10) NM	Level 1 - Inform ATC	Level 2 - Inform ATC

Table 2 Matrix of risk levels depending on ML predictions.

Three prediction risk-levels are denoted numerically and depend on the different combinations of the predictions.

- Level 0. Both ML predictors predict no SI will occur.
- Level 1. One ML predictor predicts SI will occur, but the other ML predictor denies it.
- Level 2. Both ML predictor predicts SI will occur.

Level 1 is characterised because one of the two ML predictors provides an aircraft pair classified as SI. To consider level 1 as information to provide to the ATC means an increase in the ATC monitoring task and can reduce the capacity. Further research should study if level 1 cases should be notified to ATC regarding the percentage of samples denoted as Level 1 and the impact on ATC workload. Lastly, the colours used in Table 2 are a proposal about how the ATCO could receive this information graphically. Therefore, the application of ML techniques for conflict detection can be improved with the results of two independent ML predictions, see Figure 11.

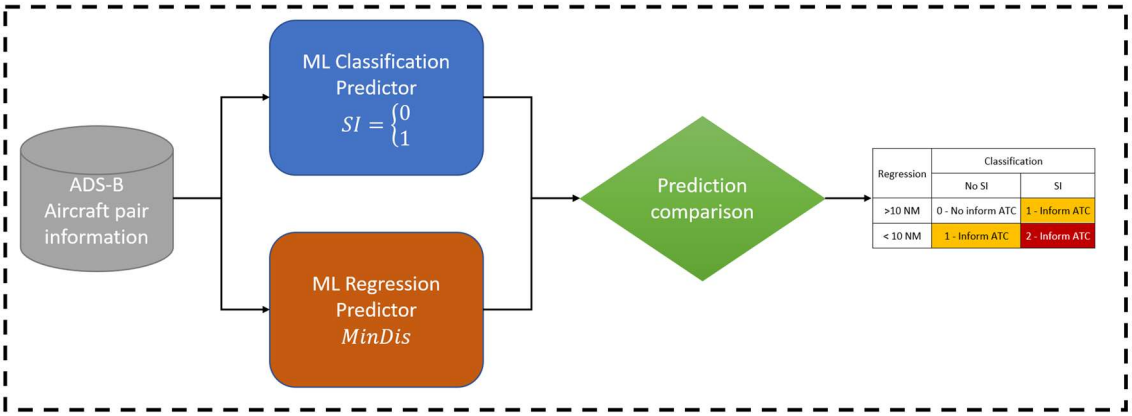

Figure 11 Information flow for different ML predictions.

3 Switzerland airspace and OpenSky as database

This work focuses on en-route airspace characterised by a low rate of climbing and descending aircraft and, on the other hand, a high number of cruise operations. According to the WP3's tasks, the LSAZM567 airspace volume in Switzerland and the time period is the AIRAC cycle of June 2019 are selected. This section describes this airspace and analyses the operational features of the air traffic sample.

3.1 Airspace description

Switzerland airspace is located in central Europe. At higher Flight Levels (FLs), the air traffic is characterised by overflights that connect different European countries. Switzerland airspace is split into lower (from the ground to FL195) and upper (from FL195 to upwards). The controlled airspace is limited in the upper airspace up to FL660, and aircraft must follow published airways, see Figure 12.

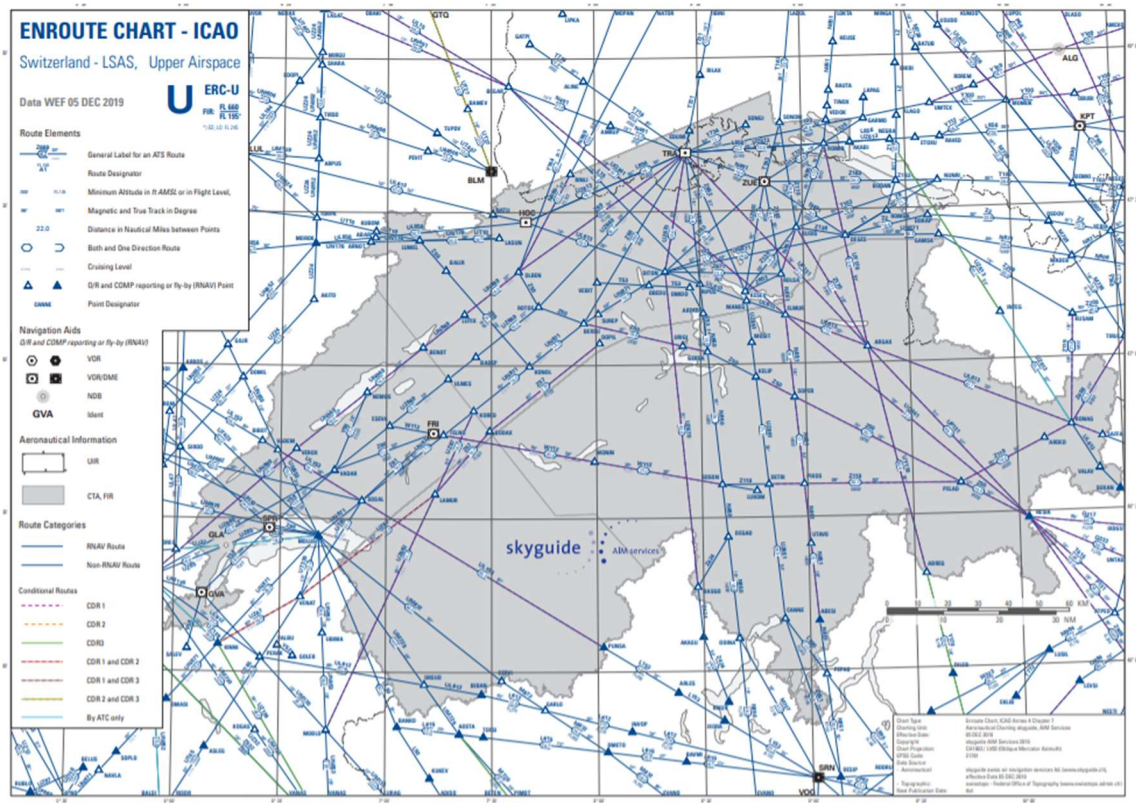

Figure 12 Switzerland en-route chart upper airspace.

This en-route chart is dated 05 December 2019, the same year of the air traffic data used in this work. Nonetheless, this chart could present minor modifications with the chart published for the period of study.

The typical distribution of airspace sectors in the European airspace is geographical split into lower and upper. The vertical division between lower and upper sectors generally do not match with lower and upper airspace boundary. The vertical division between lower and upper sectors are usually located in a stretch from FL300 to FL350, depending on the air traffic distribution. The higher the air traffic in the upper flight levels, the higher the sector boundary is.

Switzerland airspace does not precisely follow this geographical sectorisation. Two airspaces constitute Switzerland airspace (Geneva – LSAGUTA and Zurich – LSAZUTA), split vertically into several sectors. The different sectors cover the same geographical boundaries, but they have other FL boundaries. The reason is the high number of aircraft flying cruise trajectories. This work has selected the airspace sector LSAZM567 from the Zurich airspace with vertical boundaries from FL355 to FL660. Figure 13 shows an image of the LSAZM567 airspace in Switzerland.

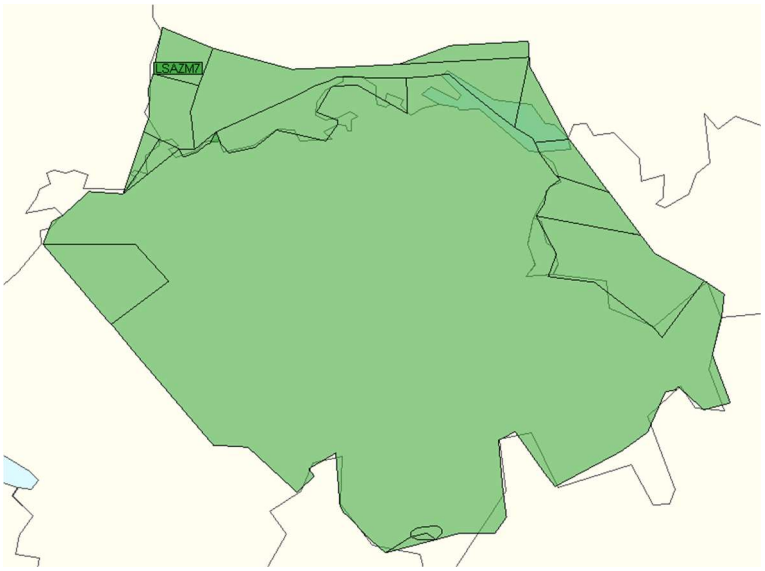

Figure 13 Geographical location of LSAZM567 [52]**.**

3.2 ADS-B data from OpenSky: Database analysis

Intending to apply ML techniques, the first need is to dispose of a 4D trajectory database, i.e., every aircraft position is referenced with a timestamp. The aircraft database's typical workflow is to build it by simulations based on ideal or similar conditions of a particular case study. However, it is becoming increasingly common to find aeronautical databases that provide real aircraft trajectories. This work takes advantage of ADS-B trajectories available (with license) in OpenSky [53]. OpenSky provides aircraft trajectories based on ADS-B upon gathering that information from aircraft in the European airspace. These trajectories were extracted from different ADS-B receptors in the European airspace. This technology allows aviation engineers for arranging an ADS-B database from restricted access to the users. Nonetheless, it could be extracted other real trajectories from different data sources as Flight Aware (ADS-B) [54], radar (DDR2) [52], or directly received by some ANSP. Appendix A provides an in-depth description of the ADS-B data and their application to this problem.

One of the assumptions previously mentioned is the trajectories belong to the AIRAC cycle of June 2019 in the LSAZM567 airspace. Although the whole month's ADS-B information has been downloaded, it only has been studied for 15 days due to the high computational workload. The problem of this reduction is because the combination of aircraft trajectories (n) to generate aircraft pairs is O(n^2). If there is around 10^3 trajectories for one day; over 10^6 aircraft pairs can be generated. To evaluate the whole month, it would be required to optimise the process described in this work. This task is not part of this work and could be performed in further research. However, the limitation of the number of trajectories used is not a problem for this task.

Besides, aircraft that operate from 22 to 7 hours have been considered. This period is regarded as the night period in which the air traffic demand drops sharply. The airspace configuration is modified up to only one airspace volume that handles the whole Switzerland airspace. This lower air traffic demand allows ATCOs to divert trajectories from flight plans by ATC instructions. These instructions provide benefits regarding time, consumption or distance. Then, trajectories operated during the night period can be considered as non-typical situations. These non-typical situations are difficult to identify during the daytime period when ATCOs are limited due to the high number of aircraft and workload. Therefore, these trajectories' embedding provides non-typical situations to the database, and the ML algorithm can consider them. Further research should analyse this influence and could develop different ML algorithms depending on the time frame.

It has been downloaded from the raw database from OpenSky from 20 June to 19 July 2019[1]. For this time period, an analysis of the raw database has been performed. The goal of this analysis is to identify statistical features of the air traffic of the operational variables. Besides, this statistical analysis will characterise the limits to identify trajectories that the ML algorithm should not evaluate. The ML algorithm should not perform predictions if the operational variables are not in the raw database range.

Firstly, the raw database is analysed as a whole for the LSAZM567 airspace. The information about the database size is:

[1] 4th July 2019 is not considered because it will be selected as the test day in the AISA system.

- Database volume: 2.1 GB.
- The number of ADS-B samples: 15477475.
- The number of trajectories: 25530.

Secondly, primary operational variables are evaluated statistically in Table 3:

	Maximum	Minimum	Mean	Standard deviation	95% limits
Altitude (ft)	47950	35500	37672	1547	[34578; 40766]
Groundspeed (knots – kts)	597	101	447	39	[389; 525]
Vertical rate (ft/min)	3264	-3264	3	247	[-491; 497]

Table 3 Statistical values of the raw database.

The maximum value of the altitude is FL480, which does not match with the 95% boundaries. The number of aircraft above FL410 is reduced; then, higher aircraft could be treated as outliers. The minimum 95% limit is lower than the real minimum value of the database. This result was expected because the lower vertical boundary is FL355. Then, the altitude variable cannot be treated as a normal distribution.

Ground Speed (GS) behaves similar to a normal distribution. 95% limits are from 389 to 525 kts. The maximum value seems operationally valid, but the minimum value is an outlier. Lower values than -95% should be considered as outliers considering the statistical distribution.

The vertical rate cannot be modelled as a normal distribution. Values grouped by 25%, 50% and 75% match values near 0 ft/min. The reason is most aircraft perform cruise flights. Nonetheless, there are aircraft climbing and descending with a vertical rate of around $\pm$1500 ft/min. Therefore, the limits of the vertical rate could be from -1500 to 1500 ft/min. However, they demand further statistical analysis.

Figure 14 shows the data representation of the three variables analysed for LSAZM567 airspace.

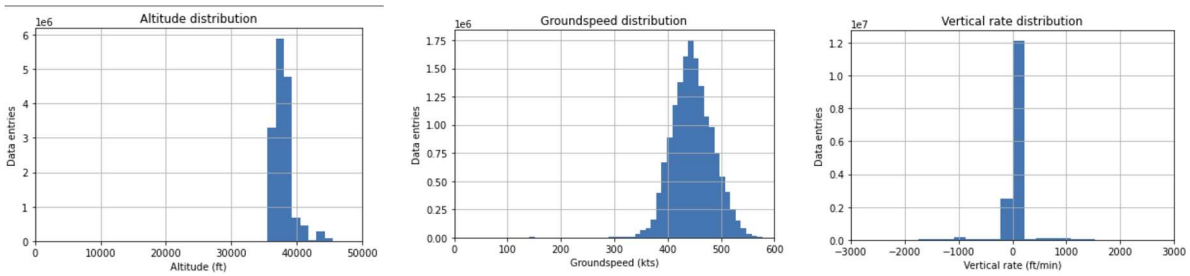

Figure 14 Data representation of LSAZM567 airspace: left) altitude, medium) groundspeed and right) vertical rate.

An individual analysis of the raw database for the first week (from 20 to 26 June 2019) has also been carried out. The goal is to analyse differences between the days. Table 4 gathers the statistical data.

		Week 1	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7
ADS-B data		4612757	610461	669015	736336	708139	627106	643869	617673
Trajectories		5587	685	748	830	855	763	797	738
Aircraft pairs		$2e^7$	$1e^6$	$1e^6$	$1e^6$	$1e^6$	$1e^6$	$1e^6$	$1e^6$
Altitude	Mean	38027	38007	38012	37925	38048	38093	38013	38108
	SD	1640	1561	1667	1502	1870	1686	1504	1642
Ground Speed	Mean	449	447	444	449	455	449	449	450
	SD	29	31	30	20	43	32	20	19
Vertical rate	Mean	12	12	10	13	12	14	8	12
	SD	257	248	260	265	249	265	241	268

Table 4 Statistical data of LSAZM567 airspace for the first week of AIRAC Jun 2019.

As can be extracted from Table 4, the air traffic distribution is similar for all days. There is an upturn in air traffic on days 3 and 4 compared with the rest of the days, and day 1 presents the lowest traffic. The main feature is that there are no significant operational differences because they are quite similar regarding altitude, GS and vertical rate. Besides, there are no statistical differences with the raw database.

4 Feature engineering

This section describes the modifications and adaptations required by OpenSky's raw data to constitute the final database. The database building for task 3.2 demands the resolution of two complex problems. The first problem refers to the constitution of aircraft pairs and their scalability when the number of combinations increases. The second problem refers to the difficulty to simulate SI for those aircraft pairs. This section describes the approach developed by the authors to solve both problems. Figure 15 shows the flow of this process to reach a database suitable for ML.

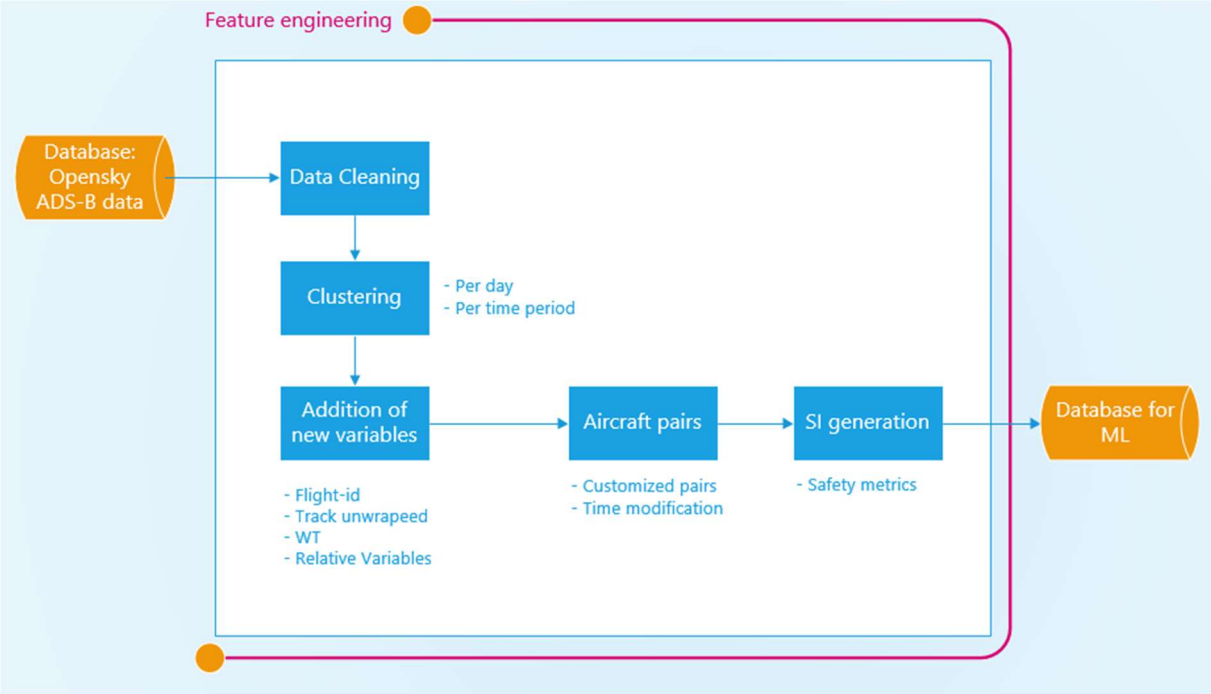

Figure 15 Flow of feature engineering process.

All of the steps in Figure 15 are described throughout this section. Finally, the primary conclusions and statistical analysis of the databases are presented.

4.1 Data cleaning & clustering

The first step with the path raw data is to clean them and to remove trajectories that have errors. The process of data cleaning is as follows:

- Filtering of data duplicity: trajectories duplicated have been removed by applying the function *.drop_duplicates()* from the OpenSky library.
- Erroneous paths: those trajectories with invalid ADS-B data have been removed by applying the function *.clean_invalid()* from the OpenSky library. This function detects trajectories with at least 10 ADS-B erroneous data.
- NaN or missed data: missed data with the average value based on the previous and subsequent ADS-B data has been replaced by the function *SimpleImputer()*. This error arises because the ADS-B signal is deficient, missing some values such as velocity, altitude or position.
- Outbound paths: trajectories that do not belong to the airspace boundaries of LSAZM567 have been removed from the database.

Upon further applying this data cleaning, it is assumed the database's data quality is sufficient. Although it is known the trajectories can present some inconsistencies, no reliability analysis of trajectories is performed. Further work should focus on ensuring the reliability of the trajectories extracted and cleaned from OpenSky. For this work, it is assumed the ADS-B trajectories present enough reliability and accuracy to be used.

Also, it has been performed temporary clustering to handle a huge database:

- Clustering by day: the database has been filtered to cluster the trajectories based on an operational day. This clustering has been performed to reduce the number of trajectories and computational requirements. The function *filteringdays()* has been created to this end.
- Clustering by hours: the database has been filtered cluster the trajectories based on the operational hours. This clustering has been performed to reduce the number of trajectories and computational requirements. The trajectories are grouped by temporary periods where the meteorological and operational conditions could be the same. The function *groupbyhours()* has been created to this end.

One day is split into 9 time periods in this work, although the user can fix those values.

4.2 Addition of new variables

The next step is to generate new information that is not directly included in the raw data. This information is a modification of the raw data to build the final database.

Flight identification

In the raw database, there are several trajectories with the same callsign. This repetition is a limitation to the process because it is necessary to have individual trajectories. To identify the trajectories correctly, the function *flight_id()* from OpenSky is applied.

Unwrap angles

To avoid problems with interpreting the aircraft courses between 0º to 359º, the function *unwrap()* from OpenSky is applied.

Wake-turbulence category

ADS-B data do not provide information about the wake-turbulence type or the aircraft model. The information provided by ADS-B data to identify the aircraft is the icao24 and the callsign. With this information can be identified the aircraft model but the wake-turbulence type not. For that purpose, it is necessary to access the ICAO's document 8643 [55]. This document links the information about the icao24 with the manufacturer mode, aircraft type and wake-turbulence type. OpenSky provides a database to deal with this information, although they do not directly link the icao24 with the wake-turbulence category. The function *add_WT()* has been developed.

A139	B407	B78X	C650	DH8D	F5EX	GL6T	P180
A20N	B733	BCS1	C680	DIMO	F900	GLEX	P28A
A21N	B734	BCS3	C68A	DR40	F9DX	GLF4	P28B
A306	B736	BD-700-1A11	C750	E170	F9EX	GLF5	P68
A318	B737	BE20	C77R	E190	F9LX	GLF6	PA46
A319	B737	BE40	CL30	E195	FA7X	GLID	PC12
A320	B738	C17	CL35	E35L	FA8X	H25B	PC21
A321	B739	C172	CL60	E50P	Falcon7X	H900	PC24
A332	B744	C182	CL61	E545	FOX	HUSK	PRM1
A333	B748	C210	CL64	E550	G150	J3	PULS
A339	B752	C25A	CL65	E55P	G200	KODI	R66
A343	B753	C25B	CL85	E75L	G280	LJ35	RF6
A346	B762	C25C	CLON	E75S	G2CA	LJ36	RJ85
A359	B763	C25M	CRJ-1000	EC20	G300	LJ40	RV7
A35K	B764	C510	CRJ2	EC45	G450	LJ45	S22T
A388	B772	C525	CRJ9	EVSS	G550	LJ55	TB20
AS50	B77L	C550	CRJX	F2EX	G650	LJ60	TLEG
ASTR	B77W	C55B	DC3	F2LX	G650ER	LJ75	
AT43	B788	C560	DG1T	F2TH	GA5C	MD11	
AT72	B789	C56X	DG80	F2TS	GL5T	MM16	

Table 5 Aircraft models identified in the LSAZM567 database.

However, not every aircraft model is included in the databases, or some are erroneously identified. Around 1000 icao24 identifiers were detected, and over 150 cannot be linked to its wake-turbulence category. A new wake-turbulence category (denoted as 'U') was added to group all of those aircraft. Further work should improve this process to avoid any aircraft identified as U wake-turbulence category. On the other hand, 157 aircraft models (see Table 5) were identified, and just 12 could not be identified based on the icao24.

4.2.1 Relative variables between aircraft pairs

The last information to include in the database is the outcomes from the operational analysis of aircraft pairs. The relative variables are detailed in Appendix A.2. They acquire different values depending on the Static or Dynamic mode. For the Static mode, they only consider the information at the initial instant when one aircraft pierces into the airspace. This information is only considered because it is when the prediction is performed. The Dynamic mode calculates the relative variables at each timestamp at which operational information is available. The mathematical description of the relative variables is:

- Horizontal separation ($s_{i,j}(t)$): horizontal separation at time t between positions of an aircraft pair:

$$s_{i,j}(t) = position_i(t) - position_j(t) \quad (5)$$

- Vertical separation ($\Delta h_{i,j}$): Altitude separation (h_i, h_j) between an aircraft pair.

$$\Delta h_{i,j}(t) = h_i(t) - h_j(t) \quad (6)$$

- Course ($\gamma_{i,j}$): Course that links the position between an aircraft pair.

$$\gamma_{i,j}(t) = \gamma_i(t) - \gamma_j(t) \quad (7)$$

- Track variation ($\Delta\theta_{i,j}$): Track variation (θ_i, θ_j) between an aircraft pair

$$\Delta\theta_{i,j}(t) = \theta_i(t) - \theta_j(t) \quad (8)$$

- GS variation ($\Delta GS_{i,j}$): Difference in module of the GS (GS_i, GS_j) between an aircraft pair.

$$\Delta GS_{i,j}(t) = \left|GS_i(t) - GS_j(t)\right| = \sqrt{\left(GS_i^x + GS_j^x\right)^2 + \left(GS_i^y + GS_j^y\right)^2} \quad (9)$$

- Vertical rate variation ($\Delta\dot{h_{i,j}}$): Variation of the vertical rate ($\dot{h_i}, \dot{h_j}$) between an aircraft pair.

$$\Delta\dot{h_{i,j}}(t) = \dot{h_i}(t) - \dot{h_j}(t) \quad (10)$$

The relative variables are calculated with the function *relativevariablescalculation()*. The library *geographiclib* is used to calculate the headings and separations based on the aircraft positioning (from longitude and latitude). This library performs the calculations without transforming the aircraft positioning to Cartesian coordinates [56].

4.3 Generation of customised aircraft pairs

This section describes the process followed to generate customised aircraft pairs. The aircraft pair underlies the pillars to analyse SI. The aircraft pair generation is different for the Static and Dynamic mode. The reasoning is explained in detail in this section regarding the characteristic of the Static and Dynamic mode.

4.3.1 Generation of aircraft pairs for Static mode in the same temporary period

The Static mode performs SI predictions when one aircraft pierces in the airspace. The database must be developed according to this operational concept. The main problem to build a database with enough SI is real trajectories barely provide samples. It is required to modify the trajectories flown to force situations in which an SI would have emerged. The average flight time in LSAZM567 airspace is about 11 minutes; then, it is required to modify the airspace's entry time. To group in the same time period, the trajectories imply the possibility of emerging SI between aircraft pairs that would not have appeared under real circumstances. This temporary modification allows considering for each aircraft pairs multiple situations piercing the airspace. It only evaluates the aircraft that are within the airspace sector. Two functions have been developed to achieve this goal:

1) *Generatepairs_STATIC()*: it generates every aircraft pair to evaluate for the Static mode.
2) *DifferentEntryTime_STATIC()*: it modifies the aircraft's entry time as a function of the selected aircraft that pierces into the Static mode's airspace.

The control variable of both functions are $(t_{entry}^{init}, t_{entry}^{end})$. t_{entry}^{init} indicates the initial time an aircraft can pierce the airspace and t_{entry}^{end} indicates the last time an aircraft can pierce the airspace.

Suppose there is a sample of n trajectories that generates $(n-1)n$ aircraft pairs. For each aircraft i are generated $(n-1)$ aircraft pairs to evaluate. The aircraft i is selected to pierce the airspace at the fixed time t_{a_i} and the rest of the aircraft modifies their entry times at time t_{a_j} between the above limits. The temporary restriction is that the aircraft should be within the airspace when the aircraft i pierces the airspace, the limits t_{entry}^{init} and t_{entry}^{end} should ensure that condition. Aircraft outside the airspace that will pierce after the aircraft i are not considered. The new entry times are calculated randomly for each aircraft. Then, $n-1$ aircraft associated with aircraft i pierces into the airspace before the aircraft i and randomly between specific time boundaries. This process provides new conditions to ensure potential interactions between aircraft pairs in the airspace. Figure 16 represents the aircraft-pair generation for the Static mode.

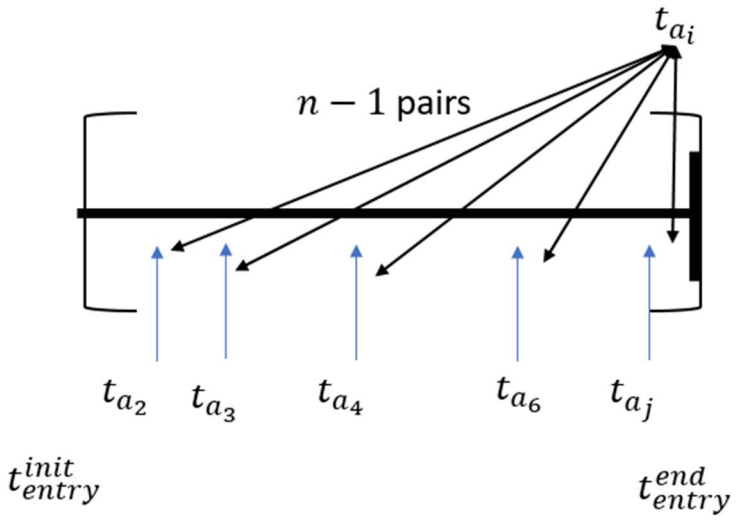

Figure 16 Aircraft-pair generation for Static mode.

Besides, this process is repeated for every aircraft considered in the same time period. The main characteristic of aircraft-pair generation is the process that generates for each aircraft a set of $n-1$ aircraft pairs.

4.3.2 Generation of aircraft pairs for Dynamic mode in the same temporary period

The Dynamic mode considers the aircraft's evolution within the airspace, not only when the aircraft pierces. It considers aircraft that pierces into the airspace later than the aircraft in consideration. In addition, the Dynamic mode considers a 4DT prediction is available to be used by the system. This work considers the ADS-B trajectories stored in OpenSky to be used as 4DT predictions. If the system would have more accurate predictions, the process will be the same.

The Dynamic mode requires a similar function to generate aircraft pairs with some modifications. In the Static mode, an aircraft i pierces the airspace in the time t_i and analyses the conditions of the different aircraft that are already within the airspace. This process is repeated for every aircraft, building the database with the information when aircraft j pierces the airspace. The Dynamic mode is broader because it does not consider only the aircraft within the airspace, but also the aircraft will pierce later (with temporary limits). Therefore, every aircraft modifies its entry time randomly between the temporary boundaries ($t_{entry}^{init}, t_{entry}^{end}$). Two functions have been developed to achieve this goal:

1) *Generatepairs_DYNAMIC ()*: this function generates aircraft pairs for the Dynamic mode.
2) *DifferentEntryTime_DYNAMIC()*: this function modifies the aircraft's entry time between some temporary boundaries for the Dynamic mode.

Similar to the Static mode, the control variable of both functions are ($t_{entry}^{init}, t_{entry}^{end}$). Suppose that there is a sample of n trajectories that generates $(n-1)n$ aircraft pairs. For each aircraft i are

generated $(n-1)$ aircraft pair to evaluate. The temporary restriction is that every aircraft should be in the airspace between the limits t_{entry}^{init} and t_{entry}^{end}. The new entry times are calculated randomly for each aircraft. This process provides new conditions to ensure potential interactions between aircraft pairs in the airspace, as Figure 17 represents.

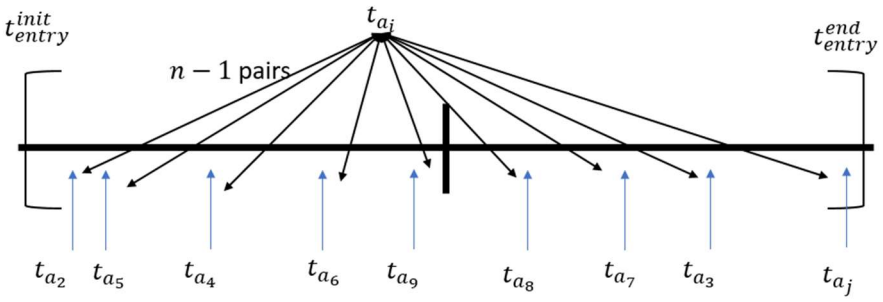

Figure 17 Aircraft-pair generation for Dynamic mode.

Each ADS-B trajectory receives a 4DT prediction based on similarity, as described in section 2.4.2. This 4DT prediction receives the same entry time as the 4DT to ensure compatibility between them. Finally, this process is repeated for every aircraft during the time period considered avoiding duplicities.

4.3.3 Features adding

The next step is to join the customised trajectories for every aircraft pair based on the timestamp. Then, the last step is to calculate the relative variables for each aircraft pair. There are different functions for the Static and Dynamic mode, although the principles are the same. The main difference between the modes is the notation and the information provided. The Static mode provides information only about when one aircraft pierces into the airspace, and the Dynamic mode provides information throughout the trajectory evolution. Table 6 shows the features included for each aircraft pair (i, j).

The function *generatorDFconflicts_STATIC()* associates the features of aircraft i when pierces the airspace and the features of aircraft j at the same timestamp. The function *generatorDFconflicts_DYNAMIC()* associates the features of aircraft i and the features of aircraft j at the same timestamp. This process is iterative for every timestamp that the aircraft coincide at the airspace. When both aircraft do not operate at the same timestamp within the airspace, the function fills the NaN gaps. Once finalised the whole process, NaN samples are removed. This process is repeated for 4DT predictions of the Dynamic mode. After the assembling, the relative variables between aircraft are calculated by the function *relativevariablescalculation*(). This function works for Static and Dynamic mode.

For aircraft i, j (22 features)	For each aircraft pair (6 features) in Static mode	For each aircraft pair (6 features) in Dynamic mode
Altitude	Initial horizontal separation	Var horizontal separation
Groundspeed	Initial vertical separation	Var vertical separation
Latitude	Initial azimuth variation	Var azimuth
Longitude	Initial ground speed variation	Var ground speed
Timestamp	Initial track variation	Var track
Track	Initial vertical rate variation	Var vertical rate
Vertical_rate		
Flight_id		
WT		
Track_unwrapped		

Table 6 Features addition by aircraft pair generation.

Figure 18 shows the data pre-processing process for the Static Mode. This process is the summarise of the different tasks performed previously. First, the ADS-B data is obtained for two aircraft. This ADS-B data is cleaned and joined to generate the aircraft pairs. Then, the relative variables are calculated and integrated into the label of each aircraft pair. Finally, this data can be provided to the ML model to predict new instances.

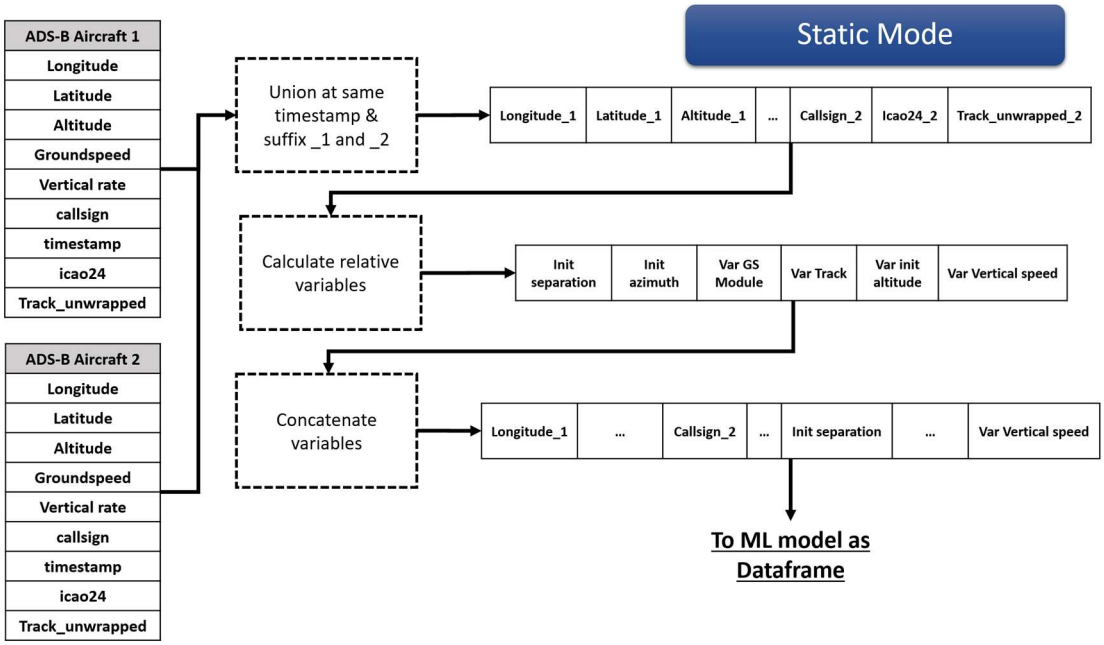

Figure 18 Data pre-processing process from ADS-B to ML model for the Static mode.

Figure 19 shows the data pre-processing process for the Dynamic Mode. This process is the summarise of the different tasks performed previously. First, the ADS-B data is obtained for two aircraft. This ADS-B data is cleaned and joined to generate the aircraft pairs. Then, the relative variables are calculated and integrated into the row of each aircraft pair. Simultaneously, similar trajectories are obtained from the ADS-B trajectory database to work as 4DT predictions for the aircraft pairs. Safety metrics are calculated for the 4DT predictions and joined with the real ADS-B information. Finally, this data can be provided to the ML model to predict new instances.

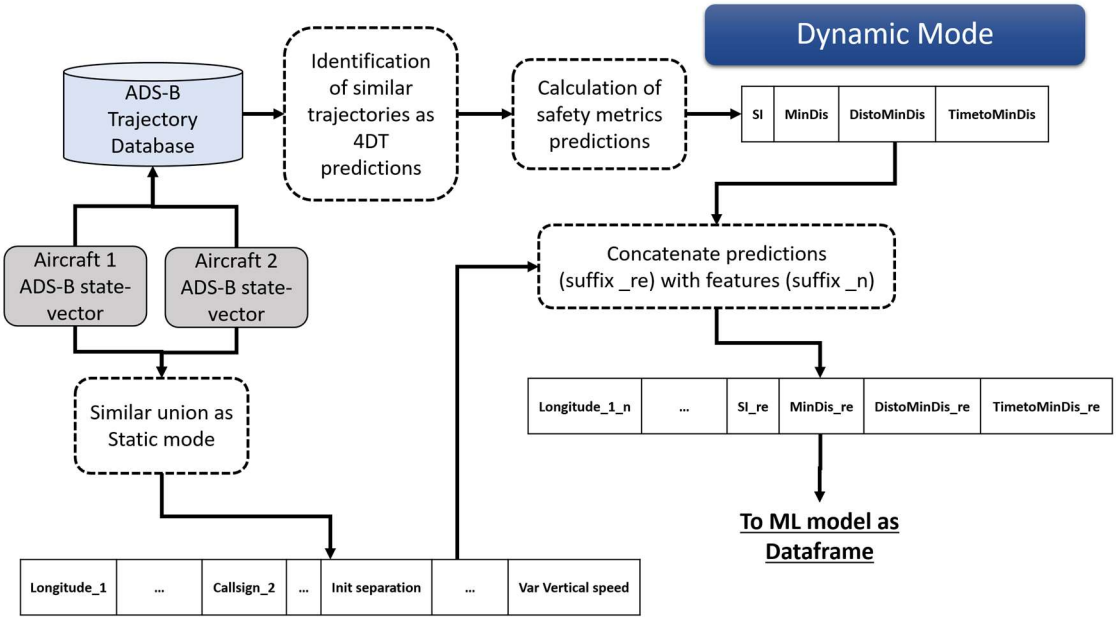

Figure 19 Data pre-processing process from ADS-B to ML model for the Dynamic mode.

4.4 SI and safety metrics analysis

Once the trajectories have been customised and paired, the next step is to analyse each aircraft pair's separation. These temporarily modified trajectories constitute a good standpoint to explore what could have happened if these trajectories were flown. The goal is to identify SI between aircraft pairs and the safety metrics of those situations. Finally, a database with enough SI samples is expected to be constituted for the ML algorithms application. Different sections are provided for each mode.

4.4.1 Separation analysis for Static mode

The Static mode builds a database in which each sample represent an aircraft pair. Then, separation analysis of Static mode must provide information about SI and safety metrics. This information about the evolution of the separation is added to the database for each aircraft pair. This isolated sample of each aircraft pair contains all information about the separation evolution throughout the airspace (safety metrics).

To this end, it is necessary to evaluate the horizontal and vertical separation throughout the trajectory. Currently, OpenSky's API [53] provides a function to calculate the evolution of the separation between aircraft denoted as *closest_point_of_approach*(). This function compares an aircraft pair's trajectories, calculating at each timestamp the horizontal and vertical separation. The function performs first filtering to consider whether the aircraft will cross with some horizontal and vertical separation previously indicated. The aircraft that do not fulfil this condition are discarded. Therefore, the control variables of this function are *horizontal_separation* and *vertical_separation*. These control variables do not match the horizontal and vertical separation minima (5 NM and 1000 ft). It is more convenient to select higher parameters to obtain a larger number of samples. In this work, those values were 100 NM and 5000 ft, respectively.

However, this function presents two limitations for this work: 1) it demands the knowledge of the ADS-B trajectory prior to evaluating the trajectories, and 2) this function only provides the information of the aircraft pairs that infringe the control variables, i.e., they do not provide information when the aircraft cross further than the control variables. This is the reason separations so high for the control variables have been selected. Therefore, this function can be used for the Static mode but not for the Dynamic mode.

The output of the function *closest_point_of_approach()* is a data frame that provides the information about the lateral and vertical separation for each timestamp. With this data frame it is possible to calculate the safety metrics defined in section 2.2.2. This process is not straightforward and has required the development of several functions to extract the information. The functions *pairswithconflict(), conflictdetection()* and *mod_conflicts()* provide the safety metrics for each aircraft, classifying them as SI or no SI. This information is included in each sample and constitutes all the ML algorithms' information: features plus labels.

flight_id_1	flight_id_2	SI	MinDis	TimeMinDis	DistoMinDis
VLG1885_726	2JFAA_000	0	14.143964	228	28.686233
VLG1885_726	AFR23TN_015	0	65.034417	1	0.138735
VLG1885_726	AFR44HG_017	1	6.787403	266	33.426594
VLG1885_726	BAW2563_062	0	57.686744	502	63.080780
VLG1885_726	BAW46PX_068	0	26.385792	160	20.178851

Figure 20 Example of safety metrics results.

Figure 20 shows an example of the safety metrics obtained for each aircraft pair. There are aircraft pairs that constitute SI (1) and no SI (0). Aircraft pair of row 3 reaches a MinDis of 6.8 NM in 33 NM or 266 seconds. The separation analysis is performed for all aircraft pairs and repeated for every temporary period and day. When new trajectories are available, the database will be feed with new situations that the ML model could consider.

The main downside is the high computational time required to evaluate the aircraft pairs. For LSAZM567 airspace, it was needed to cluster the aircraft based on the days and different temporary periods. Table 7 shows the results of one-week simulations for the Static mode.

Number of aircraft	5849
Number of pairs generated	551275
Number of pairs evaluated	507731
Number of SI (%)	24374 (4.8%)
Number of conflicts (%)	10188 (2%)
Computational time (hours)	634

Table 7 Results of one-week simulations for Static mode.

The number of aircraft for one week through the airspace LSAZM567 is 5849. The number of pairs generated is far from the real combinatorial number of aircraft pairs because the simulations were performed for specific time periods. The number of aircraft varies during the time periods from 80 to 120. There is a difference between the number of pairs generated and evaluated (around 8%). There are aircraft pairs not assessed because they do not share the LSAZM567 airspace during the same time period. The number of SI detected is about 5% and the number of conflicts about 2%. These results confirm the strong imbalance of the database.

4.4.2 Separation analysis for Dynamic mode

The Dynamic mode updates the information on the safety metrics throughout the 4D trajectory. It also provides information about safety metrics expected from the 4DT predictions. The Dynamic mode provides for each aircraft pair so many samples as timestamps recorded in the database.

Conversely to Static mode, the function *closest_point_of_approach*() developed by OpenSky is not recommendable for this analysis because it does not store the whole set of points throughout the trajectory. Hence, a new function called *conflict_detection()* has been developed. The concept of the function is similar but trying to solve some drawbacks and optimising the separation analysis. The function evaluates the separation for each timestamp between an aircraft pair and analyses the separation evolution metrics. The main difference is this function stores all the information generated for each aircraft pair throughout the airspace. The control variables are now three: *horizontal_separationminima*, *vertical_separationminima and distance_interest*. The values are 5 NM, 1000 ft and 10 NM and can be specified by the ANSP. The distance of interest is the fixed value that characterises a situation as SI or no SI.

This analysis is performed for every aircraft pair considered in the same time period. Based on the trajectories, it is calculated the minimum separation reached, distance and time to MinDis. Based on the minimum separation position, it can be calculated the distance to this position and the time. These safety metrics are added to the aircraft pair database and classified as SI and no SI. Therefore, safety metrics acquire a dynamic value depending on the timestamp the situation is evaluated. Figure 21 shows an example of the results.

	flight_id_1_n	flight_id_2_n	timestamp_n	altitude_1_n	altitude_2_n	SI_n	MinDis_n	distomindis_n	timetomindis_n
0	AFR82PQ_003	AFR94LA_004	2019-06-20 01:00:40+00:00	36025.0	39000.0	0	85.797243	31.771920	276
1	AFR82PQ_003	AFR94LA_004	2019-06-20 01:01:00+00:00	36025.0	39000.0	0	85.797243	29.445700	256
2	AFR82PQ_003	AFR94LA_004	2019-06-20 01:01:40+00:00	36025.0	38975.0	0	85.797243	24.824295	216
3	AFR82PQ_003	AFR94LA_004	2019-06-20 01:02:00+00:00	36025.0	38975.0	0	85.797243	22.534665	196
4	AFR82PQ_003	AFR94LA_004	2019-06-20 01:02:20+00:00	36025.0	38975.0	0	85.797243	20.257336	176

Figure 21 Example of results of the function *conflict_detection*().

This figure shows an example of the function *conflict_detection()* with the results for the aircraft AFR82PQ_003 and AFR94LA_004. Although it does not appear in the figure, the relative variables described in section 4.3.3 are stored. This calculation is an iterative process that must be repeated for the temporary periods the database is split into and the number of days considered. The database could be increased when new aircraft were introduced. Lastly, the computational time required is lower than the Static mode, but it is still very high. Table 8 shows the results for the whole simulations performed for one week.

The number of pairs generated is far from the real combinatorial number of aircraft pair because the simulations were performed for specific time periods. The number of aircraft pair varies during the time periods from 80 to 120. The number of SI detected is about 18% and the number of conflicts about 4%. These values are higher than the Static mode, although they are far from constituting a balanced database. These results represent the problem of generating separation infringements in spite of performing specific simulations. To increase these rates, the solutions would be to perform simulations only for pairs of air traffic flows (something studied in this work, but it was not possible due to the inefficiencies of clustering algorithms) and reduce the size of the airspace to consider. Additionally, the computational time gives an idea of the problem due to the high number of hours required.

Number of aircraft	5849
Number of pairs generated	551275
Number of pairs evaluated	183762
Number of SI (%)	1235648 (18%)
Number of conflicts (%)	266323 (4%)
Computational time (hours)	420

Table 8 Results of the simulations for the Dynamic mode.

4.5 Final database

This section shows the outcomes of the final database that the ML algorithms will learn. Every simulation has been performed in a computer Intel® Core™ i5-6600 CPU @ 3.30GHz, RAM 8,00 GB, 64 bits.

4.5.1 Database for Static mode

The Static mode database has been generated based on a sample of 15 days between 20 June and 3 July 2019. It was not feasible to analyse more traffic due to the high computational cost in terms of workload and time. The total time to perform all the simulations of the 15 days split into the temporary periods (without considering erroneous simulations) was around 1200 hours (50 days). About the size of the database:

- Database weight: around 520 MB.
- Number of samples: 520605 samples. Each one of the samples represents an aircraft pair situation.

Regarding safety metrics results, Figure 22 shows the distributions obtained:

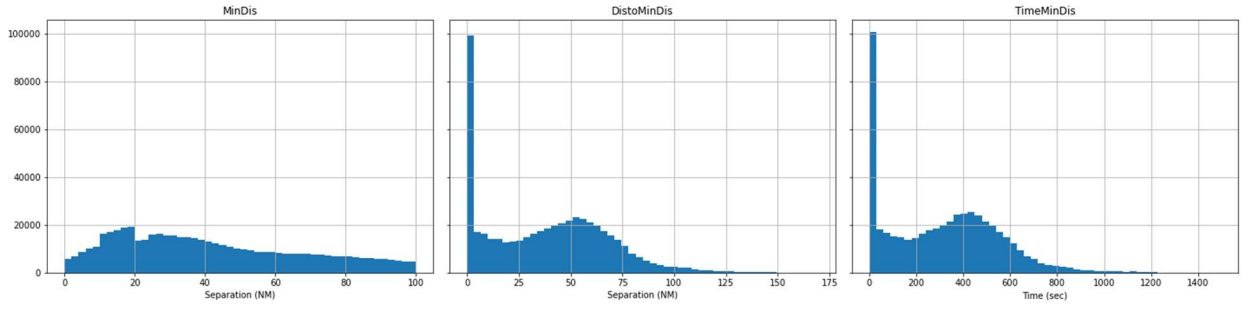

Figure 22 Histograms of safety metrics for the Static database.

The MinDis provides a smooth distribution for every sample from 0 to 100 NM, although there is a slight increment from 10 to 20 NM. It can be identified that the number of samples SI and no SI is

entirely unbalanced. DistoMinDis provides information about the distance that must cover the aircraft that pierces the airspace to reach the MinDis. The vast majority of the samples is located near zero. This implies that a large number of aircraft are moving away in the airspace. TimetoMinDis provides information about the time required to reach the MinDis. It presents a similar distribution to DistoMinDis, in which a large number of aircraft are moving away in the airspace.

4.5.2 Database for Dynamic mode

The Dynamic mode database has been generated based on a sample of 15 days between 20 June and 3 July 2019. It was not feasible to analyse more traffic due to the high computational cost in workload and time. The total time to perform all the simulations of the 15 days split into the temporary periods (without considering erroneous simulations) was around 850 hours (36 days). It has been achieved a significant reduction in the computational time compared with the Static mode. About the size of the database:

- Database weight: 4.2 GB. This database is four times bigger than the Static database.
- Number of samples: 6753283 samples. Each one of the samples represents an aircraft pair situation at a specific timestamp. The number of samples is over ten times the Static database.

Regarding safety metrics results, Figure 23 shows their distributions:

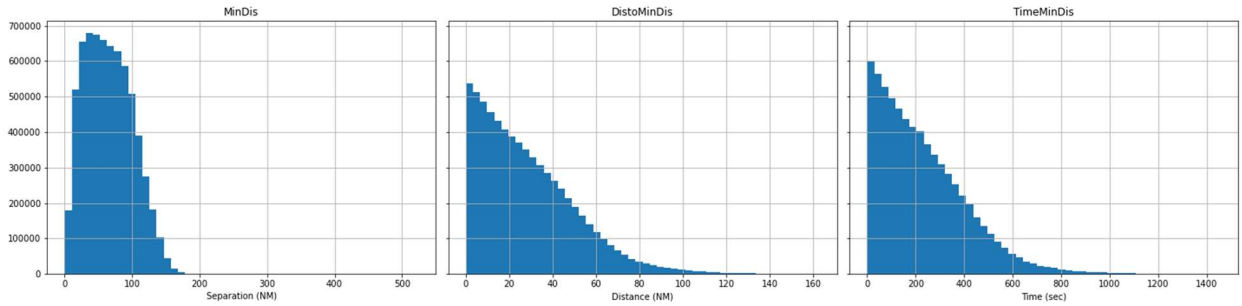

Figure 23 Histograms of safety metrics for the Dynamic database.

The range of MinDis distribution is from zero to over 150 NM. The larger density is located between 10 to 20 NM, similar to the static database. There is also a high imbalance between SI and no SI samples. However, the number of SI samples is about 18%, while the Static mode was 5%. Both DistoMinDis and TimetoMinDis provide similar distribution between them. The density decreases softly as the variable increases. Besides, the range is identical to the Static mode. Figure 24 shows the results of the safety metrics for ADS-B and resampled trajectories jointly.

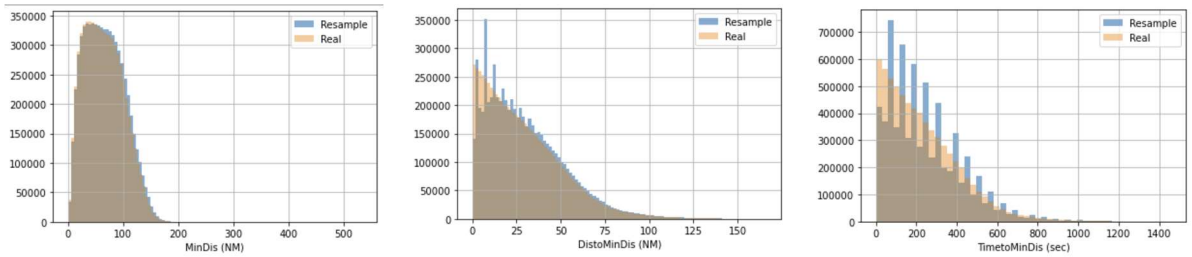

Figure 24 Comparison of safety metrics for ADS-B and resample database for Dynamic mode.

The distribution of both databases is similar, as was expected. The main difference is that ADS-B data presents more continuous behaviour while the resampled trajectories provide peaks throughout the distribution. Those peaks focused on the update period of the resample technique selected.

4.6 Problems identified during the database building

This section brings to light the main problems or issues identified during the data treatment from OpenSky, separation analysis and building of the final databases:

- Reliability of ADS-B data. This is a problem not analysed directly in this work, but it affects the database integrity. It has been identified non-reliably ADS-B data during the process, such as null-velocity or altitude over FL1200. This type of data was discarded, but it is necessary to provide a completely reliable database.
- Trajectory data-size. Typically, trajectories provided by OpenSky have information about each second. This means that one trajectory that flights through LSAZM567 over 10 minutes provide 600 ADS-B data. However, there are trajectories with more than 2000 ADS-B data. This issue affects the computational time required to evaluate the possible combinations.
- Airspace size. The filtering of the trajectories to the LSAZM567 airspace was not perfect. Trajectories identified outside the airspace boundaries were removed, although it would be better to improve the filtering process.
- Lack of separation infringements. The current status of the trajectories does not work for the analysis of conflicts based on ADS-B data. This is a typical issue because the number of conflicts is deficient. It would be desirable to develop a public database with enough conflicts or SI that researchers could handle.
- Aircraft clusters. It has been necessary to perform aircraft clustering based on days and temporary periods. The goal was twofold: to reduce the computational time and consider aircraft operating with similar weather conditions.
- Weather. This work did not consider weather variables (such as temperature, wind, humidity, etc.) as features. This information was implicitly contained in the ADS-B data. Further work could introduce this type of variables to improve the database.
- Conflict at time $t = 0$. By modifying the entry time of the aircraft to operate in a similar time period, conflicts between aircraft at time $t = 0$ appeared. In other words, an aircraft pair pierces the airspace infringing the separation minima. This type of situations has been discarded because it should not exist in reality. It would imply that the ATCOs of the previous sector would not have done their labour correctly.
- Computational time. One of the main limitations of this work has been the computational time and workload required to perform the simulations.

5 ML approach

One of the current technological developments in the implementation of data-driven techniques is ML techniques. ML is one of the data-driven techniques being investigated in different areas. ML's goal is to understand the structure of data and fit that data into models that can be used to perform predictions. This work focuses on introducing ML techniques to help the air traffic controllers during monitoring tasks, particularly in conflict detection. The goal of this section is to describe the approach used in this work by using ML techniques. Section 6 provide the results of the application of these techniques.

ML models are already implemented in different platforms of computer science. The most extended are Python® and MatLab®, although others can be used. Each one of these platforms is developing libraries for different ML algorithms. Therefore, the application of ML algorithms has become a user-friendly task. This work aims not to use all ML algorithms but to analyse the feasibility of their usage.

How is evaluated the performance of the ML model? The answer is different for classification and regression problems because the metrics are different. Classification metrics focus on assessing the level of accuracy in the classification predictions. It is important to note that the whole dataset's accuracy is not the best metric for unbalanced problems. The most typical classification metrics are accuracy, recall, precision and F2. Regression metrics are different because they evaluate the ML model's performance based on the statistical differences between the predicted and the measured numerical value. The most typical regression metrics are the Mean Average Error (MAE) and Root Mean Squared Error (RMSE).

5.1 Generalization problem: underfitting and overfitting

An ML model's performance is evaluated by splitting the dataset into two sets: training and testing (or validation). In this way, the ML model is trained with the training set and compared its performance with new testing instances. Typically, the ML model's performance in the training set compared with the testing does not match. This is one of the ML techniques problems once they are trained in a specific dataset. A dataset is provided to the ML algorithm so it can learn the underlying structures to perform predictions. Two situations can occur overfitting or underfitting [31].

Overfitting appears when ML's model is too complicated relative to the amount and noisiness and fits the training data. Nonetheless, it worsens with the validation set. The possible solutions are to simplify the ML model by selecting one with fewer variables, gathering more training data or reducing the train/test split, or reducing the noise or impurities in the training data (removing data errors and outliers). Regularization is the process of constraining the ML model to simplify it and reduce the risk of overfitting. The hyperparameters of the ML model control regularization. The hyperparameter is a parameter of the ML algorithm (not a feature) and must be specified prior to the training process. The error rate on new cases is called the generalization error. The error rate is calculated by evaluating the ML model on the validation or testing set. This value is calculated by the difference between the training and the testing set metrics, and it should be the lower as possible. If the training error is low and the testing error is high, the model is overfitting the training data.

Underfitting is the opposite of overfitting. Underfitting occurs when the ML model is too simple or not enough complex to learn the underlying structure or patrons of the data. Underfitting provides poor results in the different metrics of the ML models. It should be solved by selecting more complex models (increasing the number of parameters), providing more features to the learning algorithm or modifying the regularization hyperparameters.

Cross-validation (CV) is the most extended technique to avoid this issue and validate the model's real performance. This work uses the K-fold CV technique. This technique divides the whole dataset into k folds. One of the folds is used as a validation set, and the rest constitutes the training set. This process is repeated, interchanging the different groups as the validation set. Hence, it can be evaluated an ML model in so many data sets as k-folds are. Hence, CV avoids overfitting because a different model is trained in one particular training set and evaluated for each split's specific validation set. Lastly, one important factor to consider is CV must be performed applying the random distribution of samples at each fold (shuffle) and keeping the dataset class's statistical distribution (stratification). Figure 25 shows an example of a 5-fold CV.

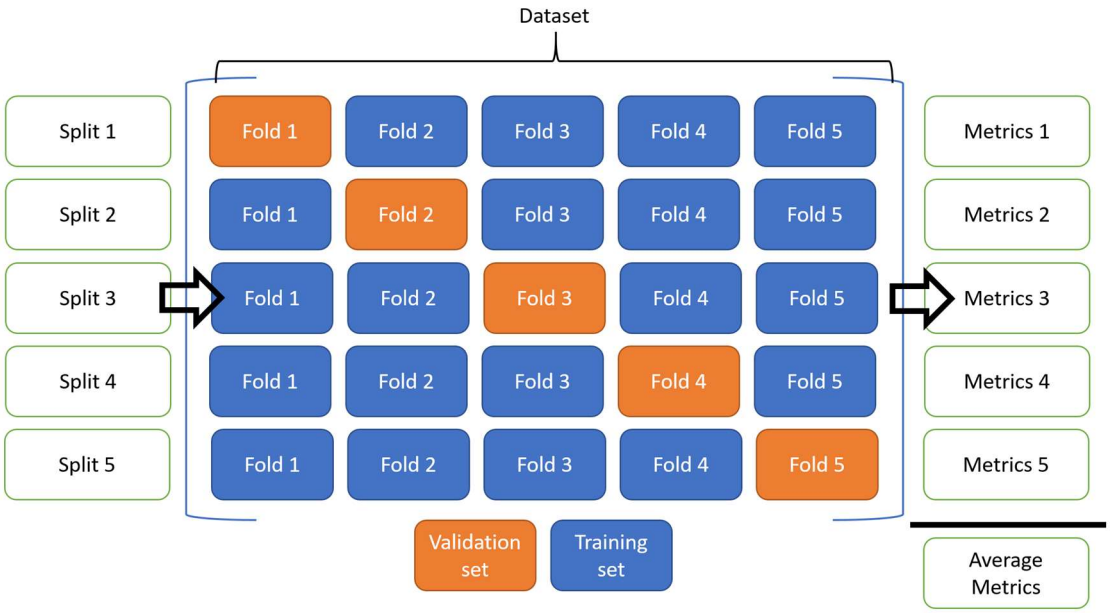

Figure 25 Cross-validation structure.

5.2 Unbalanced classification problems

As previously mentioned, unbalanced classification problems are those that one category predominates over the others, i.e., the classes are not represented balanced or equally. This is a typical problem in classification problems because the equal representation of different classes does not occur in real-life issues. Conflict detection is one of this type of problems similar to fraudulent transactions or cancer detection. Conflicts appear rarely and, although we have performed simulations to generate SI between trajectories, the SI rate is reduced (5% of samples for Static mode and 18% for Dynamic mode).

The issue of working with unbalanced sets is called the 'accuracy paradox' [57]. The accuracy paradox is the paradoxical finding that accuracy is not a suitable metric in predictive analytics. A simple model may present a higher accuracy level by predicting all samples as the majority class. This is the reason accuracy cannot be the only metric for classification problems. Recall, precision, and F1 focus on the minority class, representing this problem's core [58]. Achieving 99% classification accuracy without considering recall or precision behaviour may be trivial on an unbalanced problem.

There are several techniques to tackle the problem of unbalanced classification [59]:

- Undersampling. This technique implies selecting a subsample of the majority class such that the size matches the minority class. This technique's primary problem is that much information is lost, and it could imply scalability issues with the dataset.
- Oversampling. This technique produces artificial or synthetic instances of the minority class until the proportion is balanced. This technique is popular, but the problem is about the way of making the instances.
- One Class Learning. This technique considers that the real points compose the minority class. The majority class is random noise that produces disturbances in the dataset.
- Cost-Sensitive techniques. The cost function analyses the impact of misclassifying the samples in the dataset. Typically, the cost function equally considers the cost of different misclassifying classes. However, the weights for the cost function can be modified in order to increase the impact of misclassifying the minority class. Then, the algorithm can reduce the misclassification of the minority class. For instance, if the class distribution is 0.99/0.01 for the majority and minority classes, it can be balanced by considering a penalty of 0.99 for errors made with the minority class.

In this work, undersampling will be used to specify the best performance limits in terms of recall, precision, and F1 metrics. Cost-sensitive training techniques are applied to improve the results for unbalanced problems potentially.

5.3 Preparation of the dataset to ML algorithms

ML techniques should not be directly applied to the database without pre-processing the data. This pre-processing or data preparation aims to adequate the data into formats that ML algorithms could handle. The primary purpose of data preparation is to manipulate, modify and transform the data that will be analysed so that the information can be accessible for the ML algorithm. The tasks applied during the data pre-processing are the following:

- **Training\Validation\Testing split**: The dataset is divided into the training dataset, the validation dataset and the testing dataset. The validation dataset serves as a proxy for new data. The validation dataset (also known as the hold-out set) is not used in model training and hence can be used to evaluate metrics and determine if the model has overfitted the data. It is also used to optimise the ML model. Then, the optimised model is trained with the training and validation dataset and evaluated with the testing dataset. The goal is to build a model that generalizes well to unseen data, in this case, the testing dataset. For every experiment, the databases were split into three sets: Training set (70% of 90% of the whole database to train

the model), validation set (30% of 90% of the entire database to optimise the model) and testing set (10% of the whole database to analyse the final performance of the model).

- **One Hot Encoding**: Numeric or categorical variables can group features. ML algorithms cannot handle categorical data, and it must be transformed into numeric values. In One Hot Encoding, each categorical level becomes a separate feature containing binary values (1 or 0). For instance, wake-turbulence type can acquire four categories: Light, Medium, Heavy or Unknown. Then, one feature is transformed into new four features. This process means new variables are generated for each one of the values of the categorical variables.
- **Normalization**: Normalization rescales the values of numeric columns in the dataset. It does not distort differences in the ranges of values. There are several methods available for normalization, although, in this work, *zscore* has been selected. *Zscore* normalizes the numerical values of each variable by fitting to a normal distribution.
- **Shuffling**: It distributes the samples randomly into the training, testing and validation sets.
- **Stratification**: In addition to distributing the samples randomly, the stratification process allows distributing them to keep the classes' statistical distributions. The stratification variable considered in this work is the safety metric SI.

5.4 ML experiments

There is no one-size-fits-all value that ensures that an ML algorithm makes good or poor predictions. Regarding classification models, the 100% rate in the different metrics is an academic goal but rarely achieved. Typically, metrics over 80% can mean good or acceptable performance, and over 90% mean represent outstanding models. These values must be adapted to each problem because it depends on the ML model finality.

Classification models reach their best performance when the database is equally distributed among the classes. Then, it has been considered an ML model to provide reference values or best user metrics based on an ad-hoc 50% balanced dataset. Although it does not exist undersampling techniques for regression problems, it has been evaluated as well as the regression models for this experiment. Besides, it has been considered a Hybrid model in which a filtering process based on operational restrictions is introduced to reduce the dataset's strong imbalance. Therefore, three experiments have been analysed for classification and regression models:

1. Experiment 0 or Reference model. ML model with undersampling techniques. This model represents the theoretical best model performance for classification because the database is balanced.
2. Experiment 1 or Pure model: ML model applied to the whole database without any modification.
3. Experiment 2 or Hybrid model: ML model applied to a database previously filtered. The filtering is performed based on operational restrictions. The database only contains aircraft pairs that cross with horizontal separation minor than 20 NM and vertical separation minor than 1000 ft. This experiment's main problem is it demands implementing the aircraft pair's previous filtering during the implementation process.

Aiming to analyse different ML algorithms and to obtain the best one for each one of the three experiments, Figure 26 shows the steps followed during the ML process:

1. Analysis of different ML algorithms. The first step is to evaluate the performance of different ML algorithms in order to identify the ML model that provides better results. It has been assessed 15 ML algorithms for classification and 17 algorithms for regression.
2. Feature selection. The feature selection aims to identify the influence of the different features that constitute the database. The feature selection is performed based on graphical analysis and Recursive Feature Elimination (RFE) with CV [60]. RFE analysis evaluates which are the impact of the features on the model accuracy. Finally, features that do not influence the ML model are removed from the database.
3. ML algorithm optimisation. An optimisation process to improve the selected ML algorithm's performance is performed based on a hyperparameters grid search. Hyperparameters can be defined as the settings of an algorithm that can be adjusted to optimise the model performance. Hyperparameters must be defined in advance of the training process of the model. The metric to optimise is different for the classification and regression problem.
4. ML model finalisation. Once the ML algorithm is optimised, it will be finalised by training the ML algorithm with the whole dataset (training and validation set). The finished model represents the model that will be implemented in further work.

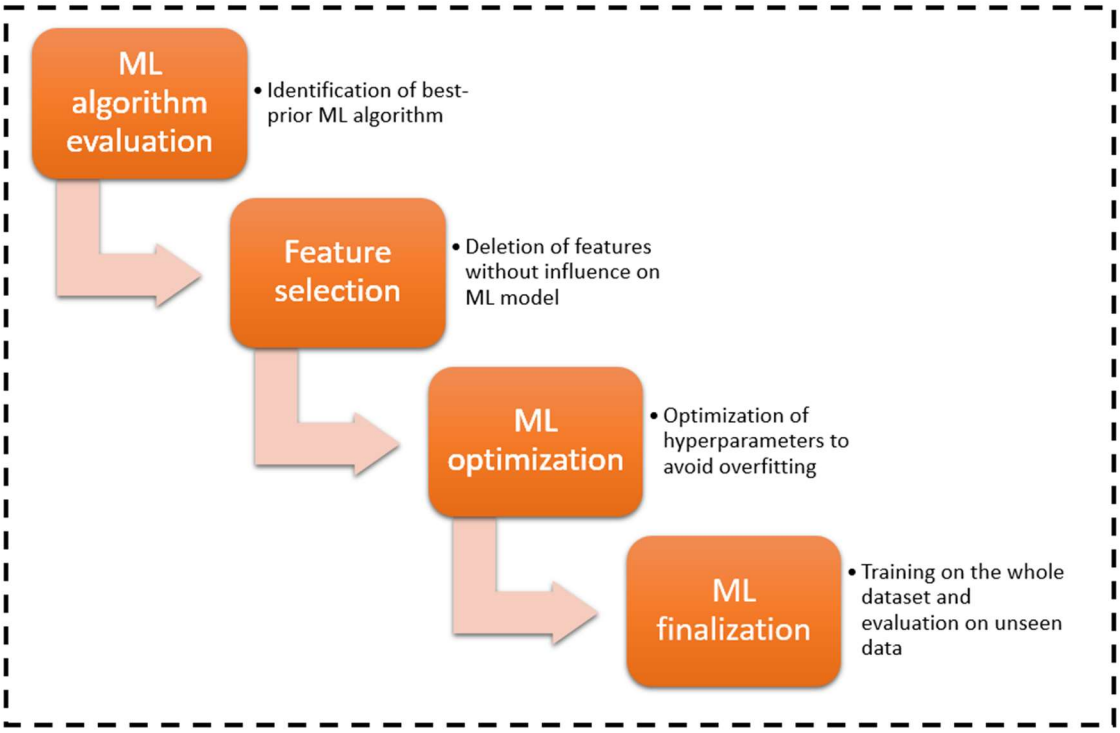

Figure 26 ML process for experiments.

6 ML Results

6.1 ML predictor for the Static Mode

This section provides the results of the Static mode. This mode is characterised because it gives a static SI prediction when one aircraft pierces into the airspace. As explained in section 5.4, three experiments have been carried out. The Reference model provides the theoretical best result (balancing SI and no SI samples). The Pure model considers the whole dataset, and the Hybrid model performs an operational filter to focus on aircraft pairs that approach a distance lower than 20 NM and 1000 ft. Table 9 shows the number of samples for each experiment; in parenthesis, the SI/no SI samples rate.

	Training set	Validation set	Test set
Reference model (50/50)	77246	33106	12262
Pure model (5/95)	792033	339444	125720
Hybrid model (35/65)	112463	48199	17852

Table 9 Number of samples of each experiment for the Static mode.

Appendix B details the process to obtain the results of the Pure model for classification techniques.

6.1.1 ML results for classification

ML classification techniques perform prediction whether an aircraft pair is SI or not, and the probability of this prediction. The issue for the classification problem is the strong imbalance of the database; the database only contains 5% SI samples.

The first step was to analyse different ML algorithms' performance, aiming to identify the best one. Accuracy is not a valuable metric for unbalanced problems because they focus on the predominant class. Then, the main metrics to analyse are recall, precision and F1. The three experiments agreed that ensemble models provided the best results. For each experiment, the three top ML models vary between Random Forest, Extra trees, Decision Tree and Extreme Gradient. Random Forest provided the best results for the reference and Hybrid model, although the decision tree provided the Pure model's best results. The models were evaluated based on stratified CV techniques to avoid overfitting on the results. The values obtained with the training set were compared with the validation set, which provided similar results.

The second step was to perform the feature selection. Three experiments provided similar results, although with some differences:

- Reference model. The most influential features were initial separation, initial azimuth and tracks; they encompass almost 40% of the ML importance. On the other hand, RFE confirmed

that Wake-turbulence type and vertical rate variables should be discarded due to their low influence.

- Pure model. The most influential features were initial separation, initial azimuth, altitude variation and tracks; they encompass almost 50% of the ML importance. On the other hand, RFE confirmed that Wake-turbulence type and vertical rate variables should be discarded due to their low influence.
- Hybrid model. The feature importance is shared between almost ten features, although the most influential features were initial separation and azimuth (20% of the ML importance). On the other hand, RFE confirmed that Wake-turbulence type, vertical rate and altitude variables should be discarded due to their low influence.

The three experiments concluded that initial separation, azimuth, altitude variation, and tracks were the most influential. The least influential features were the wake-turbulence type and vertical rate variables. Upon further removing the least influential variables, results confirmed that models with minor features provided worse values. The performance metrics worsen their value by about 1% with feature removing. This worsening implies that removing the least influential features on the model is feasible and can reduce computational cost.

The next step was the optimisation process of the ML algorithm. The goal of this process was to optimise the classification metrics F1 and recall of the model. The optimisation process has been performed, applying cost-sensitive techniques to the Pure and Hybrid model. Cost-sensitive learning takes the costs of prediction errors into account during the training process. The model penalizes misclassification errors from the minority class more than the majority class. Table 10 shows the different optimised models' results on the training set by applying the 5-fold CV.

Experiment	Model	Optimizer	Accuracy	AUC	Recall	Precision	F1
Reference model	Non-optimised Model	-	0.920	0.971	0.945	0.888	0.920
	Optimised Model	Recall	0.912	0.967	0.953	0.880	0.915
		F1	**0.921**	**0.972**	**0.949**	**0.900**	**0.923**
Pure model	Non-optimised Model	-	0.958	0.780	0.570	0.585	0.577
	Optimised Model	Recall	0.960	0.800	0.561	0.602	0.576
		F1	0.960	0.801	0.558	0.608	0.580
	Optimised Cost-sensitive Model	Recall	0.406	0.715	0.927	0.071	0.132
		F1	**0.958**	**0.801**	**0.623**	**0.559**	**0.589**
Hybrid model	Non-optimised Model	-	0.813	0.886	0.606	0.800	0.690
	Optimised Model	Recall	0.809	0.886	0.653	0.768	0.692
		F1	0.814	0.889	0.645	0.774	0.704
	Optimised Cost-sensitive Model	Recall	0.510	0.723	0.912	0.12	0.152
		F1	**0.815**	**0.889**	**0.696**	**0.751**	**0.722**

Table 10 Metrics for the different optimised ML classification models for the Static mode.

The main conclusions about these results are:

- The optimisation process improved the initial metrics, although these improvements were reduced up to 5%. This implies that identifying the correct model is crucial because the subsequent optimisation process does not improve its metrics.
- The implementation of cost-sensitive techniques could achieve this 5% improvement. Otherwise, the gains are reduced up to 2%. Then, the application of cost-sensitive techniques is meaningful for unbalanced databases.
- Improving the recall implies diminishing the precision of the model. The best-overall metrics were obtained to optimise the F1 metric, one metric that leverages recall and precision.
- Best values were obtained for the reference model as was expected. These values represent the theoretical values that could be expected if the database would be completely balanced. These are theoretical values that represent the threshold to compare the performances of the other experiments. The references metrics vary from 90 to 95% for the different metrics and should be used as reference values.
- The pure and Hybrid model provide worse values than the reference model. The Hybrid model offers better metrics than the Pure model. The Pure model's current metrics do not recommend its implementation because the missed-alert rate is 62%, and the false-alert rate is 56%. Both the Hybrid model rates are 70% and 75%, which means an improvement of about 20%.

The last step is the model finalisation, i.e., to train the model considering the training and the validation set and evaluate the metrics with the test set. It has been selected previous better models for Pure and Hybrid models. Table 11 shows the results obtained:

Experiment	Accuracy	AUC	Recall	Precision	F1
Pure model	0.961	0.814	0.651	0.587	0.617
Hybrid model	0.823	0.798	0.715	0.760	0.736

Table 11 Results of the finalised models for the Static mode.

These results provide the behaviour of the finalised model that can be expected after their deployment. The introduction of the validation set together with the training set means an improvement on the metrics over 2%. However, it is not presented in the above tables. For each one of the predictions, it is obtained the probability of the SI prediction. Therefore, it is concluded that the Hybrid model should be the one selected for the Static mode with rates close to 75% for missed and false alerts and 82% of overall accuracy.

6.1.2 ML results for regression

ML regression techniques perform numerical predictions about the safety metrics: MinDis, DistoMinDis and TimetoMinDis. The strong imbalance of the database mainly affects the classification problem, and there are no specific techniques to handle them in regression problems. Herein, the stratification problems for regression problem had considered the equal distribution of SI samples into the training, validation and testing sets.

The first step was to analyse different ML algorithms' performance, aiming to identify the best one. Two primary metrics are MAE and RMSE, although RMSE was identified as the optimisation metric. RMSE is the most typical metric used in regression problems because it represents better the impact of large-distanced predictions. The three same experiments for classification techniques were performed. The three experiments agreed that ensemble models provided the best results. For each experiment, the three top ML models varied between Extra trees, Random Forest, Catboost and Extreme Gradient. Extra Trees provided the best results for the three models. The models were evaluated based on stratified CV techniques to avoid overfitting on the results. The values obtained with the training set were compared with the validation set, which provided similar results.

The second step was to perform the feature selection. The three experiments provided similar results because the initial separation and initial azimuth were the most influential variables. GS variation and track variation encompassed together almost 50% of the influence. However, the impact of the variable changes for the different models. The pure model provided 25% of the initial separation effect, and the Hybrid model reduced their importance up to 15%.

On the other hand, the least influential features were the wake-turbulence type and vertical rate variables. Upon further removing the least influential variables, results confirmed that models with minor features provided worse values. The performance metrics diminished their value by about 1%. This implied that removing the least influential features on the model was feasible and could reduce computational cost.

The next step was the optimisation process of the ML algorithm. The goal of this process was to optimise the regression metric RMSE. Table 12 shows the results of the finalised models.

Safety metric	Experiment	RMSE	MAE	R2	RMSLE
MinDis	Reference model	7.602	4.881	0.917	0.483
	Pure model	6.542	3.808	0.936	0.293
	Hybrid model	**3.340**	2.337	0.6	0.392
DistoMinDis	Reference model	14.613	8.977	0.726	0.720
	Pure model	**12.184**	7.101	0.837	0.601
	Hybrid model	13.813	7.010	0.808	0.607
TimetoMinDis	Reference model	120.754	64.382	0.748	0.895
	Pure model	**99.272**	49.392	0.835	0.862
	Hybrid model	121.299	60.778	0.778	0.895

Table 12 Metrics for the finalised ML regression models for the Static mode.

These results provided the behaviour of the finalised model that can be expected after their deployment. The main conclusions about these results are:

- The optimisation process improved the initial metrics, although these improvements were reduced up to 0.5%. This implies that identifying the correct model is crucial because the subsequent optimisation process does not improve its metrics.
- To improve the RMSE implied to diminish the MAE of the model barely.
- The reference model provided the worst results. One reason was regression algorithms do not need balanced datasets to obtain the best behaviour.
- The hybrid model provided the best results for MinDis (3.6 NM) and Pure model for DistoMinDis (12 NM) and TimetoMinDis (99 sec). The improvements of the variables with the corresponding model were for MinDis (50% compared with Pure Model), DistoMinDis (15% compared with Pure Model) and TimetoMinDis (18% compared with Pure Model).

Therefore, it is clear which model should be selected for each safety metric, but it isn't easy to know how to integrate different experiments in the same system.

6.1.3 Analysis of results for the Static mode

The ML classification algorithms were applied to predict whether aircraft pairs constitute SI or not (and the probability of this prediction), and ML regression algorithms to predict the safety metrics MinDis, DistoMinDis and TimetoMinDis. The best algorithms for both problems were the ensemble methods. Notably, Random Forest and Extra Trees provided the best results in the majority of experiments.

Feature importance identified that wake-turbulence type and vertical rate variables should be discarded because their influence was almost null. The decrease was very low by removing them to the ML model, although computational time's impact went unnoticed. The hyperparameter optimisation process improved the ML models' results, although the improvements were reduced by 2%. Particularly for classification problems, it was most important to apply cost-sensitive techniques to handle the strong imbalance than the hyperparameter optimisation. The hyperparameter optimisation's computational time was very high, from one to three days in the worst experiments.

Three experiments were carried out. The reference model provided the hypothetical best results for classification algorithms, although it did not behave in such a way for regression algorithms. The hybrid model provided the best classification metrics (around 75% recall and F1) and MinDis (3.3 NM RMSE) for the regression algorithm. The pure model provided the best values for DistoMinDis (12.2 NM RMSE) and TimetoMinDis (100 seconds RMSE); the Hybrid model provides larger values (around 1.5 NM and 20 seconds more). Therefore, it can be concluded that the best model to implement was the Hybrid model. However, it demanded operational filtering for the aircraft pairs expected to cross with a separation lower than 20 NM and 1000 ft.

	flight_id_1	flight_id_2	SI_ML	Prob_ML	MinDis_ML	DistoMinDis_ML	TimetoMinDis_ML
0	AFR1321_494	AFR16FK_739	1	0.5152	6.87	45.03	394.0
1	AFR1321_494	EWG2LT_1471	1	0.6771	9.87	32.67	278.0
2	AFR1321_494	EJU3737_3809	0	0.8975	15.67	25.18	236.0
3	AFR1321_494	A6CAS_7863	1	0.6138	10.72	38.43	314.0
4	AFR1321_494	EWG9HA_1846	NaN	NaN	NaN	NaN	NaN
5	AFR1321_494	EJU67NL_3852	NaN	NaN	NaN	NaN	NaN

Figure 27 Example of predictions of the Hybrid model for the Static Mode

Figure 27 shows an example of the ML predictions. It provides the aircraft pair considered and the safety metrics. It has been added a value NaN to the aircraft pairs did not consider in the Hybrid model, i.e., the aircraft pair is not expected to cross with separation lower than 20 NM and 1000 ft.

6.2 ML predictor for the Dynamic Mode

This section provides the Dynamic mode results, and the process is similar to the Static mode. This mode is characterised because it gives SI prediction throughout the aircraft's evolution within the airspace and needs 4DT predictions as input. As explained in section 5.4, three experiments were carried out. The Static database's main difference is the number of samples that constitutes the database; it is ten times greater. Besides, the SI percentage of the database increases by 18%. It is noteworthy that the Hybrid model has more SI samples than no SI. Table 13 shows the number of samples for each experiment; in parenthesis, the rate of SI/no SI samples appear.

	Training set	Validation set	Test set
Reference model (50/50)	1556916	667250	247130
Pure model (18/82)	4254567	1823387	675329
Hybrid model (68/32)	1140470	488773	181027

Table 13 Number of samples of each experiment for the Dynamic mode.

Conversely, the Dynamic database's main problem was the calculation time required for training the ML algorithm. The training process of regression algorithms needed to halve the database. Appendix C details the process to obtain the results of the Pure model for regression algorithms.

6.2.1 ML results for classification

ML classification techniques perform prediction whether an aircraft pair is SI or not, and the probability of this prediction. The Dynamic database presents a high imbalance but no so sharped as the Static mode.

The first step was to analyse different ML algorithms' performance, aiming to identify the best one. The results were similar to the Static mode because the three experiments agreed that ensemble models provided the best results. For each experiment, the three top ML models varied between Random Forest, Extra trees, Decision Tree and Catboost. Nonetheless, the random forest provided the best results for the three experiments. The models were evaluated based on stratified CV techniques to avoid overfitting on the results. The values obtained with the training set were compared with the validation set, which provided similar results.

The feature selection results were completely different from the Static mode. The three experiments concluded the most influential features were the target prediction (MinDis_re and SI_re) which encompassed around 70% for each model influence. Other safety metrics predictions barely presented an effect on the model. The rest of the features provided a similar influence impact of about 2%. Based on the Static results, wake-turbulence type was not considered a feature in the Dynamic database. Therefore, no features were removed from the database.

The next step was the optimisation process of the ML algorithm. Based on the Static results, it only was an optimised F1 metric. The optimisation process was performed applying cost-sensitive techniques to the Pure and Hybrid model. Table 14 shows the different optimised models' results on the training set by using a 5-fold CV.

Experiment	Model	Optimizer	Accuracy	AUC	Recall	Precision	F1
Reference model	Non-optimised Model	-	0.984	0.999	0.991	0.986	0.988
	Optimised Model	**F1**	**0.985**	**0.999**	**0.993**	**0.984**	**0.989**
Pure model	Non-optimised Model	-	0.987	0.999	0.954	0.971	0.962
	Optimised Model	F1	0.986	0.999	0.955	0.968	0.963
	Optimised Cost-sensitive Model	**F1**	**0.986**	**0.999**	**0.956**	**0.968**	**0.963**
Hybrid model	Base Model	-	0.984	0.998	0.984	0.989	0.985
	Optimised Model	F1	0.981	0.998	0.985	0.987	0.986
	Optimised Cost-sensitive Model	**F1**	**0.980**	**0.998**	**0.989**	**0.982**	**0.986**

Table 14 Metrics for the different optimised ML classification models for the Dynamic mode.

The main conclusions about these results are:

- The results of the Dynamic mode were better than the Reference model of the Static mode. It implies that the introduction of the 4DT prediction was crucial to improving the SI predictions.

- The optimisation process barely improved the initial metrics because their values were very high; these improvements were reduced to 1%. This implies that identifying the correct model was crucial because the subsequent optimisation process did not improve the metrics.
- Cost-sensitive techniques provided minimal improvements compared with the Static mode. The best hyperparameters identified were similar to balanced ones.
- Best values were obtained for the reference model as was expected. These values represented the theoretical values that could be expected if the database would be completely balanced. These were theoretical values and represented the threshold to compare the performances of the other experiments. The references metrics were over 99% for the different metrics and should be used as reference values.
- The pure and Hybrid model provide slightly worse values than the reference model. The Hybrid model provided better metrics than the Pure model, but the difference is lower than 2%. Both models could be implemented because their values were quite similar, above 95% Pure model and 98% Hybrid model.

The last step was the model finalisation, i.e., to train the model considering the training and the validation set and evaluate the metrics with the testing set. It was selected previous better models for Pure and Hybrid models. Table 15 shows the results obtained:

Experiment	Accuracy	AUC	Recall	Precision	F1
Predictions	0.966	0.941	0.902	0.909	0.906
Pure model	0.990	0.999	0.987	0.989	0.988
Hybrid model	0.991	0.999	0.990	0.991	0.990

Table 15 Results of the finalised models for the Dynamic mode.

The metrics of both models were too high; their rates were almost 99% for every metric. However, it was required to compare these results with the 4DT predictions, i.e., to reach the improvement obtained by introducing ML techniques. The results confirmed that the introduction of ML techniques improved the predictions provided to the database. The predictions provided to the database guessed 90% of the SI, while the Hybrid or Pure model with 4DT predictions improved their metrics until 99%. This conclusion is crucial because ML techniques enhance the value of initial 4DT predictions; the ML model learns from the initial 4DT predictions and improves them. Although it is not presented in the above tables, the probability of the SI prediction for each of the predictions was obtained. Therefore, it was concluded that both the Hybrid and Pure model could be implemented because they provided results of around 99%.

6.2.2 ML results for regression

The first step was to analyse different ML algorithms' performance, aiming to identify the best one. RMSE was identified as the optimisation metric. The three same experiments for classification techniques were performed. The three experiments agreed that ensemble models provided the best results. For each experiment, the three top ML models varied between Extra trees, Random Forest, Catboost and Extreme Gradient. Extreme Gradient provided the best results for the three models. The

models were evaluated based on stratified CV techniques to avoid overfitting on the results. The values obtained with the training set were compared with the validation set, which provided similar results.

The second step was to perform the feature selection. It was completely different from the Static mode because the ML models received their importance from each target's prediction. The influence of each target prediction represented over 90% of the prediction. This implies that the rest of the features barely affected the model. However, these minor influences meant improvements in the ML models. Upon further removing all variables except the target prediction, results confirmed that models worsened their metrics by over 3%. Therefore, the features were not removed.

The following steps were the optimisation of the regression models and the finalisation process. Table 16 shows the results obtained only for the Pure and Hybrid model for the sake of clarity. In addition, it has been introduced the metrics of the 4DT predictions used in the database.

Safety metric	Experiment	RMSE	MAE	R2	RMSLE	MAPE
MinDis	Predictions	3.845	2.277	0.986	0.063	0.645
	Pure model with predictions	2.579	1.601	0.994	0.178	0.238
	Hybrid model with predictions	1.473	0.992	0.934	0.235	0.563
DistoMinDis	Predictions	5.771	2.684	0.954	0.109	0.413
	Pure model with predictions	3.696	1.944	0.973	0.244	0.280
	Hybrid model with predictions	5.331	2.401	0.956	0.290	0.348
TimetoMinDis	Predictions	38.320	21.553	0.955	0.279	0.307
	Pure model with predictions	32.631	16.984	0.967	0.367	0.293
	Hybrid model with predictions	37.469	7.376	0.965	0.371	0.294

Table 16 Metrics for the finalised ML regression models for the Dynamic mode.

The main conclusions about these results are:

- The optimisation process improved the initial metrics by up to 10% in some experiments. This implies that the Extreme Gradient model behaviour greatly depended on the optimisation process.
- The dynamic mode provided better results than Static mode. The reductions in all safety metrics were close to 60%.
- The hybrid model provided the best results for MinDis (1.5 NM) and Pure model for DistoMinDis (3.7 NM) and TimetoMinDis (32.6 sec). The improvements of the variables with the corresponding model were for MinDis (40% compared with Pure Model), DistoMinDis (30% compared with Pure Model) and TimetoMinDis (21% compared with Pure Model). The reference model provided the worst results for MinDis, but the Hybrid model provided the worst outcomes for DistoMinDis and TimetoMinDis.
- The introduction of the validation set together with the training set meant an improvement on the metrics over 20%. Then, the database volume was a crucial aspect to consider in the development of this mode.

Similar to classification models, it was required to compare these values with the ML model predictions. Pure and Hybrid model improved RMSE values of 4DT predictions, which means ML models can improve the initial 4DT predictions. However, these improvements were different for the Pure and Hybrid model. The hybrid model improved 60% RMSE values for MinDis while the Pure model 32%. The hybrid model barely improved DistoMinDis and TimetoMinDis by about 8% and 3%. Conversely, the Pure model improved its values by 35% and 15%. The more considerable benefits could be obtained with the Hybrid model for MinDis.

The above conclusions were similar to the regression models of Static mode. Therefore, it was clear which model should be selected for each one of the safety metrics. However, it isn't easy to know how to integrate different ML models in the same system.

6.2.3 Analysis of results for the Dynamic mode

The best algorithms for classification and regression problems were ensemble methods. Notably, Random Forest for classification and Extreme Gradient for regression provided the best results in the experiments. The Dynamic mode's main problem was the large dataset used to train the models that demanded substantial computational time and computer resources.

Feature importance was different from the Static mode because the predicted variables of 4DT were the most influential (above 90%). The rest of the variables provided a meagre influence; however, they had not been discarded based on operational knowledge. The hyperparameter optimisation process improved the results of the ML models. In Classification, the improvements were reduced, but the Extreme Gradient model improved the values up to 20%. Cost-sensitive techniques were applied, although the benefits were reduced compared with the Static mode.

The Dynamic mode remarkably increased the performances of both classification and regression models. Pure and Hybrid models provided almost 99% of reliability in classification models. They improved their values from 90% of the 4DT predictions supplied to the database. However, the improvements achieved by regression models varied depending on the model and the target. The hybrid model provided the best values for MinDis. It improved the RMSE value of the predictions provided to the database from 3.8 to 1.5 NM. However, the benefits of the Hybrid model were reduced for DistoMinDis and TimetoMinDis. The best values of both safety metrics were obtained for the Pure Model. They reduced the RMSE value of the predictions provided to the database from 5.8 to 3.7 NM and 38 to 32 seconds.

Therefore, the Dynamic model's classification results seemed excellent, although the goal must be to reach a 0% error rate. Regression results improved the values for the Static mode. However, they required more research to enhance their errors, particularly for DistoMinDis and TimetoMinDis.

	flight_id_1	flight_id_2	timestamp	SI_ML	Prob_ML	MinDis_ML	DistoMinDis_ML	TimetoMinDis_ML
0	A6CAS_000	TOM4BA_044	2019-06-20 12:20:10+00:00	1	1	1.21	28.8	220.0
1	AFR1321_002	AFR16FK_003	2019-06-20 12:20:10+00:00	1	1	5.52	7.9	70.0
2	AFR1321_002	NJE614U_025	2019-06-20 12:20:10+00:00	0	0.7254	19.56	2.6	13.0
3	AFR1321_002	SVA126_038	2019-06-20 12:20:10+00:00	0	0.9699	15.99	30.1	242.0
4	AFR16FK_003	EWG2LT_014	2019-06-20 12:20:10+00:00	0	0.9374	16.86	4.1	32.0
5	AFR16FK_003	TOM4BA_044	2019-06-20 12:20:10+00:00	0	0.5694	13.06	34.8	261.0
6	SVA126_038	TOM4BA_044	2019-06-20 12:20:10+00:00	1	1	5.31	61.7	456.0
7	A6CAS_000	SVA126_038	2019-06-20 12:20:10+00:00	NaN	NaN	NaN	NaN	NaN
8	AFR1321_002	IBK1CH_019	2019-06-20 12:20:10+00:00	NaN	NaN	NaN	NaN	NaN

Figure 28 Example of predictions of the Hybrid model for the Dynamic Mode

Figure 27 shows an example of the ML predictions. It provides the aircraft pair considered and the safety metrics. It has been added a value NaN to the aircraft pairs did not consider in the Hybrid model, i.e., the aircraft pair is not expected to cross with separation lower than 20 NM and 1000 ft.

7 Conclusions

This deliverable describes task 3.2 about the application of ML techniques for conflict detection. This work deals with monitoring tasks focusing on situational awareness. With this aim, this work uses the concept of Situation of Interest (SI). One SI is when an aircraft pair is expected to intersect with a horizontal separation lower than a pre-defined separation and infringe the vertical separation minima. Herein, one SI is an aircraft pair expected to cross with a separation lower than 10 NM and 1000 ft. Nonetheless, this value can be specified by ANSP in future work.

To tackle this problem, it has been developed two approaches with similar roles of the ATCO team. The Static mode focuses on planner ATCO. This mode predicts SI and their safety metrics when an aircraft pierces into the airspace with the aircraft located within the airspace. This prediction is performed once, and it is not updated. It does not receive the 4DT prediction of the different aircraft as input. The Dynamic mode focuses on tactical ATCO. This mode predicts SI and their safety metrics throughout the evolution of the aircraft within the airspace. The predictions can be performed dynamically and not once as the Static mode. Another difference is the Dynamic mode receives the 4DT prediction of the aircraft within the airspace, and the aircraft will pierce the airspace in the same time period. The Dynamic mode considers the prediction of SI, and safety metrics evolve with the trajectory evolution.

The work has been developed considering ADS-B data only. ADS-B trajectories have been extracted from The OpenSky Network and constituted the basis for the ML database. This database of ADS-B trajectories has also been used as 4DT predictions for the Dynamic mode. The Dynamic mode selected the most similar trajectory stored in the database, using it as 4DT prediction in the ML model's input data. LSAZM567 airspace in Switzerland is the case study of this work. However, this methodology is generalizable and could be applied to different airspaces.

Current ADS-B trajectories do not provide SI or conflicts. Hence, it has been necessary to build a database with SI samples. The main problem to build a database with enough SI was to modify the trajectories flown to force situations in which an SI would have emerged. The trajectories have been temporarily modified to pierce the airspace within the same time period. The way to generate aircraft pairs and new entry times is different for the Static and Dynamic modes. Lastly, the evolution of each aircraft pair's trajectories is evaluated to calculate the safety metrics; in turn, the safety metrics are labelled for each database sample.

Both modes apply classification and regression algorithms. Classification algorithms perform predictions about whether an aircraft pair is an SI or not, which is the probability of this prediction. Regression algorithms perform numerical predictions for the safety metrics: minimum distance to reach between an aircraft pair, the distance and the time to reach this point. Then, there are five outputs calculated from independent ML predictors. Three experiments have been carried out to tackle the strong imbalance of the database regarding SI samples (5% in the Static and 18% in the Dynamic mode). The Reference model provides the theoretical best result (balancing SI and no SI samples) in classification problems. The Pure model considers the whole dataset. The Hybrid model performs an operational filter to focus on aircraft pairs that approach a distance lower than 20 NM and 1000 ft.

The Static mode presents lower accuracy because it can predict the safety metrics when an aircraft pierces into the airspace with other aircraft within the airspace (without 4DT predictions). The

hypothetical operational limit is around 95%, with which should be compared the different ML models. The best classification metrics were obtained for the Hybrid model (75%) and the MinDis safety metric (RMSE of 3.6 NM). DistoMinDis and TimetoMinDis metrics achieved their best values with the Pure model. Their values were 12 NM and 92 seconds, which did not represent high rates for conflict prediction issues.

The Dynamic mode remarkably increased the performances of both classification and regression models of the Static mode. The classification algorithms provided almost 99% of reliability, while the 4DT predictions provided as inputs were 90%. Although the Hybrid model was better, the difference with the Pure model could be negligible. The Hybrid model also offered the best values for MinDis safety-metric; it improved the RMSE value of 4DT predictions from 3.8 to 1.5 NM. However, the other safety metrics barely improved their values with the Hybrid model. The pure model provided the best values for DistoMinDis from 5.8 to 3.7 NM and for TimetoMinDis from 38 to 32 seconds.

Therefore, this work confirmed that ML techniques could be applied for conflict detection purposes. They can be used in isolation of 4DT predictions or with them. ML techniques improved the prediction ability from the 4DT predictions considered as data input. This conclusion is significant because it means that ML techniques improve initial 4DT predictions, i.e., the ML model learns from the initial prediction and improves them.

It has been identified several problems throughout this work. The most significant was the substantial computational cost regarding time and workload. The performance of simulations demanded more than 1450 hours. Similarly, the ML training and optimisation process required from one to three days for each experiment. It was clear that the process should be improved by reducing computational time in further work. Conversely, the increase of the database can improve the metrics of the ML predictor. The cost is to increase computational training time. The first results confirmed that the metrics score increase was not proportional to the number of samples.

Moreover, several research lines should be considered in further work. Once it has been proved the validity for conflict detection, classification techniques should be applied to conflicts. The improvements of the Dynamic mode are evaluated against ADS-B trajectories customized to be used as 4DT predictions. Then, it should be considered the introduction of real 4DT predictions by implementing ML techniques in a real system. This work evaluated different ML models and should step forward by considering Deep Learning or neural networks techniques.

8 Bibliography

[1] ICAO, “Air Traffic Management - Doc 4444,” 2012.

[2] ICAO, “Doc 9689-AN/953 - Manual on airspace planning methodology for the determination of separation minima,” pp. 1–205, 1998, doi: 10.1007/s13398-014-0173-7.2.

[3] P. G. Reich, “Analysis of long-Range air traffic systems — Separation Standards I,” *J. Inst. Navig.*, vol. 50, no. 3, pp. 436–477, 1966.

[4] P. G. Reich, “Analysis of long-range air Separation Standards — II,” *J. Navig.*, no. 19(2), pp. 169–186, 1966, doi: 10.1017/S0373463300047196.

[5] ICAO, “Doc 9689-AN/953 - Manual on airspace planning methodology for the determination of separation minima,” 1998. doi: 10.1007/s13398-014-0173-7.2.

[6] ICAO, “Doc 9574 - Manual on Implementation of a 300 m (1 000 ft) Vertical Separation Minimum Between FL 290 and FL 410 Inclusive,” 2001.

[7] F. Netjasov and M. Janic, “A review of research on risk and safety modelling in civil aviation,” *J. Air Transp. Manag.*, vol. 14, no. 4, pp. 213–220, 2008, doi: 10.1016/j.jairtraman.2008.04.008.

[8] F. Netjasov, “Framework for airspace planning and design based on conflict risk assessment Part 1 : Conflict risk assessment model for airspace strategic planning,” *Transp. Res. Part C Emerg. Technol.*, vol. 24, pp. 190–212, 2012, doi: 10.1016/j.trc.2012.03.002.

[9] F. Netjasov, “Framework for airspace planning and design based on conflict risk assessment. Part 2: Conflict risk assessment model for airspace tactical planning,” *Transp. Res. Part C Emerg. Technol.*, vol. 24, pp. 213–226, 2012, doi: 10.1016/j.trc.2012.03.003.

[10] F. Netjasov and O. Babic, “Framework for airspace planning and design based on conflict risk assessment. Part 3: Conflict risk assessment model for airspace operational and current day planning,” *Transp. Res. Part C Emerg. Technol.*, vol. 32, pp. 31–47, 2013, doi: 10.1016/j.trc.2013.04.002.

[11] S. H. Kim, “Conflict Risk Assessment of Structured and Unstructured Traffic of Small Unmanned Aircraft Systems,” in *2018 Aviation Technology, Integration, and Operations Conference*, 2018, vol. 15, no. 12, pp. 25–29, doi: 10.2514/6.2018-3033.

[12] J. A. Pérez-Castán, F. Gómez Comendador, Á. Rodríguez-Sanz, and R. M. Arnaldo Valdés, “Conflict-risk assessment model for continuous climb operations,” *Aerosp. Sci. Technol.*, vol. 84, pp. 812–820, 2019, doi: 10.1016/j.ast.2018.11.030.

[13] Siddiqee W., “A mathematical model for predicting the number of potential conflict situations at intersecting air routes,” *Transp. Sci.*, vol. 7, no. 2, pp. 571–577, 1973, doi: 10.1287/trsc.7.2.158.

[14] E. Sunil, J. Ellerbroek, J. M. Hoekstra, and J. Maas, “Three-dimensional conflict count models for unstructured and layered airspace designs,” *Transp. Res. Part C Emerg. Technol.*, vol. 95, no. June, pp. 295–319, 2018, doi: 10.1016/j.trc.2018.05.031.

[15] M. Mitici and H. A. P. Blom, “Mathematical Models for Air Traffic Conflict and Collision Probability Estimation,” *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 3, pp. 1052–1068, 2019, doi: 10.1109/TITS.2018.2839344.

[16] C. Gong and D. McNally, “A methodology for automated trajectory prediction analysis,” in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2004, no. August, pp. 1–14, doi: 10.2514/6.2004-4788.

[17] C. A. Persiani and S. Bagassi, “Route planner for unmanned aerial system insertion in civil non-segregated airspace,” *Proc. Inst. Mech. Eng. Part G J. Aerosp. Eng.*, vol. 227, no. 4, pp. 687–702, 2013, doi: 10.1177/0954410012439975.

[18] J. K. Kuchar and L. C. Yang, “Conflict detection and resolution, air traffic control, alerting systems, warning systems. I. I,” in *IEEE Transactions on Intelligent Transportation Systems*, 2000, vol. 1, no. 4, pp. 179–189.

[19] J. A. Pérez-Castán, F. G. Comendador, Á. Rodríguez-Sanz, R. Barragán, and R. M. Arnaldo-Valdés, “Design of a conflict-detection air traffic control tool for the implementation of continuous climb operations: A case study at Palma TMA,” *Proc. Inst. Mech. Eng. Part G J. Aerosp. Eng.*, vol. 0, no. 0, p. 095441001983119, 2019, doi: 10.1177/0954410019831198.

[20] J. A. Pérez-Castán, F. G. Comendador, Á. Rodriguez-Sanz, R. M. Arnaldo Valdés, and G. Agueda, “RPAS integration in non-segregated airspace: Safety metrics for tactical planning,” *Proc. Inst. Mech. Eng. Part G J. Aerosp. Eng.*, vol. 0, no. 0, p. 095441001986126, 2019, doi: 10.1177/0954410019861263.

[21] J. A. Pérez-Castán *et al.*, “Probabilistic Strategic Conflict - Management for 4D Trajectories in Free - Route Airspace,” *Entropy*, vol. 22, pp. 1–17, 2020, doi: 10.3390/e22020159.

[22] Y. Matsuno and T. Tsuchiya, “Stochastic 4D trajectory optimization for aircraft conflict resolution,” 2014, doi: 10.1109/AERO.2014.6836275.

[23] J. Krozel, M. E. Peters, and G. Hunter, “Conflict detection and resolution for future air transportation management - TR97138-01,” 1997.

[24] H. Erzberger, T. Nikoleris, R. A. Paielli, and Y. C. Chu, “Algorithms for control of arrival and departure traffic in terminal airspace,” *Proc. Inst. Mech. Eng. Part G J. Aerosp. Eng.*, vol. 230, no. 9, pp. 1762–1779, 2016, doi: 10.1177/0954410016629499.

[25] H. Erzberger and K. Heere, “Algorithm and operational concept for resolving short-range conflicts,” *Proc. Inst. Mech. Eng. Part G J. Aerosp. Eng.*, vol. 224, no. 2, pp. 225–243, 2010, doi: 10.1243/09544100JAERO546.

[26] K. Margellos and J. Lygeros, “Toward 4-D Trajectory Management in Air Traffic Control: A Study based on Monte Carlo Simulation and Reachability Analysis,” *IEEE Trans. Control Syst. Technol.*, vol. 21, no. 5, pp. 1820–1833, 2013, doi: 10.1109/TCST.2012.2220773.

[27] S. Alam, K. Shafi, H. A. Abbass, and M. Barlow, “An ensemble approach for conflict detection in Free Flight by data mining,” *Transp. Res. Part C Emerg. Technol.*, vol. 17, no. 3, pp. 298–317, 2009, doi: 10.1016/j.trc.2008.12.002.

[28] Z. Wang, M. Liang, and D. Delahaye, “Data-driven conflict detection enhancement in 3d airspace with machine learning,” *2020 Int. Conf. Artif. Intell. Data Anal. Air Transp. AIDA-AT 2020*, no. February, 2020, doi: 10.1109/AIDA-AT48540.2020.9049180.

[29] EASA, “Artificial Intelligence Roadmap A human-centric approach to AI in aviation,” 2020. [Online]. Available: https://www.easa.europa.eu/sites/default/files/dfu/EASA-AI-Roadmap-v1.0.pdf.

[30] T. M. Mitchell, *Machine Learning*. McGraw-Hill Science/Engineering/Math, 1997.

[31] A. Geron, *Hands-On Machine Learning with Scikit-Learn and TensorFlow*. O’Reilly Media, Inc., 2017.

[32] B. Sridhar, “Applications of machine learning techniques to aviation operations: Promises and challenges,” *2020 Int. Conf. Artif. Intell. Data Anal. Air Transp. AIDA-AT 2020*, pp. 1–12, 2020, doi: 10.1109/AIDA-AT48540.2020.9049205.

[33] J. Hegde and B. Rokseth, “Applications of machine learning methods for engineering risk assessment – A review,” *Saf. Sci.*, vol. 122, no. September 2019, p. 104492, 2020, doi: 10.1016/j.ssci.2019.09.015.

[34] Y. Lv, Y. Duan, W. Kang, Z. Li, and F. Y. Wang, “Traffic Flow Prediction with Big Data: A Deep Learning Approach,” *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 865–873, 2015, doi: 10.1109/TITS.2014.2345663.

[35] Y. J. Kim, S. Choi, S. Briceno, and D. Mavris, “A deep learning approach to flight delay prediction,” *AIAA/IEEE Digit. Avion. Syst. Conf. - Proc.*, vol. 2016-Decem, pp. 1–6, 2016, doi: 10.1109/DASC.2016.7778092.

[36] E. C. Fernández *et al.*, “DART: A machine-learning approach to trajectory prediction and demand-capacity balancing,” *SESAR Innov. Days*, no. iii, 2017, [Online]. Available: https://www.sesarju.eu/sites/default/files/documents/sid/2017/SIDs_2017_paper_65.pdf.

[37] Y. Le Fablec and J. M. Alliot, “Using Neural Networks to predict aircraft trajectories,” *Int. Conf. Artif. Intell.*, pp. 524–529, 1999, [Online]. Available: http://alliot.info/papers/icai99.pdf%0Ahttps://www.researchgate.net/publication/2487739_Using_Neural_Networks_to_predict_aircraft_trajectories.

[38] A. M. P. de Leege, M. M. van Paassen, and M. Mulder, “A machine learning approach to trajectory prediction,” *AIAA Guid. Navig. Control Conf.*, pp. 1–14, 2013, doi: 10.2514/6.2013-4782.

[39] R. Alligier, D. Gianazza, and N. Durand, “Machine Learning and Mass Estimation Methods for Ground-Based Aircraft Climb Prediction,” *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 6, pp. 3138–3149, 2015, doi: 10.1109/TITS.2015.2437452.

[40] R. Alligier *et al.*, “Machine Learning Applied to Airspeed Prediction During Climb,” in *ATM seminar 2015, 11th USA/EUROPE Air Traffic Management R&D Seminar, FAA & Eurocontrol*, 2015, pp. 1–10, [Online]. Available: https://hal-enac.archives-ouvertes.fr/hal-01168664.

[41] R. Alligier and D. Gianazza, “Learning aircraft operational factors to improve aircraft climb

prediction: A large scale multi-airport study," *Transp. Res. Part C Emerg. Technol.*, vol. 96, no. July, pp. 72–95, 2018, doi: 10.1016/j.trc.2018.08.012.

[42] Z. Wang, M. Liang, and D. Delahaye, "A hybrid machine learning model for short-term estimated time of arrival prediction in terminal manoeuvring area," *Transp. Res. Part C Emerg. Technol.*, vol. 95, no. July, pp. 280–294, 2018, doi: 10.1016/j.trc.2018.07.019.

[43] C. S. Bosson and T. Nikoleris, "Supervised learning applied to air traffic trajectory classification," *AIAA Inf. Syst. Infotech Aerospace, 2018*, no. 209989, 2018, doi: 10.2514/6.2018-1637.

[44] S. Alam, K. Shafi, H. A. Abbass, and M. Barlow, "An ensemble approach for conflict detection in Free Flight by data mining," *Transp. Res. Part C Emerg. Technol.*, vol. 17, no. 3, pp. 298–317, 2009, doi: 10.1016/j.trc.2008.12.002.

[45] D. T. Pham, N. P. Tran, S. Alam, V. Duong, and D. Delahaye, "A machine learning approach for conflict resolution in dense traffic scenarios with uncertainties," 2019, [Online]. Available: https://hal-enac.archives-ouvertes.fr/hal-02138135/document.

[46] Z. Wang, M. Liang, D. Delahaye, and W. Wu, "Learning real trajectory data to enhance conflict detection accuracy in closest point of approach problem," *13th USA/Europe Air Traffic Manag. Res. Dev. Semin. 2019*, 2019, [Online]. Available: https://www.semanticscholar.org/paper/Learning-Real-Trajectory-Data-to-Enhance-Conflict-Wang-Liang/797957c9992f5176bc23f870f2156b2985ab0408.

[47] Boeing Commercial Airplanes, "Statistical Summary of Commercial Jet Airplane Accidents World Wide Operations 1959-2018," 2019. [Online]. Available: http://www.boeing.com/commercial/safety/investigate.html.

[48] S. Chaimatanan, D. Delahaye, and M. Mongeau, "Aircraft 4D trajectories planning under uncertainties," *Proc. - 2015 IEEE Symp. Ser. Comput. Intell. SSCI 2015*, pp. 51–58, 2015, doi: 10.1109/SSCI.2015.18.

[49] EUROCONTROL, "emand data repository (DDR) | EUROCONTROL." https://www.eurocontrol.int/ddr.

[50] SESAR, "Digitalising Europe's aviation infrastructure A discussion paper," 2017. [Online]. Available: https://www.sesarju.eu/sites/default/files/documents/reports/Digitalising_Europe_Aviation_Infrastructure.pdf.

[51] SESAR Joint Undertaking, "EUROPEAN ATM MASTER PLAN - Digitalising Europe's aviation Infraestructure," 2020. doi: 10.2829/10044.

[52] EUROCONTROL, "NEST modelling tool." 2017.

[53] T. O. Network, "The OpenSky Network API," 2017. .

[54] FlightAware, "FlightAware," 2020. https://es.flightaware.com/ (accessed Nov. 20, 2020).

[55] ICAO, "Part 2 — Aircraft Type Designators (Decode)." [Online]. Available: https://cfapp.icao.int/Doc8643/reports/Part2-By Type Designator(Decode).pdf.

[56] C. F. F. Karney, “geographiclib,” 2015. https://geographiclib.sourceforge.io/html/python/ (accessed Sep. 10, 2020).

[57] M. F. Uddin, “Addressing Accuracy Paradox Using Enhanched Weighted Performance Metric in Machine Learning,” in *ITT 2019 - Information Technology Trends: Emerging Technologies Blockchain and IoT*, 2019, pp. 319–324, doi: 10.1109/ITT48889.2019.9075071.

[58] C. Goutte and E. Gaussier, “A Probabilistic Interpretation of Precision, Recall and F-Score, with Implication for Evaluation,” *Lect. Notes Comput. Sci.*, vol. 3408, pp. 345–359, 2005, doi: 10.1007/978-3-540-31865-1_25.

[59] G. Weiss, K. McCarthy, and B. Zabar, “Cost-sensitive learning vs. sampling: Which is best for handling unbalanced classes with unequal error costs?,” in *DMINd*, 2007, pp. 1–7, [Online]. Available: http://storm.cis.fordham.edu/~gweiss/papers/dmin07-weiss.pdf.

[60] G. Varoquaux, L. Buitinck, G. Louppe, O. Grisel, F. Pedregosa, and A. Mueller, “Scikit-learn,” *GetMobile Mob. Comput. Commun.*, vol. 19, no. 1, pp. 29–33, 2015, doi: 10.1145/2786984.2786995.

[61] ICAO, “Aeronautical Surveillance Manual,” 2010. [Online]. Available: http://www.aviationchief.com/uploads/9/2/0/9/92098238/icao_doc_9924_-_aeronautical_surveillance_manual_-_1st_edition_-_2011.pdf.

[62] P. Branco, L. Torgo, and R. Ribeiro, “A Survey of Predictive Modelling under Imbalanced Distributions,” *CoRR*, vol. abs/1505.0, pp. 1–48, 2015, [Online]. Available: http://arxiv.org/abs/1505.01658.

Appendix A ADS-B data

A.1 ADS-B inputs

ADS-B is a surveillance technology in which an aircraft calculates its position via satellite navigation based on the GNSS system [61]. It broadcasts the position and other operational information of the aircraft every updating period. The information on the aircraft's status is defined as state vectors and contains most tracking information. Table 17 shows the main variables provided by ADS-B data. In this way, the state vectors provide tracking information for each point of the trajectory.

	ADS-B variables	Type	Description
0	icao24	string	Unique ICAO 24-bit address of the transponder in hex string representation.
1	callsign	string	Callsign of the vehicle (8 chars). It can be null if no callsign has been received.
2	origin_country	string	Country name inferred from the ICAO 24-bit address.
3	time_position	int	Unix timestamp (seconds) for the last position update. It could be null if OpenSky received no position report within the past 15s.
4	last_contact	int	Unix timestamp (seconds) for the last update in general. This field is updated for any new, valid message received from the transponder.
5	longitude	float	WGS-84 longitude in decimal degrees.
6	latitude	float	WGS-84 latitude in decimal degrees.
7	baro_altitude	float	Barometric altitude in meters.
8	on_ground	boolean	A boolean value indicates if the position was retrieved from a surface position report.
9	velocity	float	Velocity over the ground in m/s.
10	true_track	float	True track in decimal degrees clockwise from north (north=0°).
11	vertical_rate	float	The vertical rate in m/s. A positive value indicates that the aeroplane is climbing; a negative value indicates that it descends.
12	sensors	int[]	IDs of the receivers which contributed to this state vector. It is null if no filtering for the sensor was used in the request.
13	geo_altitude	float	Geometric altitude in meters.
14	squawk	string	The transponder code aka Squawk.

15	spi	boolean	Whether flight status indicates a special purpose indicator.
16	position_source	int	Origin of this state's position: 0 = ADS-B, 1 = ASTERIX, 2 = MLAT

Table 17 Description of ADS-B variables [53].

Apart from the variables provided by the ADS-B data, it is necessary to provide new variables or features regarding the intrinsic conditions of each aircraft pair. The relative features provide information about different variables' characteristics compared between an aircraft pair in one specific timestamp. Table 18 shows the relative features considered in this work.

Relative features	type	Description
Horizontal separation	float	The horizontal separation between an aircraft pair
Heading	float	Heading between an aircraft pair
GS variation	float	GS variation between an aircraft pair
Track variation	float	Track variation between an aircraft pair
Altitude variation	float	The altitude difference between an aircraft pair
Vertical-rate variation	float	The vertical-rate variation between an aircraft pair

Table 18 Description of relative features.

Together, all of these variables provide additional information to ADS-B data that could be crucial to adjusting and improving ML techniques. These variables are proposed based on ADS-B data. If there were other types of trajectory data, it would be necessary to adapt the available information variables. Besides, if there is more information available, it could be easily added as a feature input (e.g., wind information, temperature, aircraft weight, etc.).

A.2 Output Labels

The results expected by applying ML techniques are the output labels. Output labels refer to the information about SI and safety metrics for each aircraft pair. Table 19 defines the output labels considered herein.

Labels	type	Description
SI	Boolean	A binary variable that could acquire the values 1 (SI) or 0 (No SI). This variable gets value 1 when a SI is identified for one aircraft pair. Otherwise, the samples are labelled as 0.
SI probability	float	A numerical variable that provides information about the prediction probability. This variable is calculated based on ML algorithms and represents a confidence value of the prediction.
Minimum distance	float	A numerical variable that provides information about the minimum separation calculated for each aircraft pair.
Distance to reach the minimum distance	float	A numerical variable that provides information about the distance to reach the minimum separation calculated for each aircraft pair.
Time to reach the minimum distance	float	A numerical variable that provides information about the time to reach the minimum separation calculated for each aircraft pair.

Table 19 Description of output labels.

A.3 Labels as features in the Dynamic mode

The Dynamic mode performs the SI prediction with information about the aircraft's current state vector plus 4DT prediction. This work assumes that some kind of 4DT prediction is available by the system in order to provide information about the safety metrics expected (minimum separation, distance and time to reach minimum separation). Therefore, it is required to get information about the aircraft's current situation (ADS-B data) and 4DT prediction in the airspace considered (ground or air source).

When a 4DT prediction of both aircraft trajectories is available, the predicted safety metrics can be calculated and provided that information to the ML. In this way, the ML prediction can improve the prediction based on historical data. Besides, the use of 4DT predictions as inputs for ML will confirm whether this information could improve the predictions or not. There are no available 4DT predictions about the trajectory that will fly the aircraft in the LSAZM567 airspace in this work. Therefore, this work will select one ADS-B trajectory available in the database to be employed as the 4DT, as explained in section 2.4.2.

Appendix B ML experiment for Static mode

This appendix shows the process performed to train the Pure model for the classification of the Static mode. The pure model uses the whole database and presents a strong imbalance between SI and no SI (5/95). Figure 29 shows the results of the stratification process in the three sets.

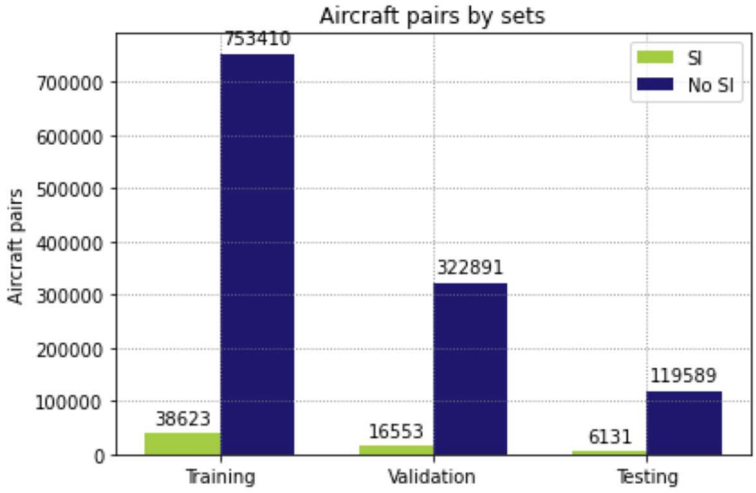

Figure 29 Distribution of SIs by sets of Pure model for the Static mode.

PyCaret allows performing a first comparison of the ML models. Table 20 shows the training set results by applying a 5-fold CV process, providing the average values. The main results for classification algorithms are:

- All algorithms provide very high accuracy (>95%), i.e., it guesses with a very high SI rate and no SI as a whole. However, this is not the correct metric to evaluate unbalanced datasets. ML models with an F1 value equal to 0 provides an accuracy of 95%, which means that all samples are classified as no SI.
- Precision acquires diverse values depending on the algorithms. Ensemble methods provide metrics above 80%.
- The recall relates the classification rate regarding only SI samples and provides the worst results. Recall acquires the best results for the Decision Tree with almost 60% when the rest of the models do not reach 50%.
- F1 is a combined metric of recall and precision. Best values are obtained for ensemble methods, and only four of them provide values above 50%.

Therefore, the best values are obtained for ensemble methods: Random forest, Extra trees and Decision tree. These results confirm that unbalanced problems get the best results with ensemble methods [62]. Although random forest provides the best value for precision and F1, there is a high difference in the decision tree's recall. The decision tree provides a more equilibrated value for the three metrics (precision, recall and F1) than the random forest. Then, the decision tree has been selected as the ML algorithm that can provide the best results.

Model	Accuracy	AUC	Recall	Precision	F1
Random Forest Classifier	0.970	0.978	0.458	0.867	0.600
Extra Trees Classifier	0.969	0.973	0.445	0.853	0.585
Decision Tree Classifier	0.957	0.780	0.577	0.561	0.569
CatBoost Classifier	0.967	0.973	0.429	0.793	0.557
K Neighbors Classifier	0.960	0.887	0.408	0.643	0.499
Extreme Gradient Boosting	0.964	0.968	0.358	0.800	0.495
Light Gradient Boosting Machine	0.961	0.958	0.258	0.830	0.393
Gradient Boosting Classifier	0.956	0.911	0.131	0.818	0.226
Ada Boost Classifier	0.950	0.849	0.065	0.412	0.112
Naive Bayes	0.924	0.709	0.124	0.215	0.102
Quadratic Discriminant Analysis	0.897	0.497	0.054	0.025	0.025
Logistic Regression	0.951	0.686	0.000	0.000	0.000
SVM - Linear Kernel	0.951	0.000	0.000	0.000	0.000
Ridge Classifier	0.951	0.000	0.000	0.000	0.000
Linear Discriminant Analysis	0.951	0.681	0.000	0.000	0.000

Table 20 Metrics of ML algorithms applied to Pure model for the Static mode.

Feature selection provides information about the influence of the features on the ML algorithm. Figure 30 shows the feature importance plot and the Recursive Feature Elimination (RFE) results with CV [60]. The variable with larger influence is the initial azimuth, which represents the course that connects both aircraft. Initial horizontal and vertical separation and the track are influential variables as well. These four features are the most important because they represent almost 50% of the feature importance. Conversely, we can identify the features that do not provide a relation with the model. The wake-turbulence type and the vertical rate variables constitute them. The application of RFE confirms these results. RFE evaluates which is the importance of the features to known which one of them do not provide an improvement in the model. The optimum number of features is 17 when initially there are 28. RFE confirms wake-turbulence type and vertical rate variables should not be considered.

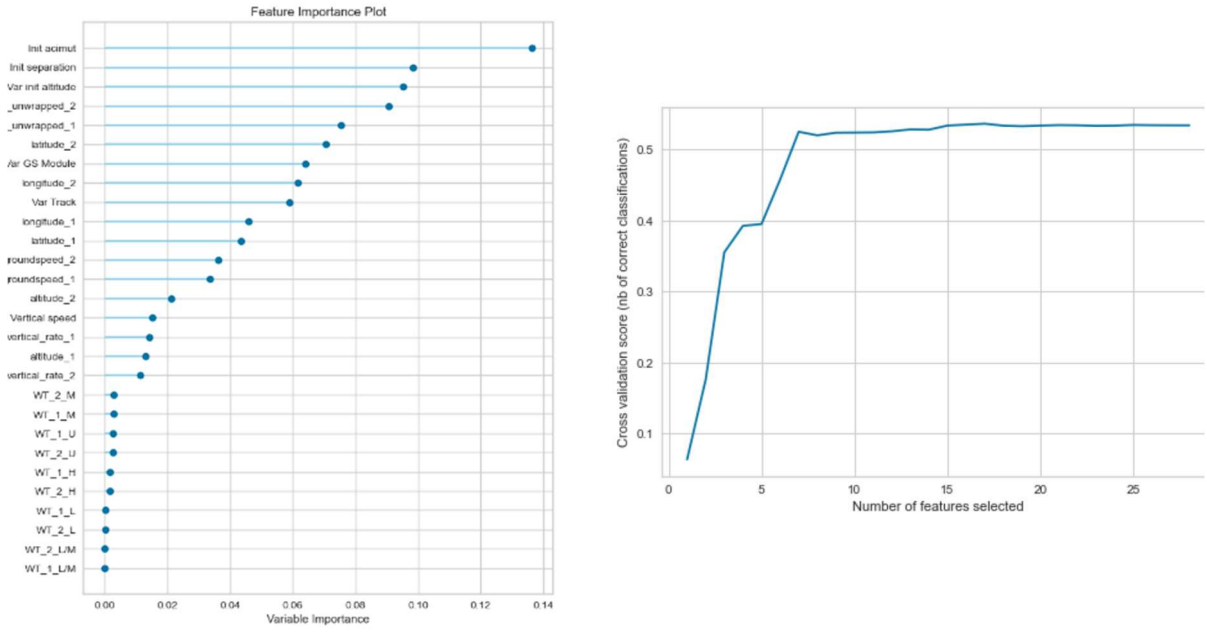

Figure 30 Left) feature importance and right) RFE function of Pure model for the Static mode.

Table 21 shows the ML model results without those features, which are pretty similar to all features.

Model	Accuracy	AUC	Recall	Precision	F1
Validation set with all features	0.957	0.780	0.577	0.561	0.569
Validation set with feature selection	0.958	0.783	0.570	0.585	0.577

Table 21 Comparison of the model performance with feature selection of Pure Model for the Static mode.

The next step is to optimise the hyperparameters of the algorithm to optimise the behaviour and avoid overfitting. Hyperparameters must be defined in advance of the training process of the model. Besides, it has been applied cost-sensitive techniques to tackle the strong imbalance of the dataset. Table 22 shows the grid search of the hyperparameters for the decision tree classifier.

Hyperparameter	Values
Class weight	[None, 'balanced', {1/5:15}]
Complexity parameter	[0:0.02]
Maximum depth	[None, 3, 5, 10, 20]
Maximum number features	[None, 5,7,10,15]

Minimum samples of leaf	[2, 10, 20, 50, 100]
Criterion	['gini', 'entropy']

Table 22 Hyperparameters selected for the Grid search of the Pure model for the Static mode.

It has been optimised the recall and F1; it has also been performed 5-fold CV. Table 23 shows the results of the different ML models evaluated during experiment 1.

Model	Optimizer	Accuracy	AUC	Recall	Precision	F1	Kappa
Non-optimised Model	-	0.958	0.780	0.570	0.585	0.577	0.547
Optimised Model	Recall	0.960	0.800	0.561	0.602	0.576	0.558
	F1	0.960	0.801	0.558	0.608	0.580	0.559
Optimised Cost-sensitive Model	Recall	0.406	0.715	0.927	0.071	0.132	0.046
	F1	**0.958**	**0.801**	**0.623**	**0.559**	**0.589**	**0.567**

Table 23 Results of the different models of Pure model for the Static mode.

The optimisation process improves the results by around 2%, but cost-sensitive techniques achieve a better improvement of up to 5%. The cost of enhancing recall is to reduce precision by 7%. To reach higher recall metrics, the algorithm prioritises guessing the minority class. Then, it is not an acceptable solution. On the other hand, F1 optimisation provides the best results for the ML model. The ML improves their values although the improvements are reduced, recall about 5%, F1 1% and precision decreases 3%.

Appendix C ML experiment for Dynamic mode

This appendix shows the process performed to train the regression ML model for Dynamic mode. Similar to Appendix B, it is shown the Pure model. PyCaret allows a first comparison between the ML models available and makes a comparison between their metrics. The performance metrics for the training set are performed based on a 5-fold CV. Notice that the values in the tables are the average value of the five folds. The results have been sorted concerning the RMSE metric. In this case, as there are three independent targets to be analysed (MinDis, DistoMinDis and TimetoMinDis), the best model for each one is selected and optimised. For the sake of clarity, it is only detailed in-depth the label MinDis. The same process has been repeated for the rest of the variables. Table 24 shows the metrics of ML algorithms for this target.

Model	MAE	MSE	RMSE	R2	RMSLE	MAPE
Extreme Gradient Boosting	1.832	8.727	2.954	0.991	0.199	0.285
Light Gradient Boosting Machine	1.991	10.067	3.173	0.990	0.209	0.350
Gradient Boosting Regressor	2.179	12.004	3.464	0.988	0.224	0.386
Linear Regression	2.313	14.160	3.763	0.986	0.254	0.356
Ridge Regression	2.313	14.160	3.763	0.986	0.254	0.357
Bayesian Ridge	2.313	14.160	3.763	0.986	0.254	0.357
Least Angle Regression	2.315	14.179	3.765	0.986	0.255	0.357
Orthogonal Matching Pursuit	2.293	14.271	3.777	0.986	0.256	0.341
Huber Regressor	2.263	14.301	3.781	0.986	0.251	0.336
Decision Tree Regressor	2.024	14.635	3.825	0.986	0.236	0.258
Lasso Regression	2.506	15.376	3.921	0.985	0.269	0.515
K Neighbours Regressor	3.975	32.717	5.720	0.968	0.368	0.921
Passive Aggressive Regressor	4.145	31.476	5.549	0.969	0.382	0.801
AdaBoost Regressor	7.510	83.060	9.111	0.919	0.637	2.549
Elastic Net	8.130	104.904	10.242	0.897	0.605	2.154
Lasso Least Angle Regression	26.181	1022.983	31.984	-0.000	1.082	6.984

Table 24 Metrics of ML algorithms applied to Pure model for MinDis.

The models with the best performance are Extreme Gradient Boosting, Light Gradient Boosting and Gradient Boosting Regressor. Extra Trees and Random Forest presented errors during the simulations that avoided obtaining consistent results and were discarded. The main characteristics obtained are:

- All models present an RMSE between 3 and 6 NM and an MAE between 1 and 5 NM. The metrics are similar between them, particularly with the ensemble methods varying from only 0.5 NM. The RMSE value is barely 1 NM higher than the MAE value in most models.
- The only model with an RMSE below 3 NM is the XGboost (2.9 NM). It confirms previous outcomes about ensemble methods that provided the best results.
- Most ensemble models present an R2 value relatively high, between 0.99 and 0.98. This means that the features and the targets are highly correlated, as will be analysed in the feature selection.

Table 25 shows the results obtained for the different safety metrics applied to the validation set.

Target	Model	MAE	MSE	RMSE	R2	RMSLE	MAPE
MinDis	XGboost	1.836	8.782	2.963	0.991	0.198	0.294
DistoMinDis	XGboost	2.113	16.327	4.041	0.967	0.259	0.313
TimetoMinDis	XGboost	18.681	1126.601	33.564	0.966	0.400	0.326

Table 25 Results for the validation set of Pure Model for the Dynamic mode.

Compared with the results obtained for the training set using CV, the validation-set results are very similar. This indicates that there is no problem with overfitting with this model. Figure 31 shows the distribution of the predicted values of the different targets.

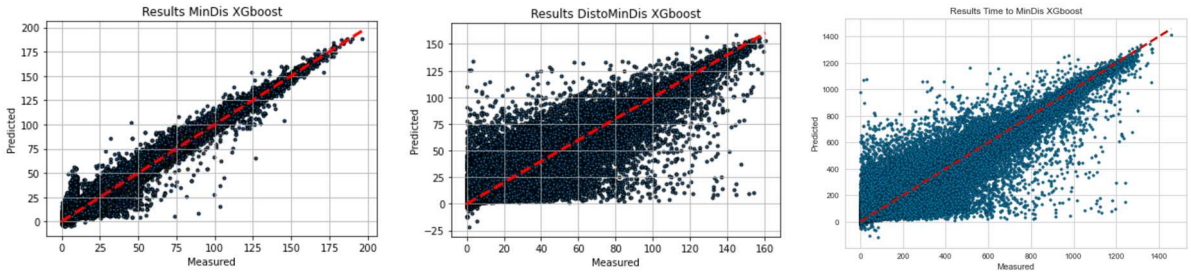

Figure 31 Predicted vs Measured values of the left) MinDis, medium) DistoMinDis and right) TimetoMinDis for the non-optimised algorithm of the Pure model.

The three targets improve their values with the same model. The predictions show inefficiencies close to 0 values because they provide predictions lower than 0, i.e., the predictions acquire negative value. These values should not be accepted during the prediction process. MinDis reduces its RMSE value from 6.6 (Static) to 2.9 NM, although the individual error can be up to 20 NM in some cases. DistoMinDis reduces its RMSE value from 12.3 (Static) to 4.0 NM, although the individual error can be up to 75 NM. TimetoMinDis reduces its RMSE value from 100 (Static) to 33 seconds, although the individual error can be up to 600 seconds. The conclusions are clear about the Dynamic mode's improvement because they reduce the Static values up to three times.

The feature selection identifies those features that do not contribute to the model and increase its complexity. The feature selection is performed for all targets because they can influence each differently to each one. Figure 32 shows the feature influence on MinDis.

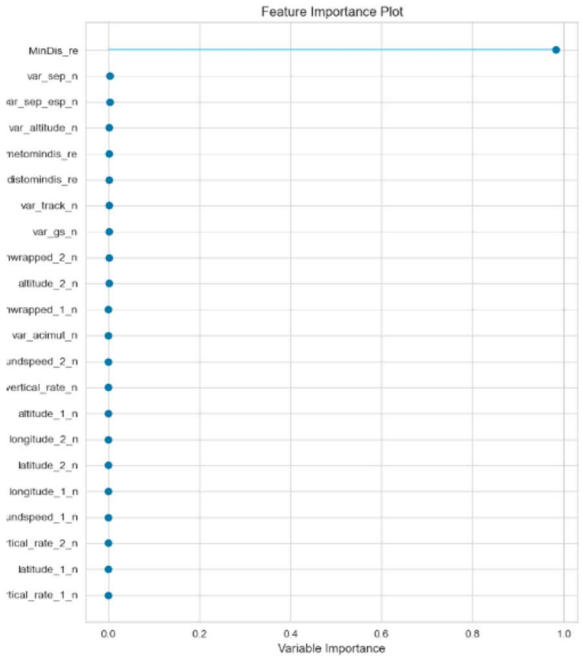

Figure 32 Feature importance of Pure model for MinDis.

The prediction of the MinDis is the most significant factor, close to 100%. Then, for this model, it is crucial to have a better prediction of the MinDis. The rest of the features barely influence the model. It is important to note that the predictions of the other safety metrics are neither significant. For the other two safety metrics, the feature influence can be seen in Figure 33.

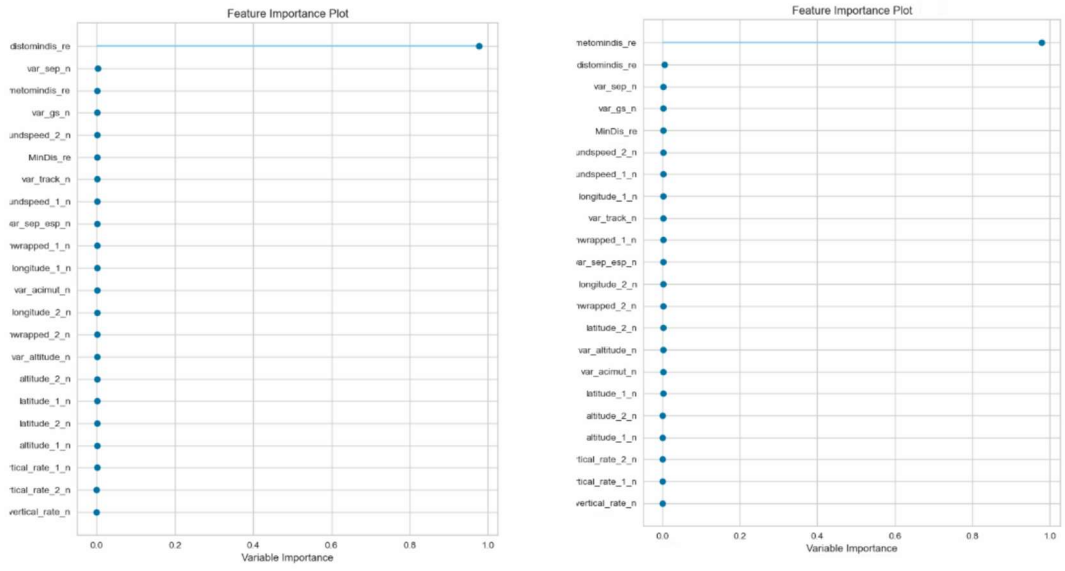

Figure 33 Feature importance of left) DistoMinDis, and right) TimetoMinDis for the non-optimised algorithm of the Pure model.

For both targets, the feature with more significant influence in the model predicts the same target used in the model near 100%. The rest of the variables do not present barely influence in the model.

It is analysed the model by removing the targets predicted as inputs (both DistoMinDis and TimetoMinDis). The model worsens the results from 32 seconds to 81 seconds (more than 100% increase). These results confirm the crucial importance of having a prediction of the safety metrics.

In order to optimise the model, it is necessary to analyse the behaviour of the model, variating the hyperparameters. There are three targets to be improved so that the optimisation will be performed for each model. A grid search is implemented with a 5-fold CV. The metric selected to be improved the RMSE. Table 26 shows the results for the optimised models.

Target	Model	Optimizer	MAE	MSE	RMSE	R2	RMSLE	MAPE
MinDis	Baseline		1.832	8.727	2.954	0.991	0.199	0.285
	Optimised	RMSE	1.601	6.652	2.579	0.994	0.178	0.238
DistoMinDis	Baseline		2.113	16.327	4.041	0.967	0.259	0.313
	Optimised	RMSE	1.944	13.661	3.696	0.973	0.244	0.280
TimetoMinDis	Baseline		18.681	1126.601	33.564	0.966	0.400	0.326
	Optimised	RMSE	16.984	1064.816	32.631	0.967	0.367	0.293

Table 26 Metrics for the optimised Pure model for the dynamic mode.

The optimization process provides different improvements on the RMSE values of the different targets: 13% for MinDis, 7% for DistoMinDis and 2% for TimetoMinDis. These results provide different improvements than the Reference and Hybrid model. In conclusion, the Pure model provides the best results for DistoMinDis and TimetoMinDis; they have fallen by half.


University of Zagreb


Technische
Universität
Braunschweig


POLITÉCNICA
UNIVERSIDAD
POLITÉCNICA
DE MADRID


skyguide


zh
aw
School of
Engineering
ZAV Center for Aviation


JKU
JOHANNES KEPLER
UNIVERSITÄT LINZ


onsulting
Slot