EXPLORATORY RESEARCH

JOINT UNDERTAKING

D 4.1: Complete data pipeline description and ML solution

Deliverable ID:	D4.1
Dissemination Level:	PU
Project Acronym:	SafeOPS
Grant:	892919
Call:	H2020-SESAR-2019-2
Topic:	SESAR-ER4-06-2019
Consortium Coordinator:	TUM
Edition date:	18 May 2022
Edition:	00.02.00
Template Edition:	02.00.03

EUROPEAN PARTNERSHIP

Co-funded by the European Union

Authoring & Approval

Authors of the document

Name / Beneficiary	Position / Title	Date
Pablo Hernández	WP4 Leader	13/05/2022
Lukas Beller	Project Officer	31/03/2022
Phillip Koppitz	Project Team Member	31/03/2022
Clara Argerich	Project Team Member	31/03/2022

Reviewers internal to the project

Name / Beneficiary	Position / Title	Date
Lukas Beller	Project Officer	18/05/2022
Phillip Koppitz	Project Team Member	31/03/2022
Damir Valput	Project Team Member	31/03/2022

Reviewers external to the project

Name / Beneficiary	Position / Title	Date

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

Name / Beneficiary	Position / Title	Date
Pablo Hernández	WP4 Leader	31/03/2022
Lukas Beller	Project Officer	31/03/2022

Document History

Edition	Date	Status	Name / Beneficiary	Justification
00.00.01	15.02.2022	Initial Structure	Pablo Hernandez / INX	
00.00.02	25.02.2022	Initial Content	Pablo Hernandez / INX	
00.00.03	15.03.2022	Final Draft	Pablo Hernandez / INX	Work provided by TUM and including ML results
00.00.04	29.03.2022	Initial Word Export	Pablo Hernandez, Ines Gomez / INX	Exporting InGrid content into SJU Template
00.01.00	31.03.2022	Initial Release	Pablo Hernandez / INX	
00.02.00	13.05.2022	Second Release	Pablo Hernandez / INX	Addressed SJU comments

SafeOPS

FROM PREDICTION TO DECISION SUPPORT - STRENGTHENING SAFE AND SCALABLE ATM SERVICES THROUGH AUTOMATED RISK ANALYTICS BASED ON OPERATIONAL DATA FROM AVIATION STAKEHOLDERS

This Complete data pipeline description and ML solution is part of a project that has received funding from the SESAR Joint Undertaking under grant agreement No 892919 under European Union's Horizon 2020 research and innovation programme.

Abstract

The **SafeOPS** project aims at investigating the impact of possible **artificial-intelligence-based decision-support systems** on routine air-traffic operations. The context selected for this investigation is the missed approach initiated by the flight crew of a landing flight and the subsequent go-around. The go-around scenario has a number of uncertainties and therefore makes it an ideal candidate for the integration of a predictive technology to support air traffic controllers (ATCO's) in managing aircraft in this situation.

To this end, three main pillars were defined in the project to develop the solution: **Operational Layer (OL)**, **Risk Framework (RF)** and **Predictive Layer (PL)**. The latter, which is developed within Work Package 4 of the SafeOPS project, addresses all big data and AI related tasks. It focuses on two main objectives. The first one covering all the related actions for the creation of the necessary automated data pre-processing and preparation pipelines. The second focuses on the AI/ML solutions for the predictive analytics that will be chosen and trained with a special focus on the human interpretability aspect of the solution. The trained AI/ML solutions will be deployed, delivering the predictive analytics to the Risk Framework (RF).

This report addresses the initial phase of the process in the compilation of the Predictive Layer. The report aims to provide an in-depth view of the **data infrastructure** developed for the project as well as the **data processing pipelines** that are responsible for structuring, fusing, feature engineering and labeling of the data. In addition, this report will also include the definition of the **case studies** of interest for the project, an initial exploration of the data collected and finally the training of the different **AI/ML solutions** for one of these case studies, analysing both their **performance** and carrying out a first study of their **explainability** and **interpretability**.

Table of Contents

List of Tables

List of Figures

Co-funded by the European Union

1 Introduction

1.1 Project overview

SafeOPS investigates the impact of **artificial-intelligence (AI) based decision-support systems** on routine air-traffic operations. Thereby SafeOPS focuses its research on a common decision-making paradigm in digitalization and predictive analytics, "**from prediction to decision".**,. The envisioned decision support concept can be summarized by expanding the current Air Traffic Management (ATM) system with an information automation-based decision intelligence. Information Automation describes the automated acquisition and processing of operational performance data through big data technologies and AI algorithms, providing new information to the ATM systems.

The **scenario** selected in SafeOPS for investigating decision support concepts in ATM is the **go-around** handling of Air Traffic Controllers (ATCOs). The go-around scenario has a number of uncertainties and safety critical factors associated with it. It is therefore an ideal candidate for studying the integration of a predictive technology, with the aim of providing greater support to ATCOs. For the selected go-around scenario, the project develops an integrated model of risk, incorporating potential uncertainties. The model allows discussing safety scenarios in a coherent, probabilistic approach. It will include historical aircraft, weather and traffic data, and the outcome of AI/ML models. The computed risk is added information, which flows into the planning and operational management of the overall ATM system. Using this approach, potential risks could be actively managed.

The question the SafeOPS project looks to answer is, how the nature of these information will change the way the system is operated. Beyond "information overflow", human operators using AI/ML systems will have to adapt not just to more information, but especially to the **probabilistic nature of this information**. While potentially very powerful, AI/ML systems are far from being deterministic; even though the advances of explainable AI allows for more transparent ML models than only black box solutions. Still, users will have to understand and interpret (to a greater or lesser extent) correctly the probabilistic nature behind these systems. Clever HMI refinements will certainly help to mitigate the potential overflow of information. However, also research on the impact of information automation on the ATM system needs to be conducted. It must show that an increase of capacity and cost-efficiency can be achieved and also the resilience of the system is maintained or further improved. SafeOPS aims to foster a collaborative paradigm that involves both the world of ATM and the world of airline operations to identify possibly hidden safety risks.

The work presented in this deliverable focuses on the more technical side of the project including the development of a data pipeline that allows automated creation of data sets, necessary for training and validating predictive AI/ML models and the creation of the first predictive AI/ML models to generate an understanding of the achievable accuracies and reliability of go-around predictions based on operational data. Thereby, this document builds on previous work done in the project (available at projects webpage https://safeops.eu/ or the official European Commission research results homepage https://cordis.europa.eu/project/id/892919/results), mainly:

- Investigate concepts for the **integration of AI/ML based decision support tools in ATM**, and evaluate the effects on capacity **safety and resilience** of the ATM operation. Several potential Use Cases were identified for a data-driven decision support tool in the go-around scenarios. This work can be found in **Deliverable 2.1** of this project (SafeOPS D2.1).

- **Enhance risk assessment methods**, such that they can cope with the introduced AI/ML component. For this purpose, a systematic review of current risk models available for application in the aviation context was conducted. The review provided a critical assessment of the existing risk models and their suitability for use in the SafeOPS Risk Framework. This work can be found in **Deliverable 3.1** of the project (SafeOPS D3.1).

1.2 Deliverable objective

This is the first deliverable of **Work Package 4 (WP4)** of SafeOPS. The overall objective of this work package is to perform all the technical tasks related to the development of an AI/ML solution (**Predictive Layer**) and provide a technical summary of the work developed during the first phase of the work package. Specifically, based on the user and functional requirements defined in WP2 (SafeOPS D2.1), we will assess different AI technologies. The work done in adapting the **BeSt (DataBeacon)** platform to allow the automated processing of selected data sources will be presented. For this purpose, specific **data pipelines** including data structuring, merging and labelling have been developed. In addition, this deliverable also collects the **feature engineering** process, by which we create new variables with meaningful information to describe the proposed operational scenarios. Additionally, four case studies are defined that will be performed throughout WP4, of which the first one is completed within the works, covered in this document. This includes a complete development of the **first case study** proposed both for **Munich airport (EDDM)** and **Frankfurt airport (EDDF)** where different **AI/ML solutions** were trained, which allowed a thorough investigation of these algorithms, comparing their performance levels. Finally, a first research on the **interpretability** and **explainability** of the developed models is presented, which builds the bases for the work done in Task 4.2.

1.3 Deliverable structure

The present deliverable includes the following sections:

- Section 2 contains an overview of the general **data infrastructure** developed to suit the needs of the project
- Section 3 contains the required **data preparation** tasks, provides a **data quality assessment** of the different data sources available, what **data verification and validation** steps are needed and an introduction to the concept of **feature engineering**;
- Section 4 establishes the **problem definition** from a machine learning perspective and introduces the **case studies** proposed for the project;
- Section 5 goes into the definition of the **processing steps** with emphasis on the **labelling** of go-arounds and **feature engineering**, an **exploratory analysis** of the processed data is carried out and the first models for one of the case studies are trained and evaluated;
- Section 6 contains the **conclusions and next steps**.

Co-funded by the European Union

2 Data infrastructure

2.1 BeSt Platform

The SafeOPS project relies on the big data infrastructure provided by **DataBeacon** for all its technical needs. This infrastructure, a service which started in the SafeClouds.eu project and run by Innaxis, focuses on data ownership, confidentiality and data protection. DataBeacon operates the **BeSt platform**, a data platform for AI applications in aviation. BeSt is a scalable, secure, on-demand multi-side computing and data storage platform that allows fast deployment of AI applications in aviation.

BeSt (BeaconStack) is a **multi-sided platform (MSP)** for **Artificial Intelligence** (AI) applications specifically designed for the **aviation** domain. In an MSP, participants are usually both **data providers and consumers of data analysis services.** Participants interact through the MSP using **secure common exploitation of data** to improve their performance among various aspects of their business. These interactions are funded over an **open, collaborative IT infrastructure that operates under a global governance model**. The goal is to consummate matches among users and facilitate the exchange of data and applications, thereby enabling value creation for all participants. Figure 1 below shows the overall structure of the platform BeSt and how the different blocks connect and interplay. BeSt uses a data de-coupling architecture, which means a data 'broker' sits in between data sources and data analysts. No analyst can directly access the data and only the data broker has access to the data repositories. This adds an additional layer of security. The analysts in SafeOPS work in the block "App / Analytics Environment”. The admin functions of de-coding, formatting and de-identification enable data pre-processing that can ensure that the data privacy is respected, giving at the output the data that the analyst can work with without compromising data privacy.

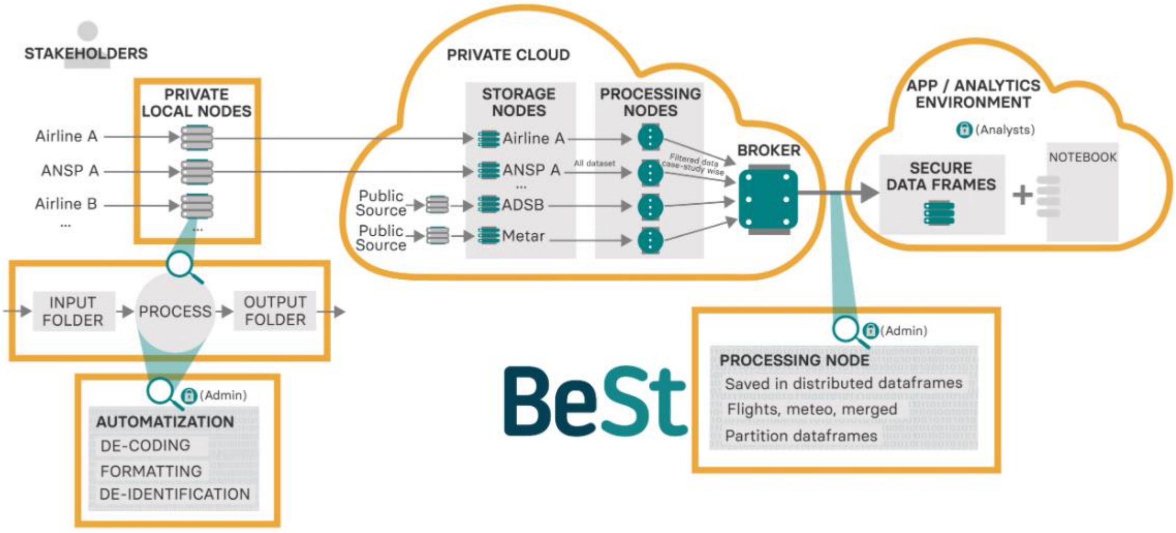


Figure 1: BeSt architecture

Co-funded by the European Union

2.2 SafeOPS Infrastructure

2.2.1 Amazon Web Services S3 Data Lake

The first key element of the SafeOPS infrastructure is a data lake repository for the storage of all the relevant data during the different phases of the project. A **data lake** is a centralised repository that allows us the storage of **both structured and unstructured data** in its natural/raw format (usually object blobs or files), at any scale. Compared to more traditional databases structures where data is stored in files or folders, a data lake uses a flat architecture and object storage to store the data. This allows data lakes to retrieve data across regions, improve performance and take greater advantage of the data. Currently, almost all data infrastructure providers, such as Google, Oracle, Microsoft, Teradata, Cloudera, MongoDB, or Amazon now have data lake offerings. Among the main advantages, a data lake has compared with more traditional data storages we can find:

- **Data volume**. Data lakes outperform most traditional data storages systems in easily adapt to a growing volume of data. The cost of using a data lake is a function of usage, so the users don't pay more than what they actually utilize.
- **Data variety**. By design, data lakes are able to handle various data sets and formats at the same time. This allows its users to progressively refine the data without the need of constantly changing the platform. Since the data lake in a project is the storage of all the data that is used, it is always up to date and contains any new data sets acquired or updated.
- **Self-service tools**. Data lakes enable users to use different tools and languages to perform different analytics tasks all at once.
- **Centralised data catalogue**. A data lake that is centralised eliminates problems with data silos (like data duplication, multiple security policies and difficulty with collaboration), offering downstream users a single place to look for all sources of data.

Data lakes offer multiple benefits compared to other systems and in particular can help manage datasets in research projects to a large extent, making research more agile as well as helping facilitate version tracking. In addition, because data in research projects is often varied, the data lake lends itself naturally to the data needs of a research project. Among the providers of these services mentioned above, access to the **Amazon Web Services S3 data lake** is available through the BeSt platform.

During the creation of the data lake, as well as for the development of the different phases of the project, some of the industry's best practices were followed, in particular the Bronze/Silver/Gold schema for the SafeOPS data lake. An overview of this schema can be seen in Figure 2. Far from being a key element, the use of this schema allows the efficient organisation of the data according to their degree of transformation and at the same time ensures the quality and verification of the data for its use in the predictive models. According to this scheme, three main levels are established depending on the state of the data:

- **Bronze**: This includes all the data in its original format (raw data). No processed data is stored at this level.
- **Silver**: This level includes data that have already undergone some transformation (e.g. filtering, outlier removal, duplication removal, cleaning, etc.) and can be used for data exploration analysis and problem definition.
- **Gold**: This level contains the highest quality data for the ML problem being addressed. All the necessary aggregations and feature engineering has been done and it is ready for its use in the predictive models.

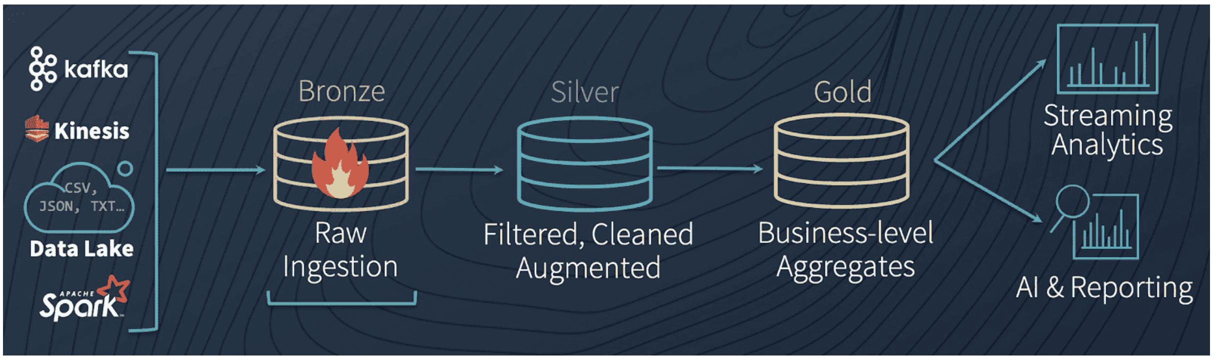


Figure 2: Bronze/Silver/Gold data lake schema (Databricks blog)

The architecture built for the data lake used in the SafeOPS project is shown below in Figure 3. It can be seen that, although the overall structure of the lake is initially fixed and should be changed as little as possible, changes are allowed by making the actual content of the files and folders in those buckets subject to dynamic changes according to the needs of the project.

```
BUCKET: databeacon-SafeOPS  /input
        /adsb-opensky
            /airport year month day
                XXX.parquet
        /adsb-safeclouds
            /relasexxx
                /XXX.parquet
        /metar
            / airport year
                /XXX.csv
        /metar-safeclouds
            /relasexxx
                /XXX.parquet
        /fdm-iberia
            /realeasxxx
                /XXX.parquet
        /fdm-pegasus
            /realeasxxx
        /etc.
    /share
        /sources
            /adsb-opensky
                /airport
                    /arrivals
                        /year
                    /departures
                        /year
            /metar
            /etc.
        /feature
            /ML_CS_01
            /ML_CS_02
            /etc.
        /training
            /ML_CS_01
                training_lgbm_1.parquet
                training_lstm_1.parquet
            /ML_CS_02
                training_model_1.parquet
                test_model_1.parquet
        /etc.

   /samples
        /source
            ...
        /feature
            ...
        /training
            ...


BUCKET: databeacon-local-link-iberia/ #stores FDM data provided by
Iberia, private

    preliminary-data/
        ...
    share/
        ...
    validation/
        ...

 BUCKET: databeacon-local-link-pegasus/  #stores FDM data provided by
Pegasus, private

    preliminary-data/
        ...
    share/
        ...
    validation/
        ...
```

Figure 3: SafeOPS data lake architecture

The SafeOPS projects relies on three different buckets in the data lake:

- databeacon-SafeOPS
- databeacon-local-link-iberia (Private)
- databeacon-local-link-pegasus (Private)
- Also: databeacon-SafeOPS-backup. This bucket has been an introduced as a back-up bucket of 'databeacon-SafeOPS', and it is periodically synchronised with it.

In principle, the reason for having thesebuckets is to be able to establish the necessary permission policies to access the data for different users in order to guarantee compliance with the Data Protection Agreements signed with the owners of the data. Specifically, the 'databeacon-local-link-XXXX' buckets are established as private and can only be accessed by users explicitly authorised by the corresponding DPA signed with Iberia and Pegasus. The databeacon-SafeOPS bucket can be considered main bucket of the project and contains all relevant data sources in different transformation stages and with the corresponding de-identification status when needed. The nature of the data in this bucket is organised as follows:

- **input**: This corresponds to the **Bronze level**. Data is stored as it is ingested from its source with hardly any modification (e.g. file format). This data is in principle immutable, and there will be no versioning. The only modification that can occur is in the increase of the data available by loading more data (e.g. more years).
- **share**: The data will be ready and prepared to be consumed by the data analysts.
    - **sources**: It contains the first set input data transformed (e.g. filtering, resampling or merging) for the proposed case studies. **Silver level**.
    - ***features***: It contains the different features computed in the project. Features might be grouped both by source dataset or by case study. Whenever a new release of features is performed adequate identifiers should be used for versioning purposes. **Silver level**.
    - ***training***: Data sets to be used to train ML models, as well as test and validate them. In this case, the data should include the relevant features and be accordingly labelled. As in the previous case, suitable identifiers are required for versioning and linking the developed models with the data sets used in their training. **Gold level**.
- ***samples***: The architecture is identical to */share* partition. It contains small samples (e.g. 1 day, some rows, etc.) of each dataset.

2.2.2 Databricks: SaaS for ML model development

The next key piece in the infrastructure developed for SafeOPS is the **Databricks platform**. The objective of Databricks is to provide a unified, open platform for all the data, a team of data scientist and developers uses in a project. It provides unique tools to data scientists, data engineers, and data analysts in a simple **collaborative environment** where they can run interactive and scheduled data analytics workloads. The strength of Databricks lies in its build upon popular **open-source projects** such as Apache Spark, Delta Lake, MLflow and Koalas to deliver a true lake architecture, combining the best of data lakes and data warehouses for a fast, scalable and reliable data platform.

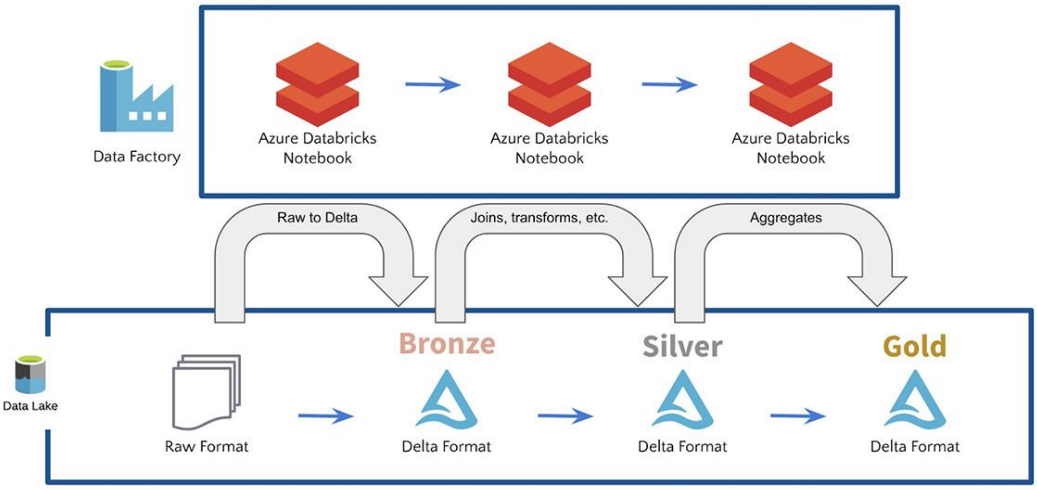


Figure 4: Extract, Transform, Load (ETL) pipeline using Databricks (taken from Databricks blog)

Databricks is built natively for the cloud and it leverages on low cost cloud object storage services such as AWS S3, which we rely on SafeOPS. It provides a single environment in which to work and perform all the tasks required in a data modelling project from data cleansing and transformation, feature development as well as training and validation of the predictive models. The user is able create and launch different clusters with multiple machines, each composed of different CPUs and/or GPUs. Furthermore, Databricks also facilitates the creation of policies to define cluster size limits and cluster configuration. Users are also able to create collaborative notebooks using Python, SQL, Scala or R (at SafeOPS we limit ourselves to working mainly with Python). Notebooks can be shared with other collaborators and can also be linked to external repositories (e.g. Github) for more effective and efficient code development. Finally, in addition to using notebooks for exploratory data analysis, Databricks also allows for powerful integration with machine learning frameworks like MLflow that facilitate training a model and testing it. Additionally, there is also MLflow Experiment tracking, which allows us to track the previous experiment runs, and look how important variables like accuracy changed over time.

In SafeOPS a Databricks workspace called **"SafeOPS-workspace"** has been created for the purposes of the project. To perform the data tasks, we have also defined a cluster available to all users who have access to the 'SafeOPS-workspace' workspace. We consider this cluster as a kind of multi-purpose cluster whose computing power should be more than enough to cover most of the computing tasks in SafeOPS, and therefore should be a default choice for most users in most situations. In addition, it is expected that on some occasions (e.g. model training) a higher performance cluster may be required. Although this cluster has not been defined at first, if necessary it will be created according to the objectives of the task. The configuration details of the multi-purpose cluster are:

- Cluster name: SafeOPS-all-purpose-cluster
- Databricks runtime 9.1 LTS (Scala 2.12, Spark 3.1.2)
- Cluster mode: standard
- Worker type: m4.xlarge, 2-8 workers (general purpose worker)
- Terminates after 90 minutes of inactivity

2.2.3 Github: software development and version control

The third and last key element of the infrastructure and tools used in the development of predictive models in SafeOPS is Github. GitHub is a **code hosting platform for version control and collaboration.** This software enables its users to work collaboratively and in an agile fashion on code development, keep track of the code development, changes and versioning. A specific Github repository we created to store SafeOPS code, including the developed data processing modules, machine learning models, as well as other auxiliary functions, is **"SafeOPS-ml"**.

Initially there will be no fixed structure for how code should be developed on Github. Analysts will be free to organise themselves as they see fit. Although it will be required that each model is developed on its own folder and a development structure will be suggested for ease of collaboration. The suggested structure for model development is:

1. **"00_exploratory_data_analysis"**: Exploration of the available datasets and their descriptive analysis, correlation analysis, etc.

2. **"01_feature_engineering"**: Creation of the features that will be used to train and test the model. This notebook should result in creation of at least two datasets, **train_set** and **test_set**.

3. **"02_machine_learning"**: Model training and validation

As mentioned, this structure is a generic one, and each analyst will adapt it as needed to the specific needs of their model. Moreover, the repository also contains different shared folders that serve for the creation of common data processing functions that can be shared between various models:

- **SafeOPS_toolbox:** This folder contains various python scripts and functions that implement a certain data processing task as well as functions for creating more complex features, label creation modules (labellers), unit conversion modules, cleaning modules, etc.
- **auxiliary_notebooks:** This folder is a repository of all notebooks that were created in the process of model development, e.g. descriptive or correlational analysis of a dataset.
- **libs:** This folder acts as intermediate storage during the development of helper functions before they are sufficiently mature and tested and are committed to the SafeOPS_toolbox

Finally, Databricks can be easily integrated with the Github account, so one can pull and push commits directly from the Databricks workspace to the Github repository as well as load the required helper functions from the SafeOPS_toolbox. To be able to do so, an analyst should generate a personal access token in their Github account with which they plan to access the "SafeOPS-ml" repository and add it to their Databricks workspace, under "Git integration" (follow the instructions at this link: https://docs.databricks.com/notebooks/github-version-control.html

2.3 Data Privacy

2.3.1 Data de-identification and Secure Data Frames

The main advantage of MSP is the combination of data from different sources to generate richer datasets that can power AI applications. But usually, the main obstacle to this is the particular privacy restrictions that different datasets may have. For example, in the case of SafeOPS, the use of flight data provided by the airlines is often restricted to limited use cases such as only using it in aggregate manner or without identifiable fields to ensure privacy of crew operations. These limitations may make comparing and merging such data with external datasets very difficult, which could limit their use in predictive models. Knowledge discovery uses general techniques to search for underlying patterns in data. Data mining techniques become completer and more accurate with more features and parameters included when finding the most probable cause of an event.

The current state-of-art solution for **de-identifying** the confidential data is to perform a **hashing operation**. The hashing function transforms each sensible data point such as date or callsign to a deterministic long alphanumeric string. This operation is a **"one-way" operation**, i.e., it cannot be reversed. Figure 5 shows how data is deidentified in the private nodes and then published on the shared environments for the analysts to work on. The **hashing-based message authentication code (HMAC)** is a specific type of message authentication code involving a cryptographic hash function and a secret cryptographic key. It may be used to verify both the data integrity and the authentication of a message, while making it impossible to retrieve the plaintext of the message. However, its strength lies in the fact that it still allows to identify whether two data samples belong to the same date (if de-identified) without data leakage and revealing the exact dates. It should also be noted that de-identification imposes some security limitations on data filtering: for example, if a sufficiently specific period (e.g. week, month, year) is known, the whole de-identification process could be compromised. To overcome these limitations, the data platform BeSt incorporates **Secure Data Fusion** technology developed in the SafeClouds.eu project. In a simplified way, Secure Data Fusion refers to the data pipeline that results in the creation of **Smart Data Frames (SDF)**. This SDFs are the de-identified (hashed) ready-for-analysis data frames that allow information from different sources to be merged without compromising privacy. In the SafeOPS project, practically all de-identification needs are covered by the BeSt platform, which provides the SDFs directly to the SafeOPS S3 storage infrastructure.

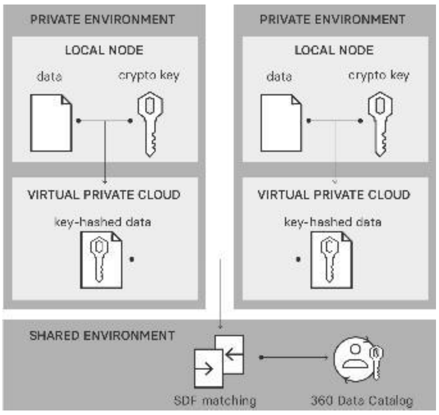


Figure 5: SDF cryptographic approach using HMAC and private key

2.3.2 Global governance model

As stated in previous subsections, the BeSt platform covers for the SafeOPS project large aspects of privacy, security and accessibility. BeSt is in charge of collecting sensitive raw data and de-identifying it as mentioned in the previous section. From the SafeOPS project, the governance model developed involves all those security and protection policies mainly in relation to the S3 storage infrastructure. The global governance model defines **who** has **when** access to **which** resources and gives the data owners the mechanisms to:

- **Join** or **leave** the platform at any time, removing any trace of their data.
- **Approve** or reject new case studies or applications.
- **Monitor** data access and usage.

AWS allows to create "**User Groups**" through which you can create and specify different permission policies applicable to one or more users. This allows SafeOPS to define and control the access limits of different users within the infrastructure. Two main user groups have been defined for the project:

- **(SafeOPS-data-managers) - Data engineer/manager**. User group in charge of maintaining and developing the data infrastructure of the project. Serves as the technical point of contact with the BeSt platform, third parties and data providers. Has access to all the raw data and administrator rights. The main tasks will include protecting (via de-identification or anonymisation) of the private data fields and merging and preparing the data from various sources. Has access to all three different buckets in the SafeOPS data lake.
- **(SafeOPS-devs) - Data scientist/analyst**. User group which main tasks include the development and deployment of the (machine learning) algorithms defined by the case studies. The users in this user group does not have access to the private buckets an only has access to the databeacon-SafeOPS bucket.

3 Data preparation

3.1 Extract, Transform and Load (ETL) pipelines

SafeOPS is a research project centred in the use of data for the detection and prediction of go-arounds. Therefore, different data sources are required, as information is scattered among several sources, such as meteorological or aviation sources. In a project involving a large use of data, it is mandatory to define a data treatment pipeline that serves as a baseline for the whole project, including steps like the data extraction, data transformation and data usage. Such pipelines are known as ETL pipeline (Extract, transform, load). The goal of the ETL pipeline is defining the processes for moving data from a given source or sources to a database (e.g. data lake). After the transformation process data shall be standardized and cleaned, so every user can access it at all times and find prepared data.

In order to ensure all available data is prepared and uncorrupted, it is important to have quality checks and data validations tools in the pipeline. The challenge on building pipelines resides on the requirements to meet: have uncorrupted and validated data (managing data quality), have a specific pipeline for the project needs (which limits reusability). Data engineers follow an "extract, transform, load" process, to extract data from a variety of sources, transform it into usable and reliable data and load it into the chosen storage platform. The role of analysts, data scientists or ML engineering is to access data and use it for modelling and prediction tasks. However, these roles may be merged sometimes due to the need of data scientists to deeply analyze and understand data. Also, sometimes valuable information may be lost in the preparing process, therefore it is important to understand the complete process and all variables in all data. For that purpose, the ETL pipelines may be substituted by ELT pipelines (Extract, Load and Transform), in which the data scientists take part into the transform process by means of deeply analysing data, prior to its final transformation. The three stages of these pipelines can be summarized as follows:

3.1.1 Extraction

The **extraction** can be done partially or fully, depending on the data needs and the system the data is extracted from. For instant, when regarding meteorological datasets, there exists available information for many past years. However, for the given project we may only need data of a selected year and a selected geographical area. Thus, a partially extraction of these data will be performed.

3.1.2 Transformation

In the **transformation** step of ETL (or ELT), the raw data extracted from the sources, needs to be transformed into a format, used by the applications/models one is going to work on. During this process, data gets cleansed, mapped and transformed, often to a specific schema, so that it meets operational needs. The cleaning process will depend on the nature of the data itself. Some data sources may be more corrupt than others, and several cleaning techniques will be required to transform data. It is important to note that during the whole data preparation process all data will be stored, that means, raw data is to be stored as extracted from the original source, and afterwards the prepared data will also be stored on another location. Raw data is not overwritten by results of its processing.

The first step in the data transformation process involves the use of well-known techniques that will help the readability and future work with the datasets. Some of the basic transformation techniques are here listed:

- **De-identification:** In the case of sensitive datasets that are subjected to a DPA we will need to de-identify them in order to work according to the established security measures.
- **Data Integrity:** The first step is to check whether some files are corrupted or missing.
- **Outliers removal and missing values identification:** some of the datasets will have missing values and those need to be consistently mapped with the same value, such as NULL or NaN along all datasets.
- **Out of range measurements:** Also, in some type of datasets there are values that are out of reasonable range due to mostly sensor malfunction, an example could be a negative altitude measurement. these values need to be either removed either replace with an interpolated value.
- **De-duplication:** Some datasets may have duplicate data, these needs to be removed for the sake of clarity.
- **Units homogenization:** A criteria is to be defined, such as the International Unit System and applied to all correspondent features along the datasets
- **Formatting of variables**: All features representing a variable shall have the same format, for example, date/time conversion.
- **Keys homogenization:** All tables or independent datasets that are meant to be used together for a given case scenario should share the same Key, so that the tables can be easily merged. This step is scenario dependent, so it is very important to establish the data sources required for each case-study prior to developing the key that will be used to merge the tables.

In the field of aviation, it is very common to deal with dynamic, time varying data (such as trajectories). This type of data is larger than static data (such as aircraft type, airline...) and requires the use of more complex transformation techniques:

- **Resampling**: A single flight may have generated information with a very high frequency, that means, thousands of entries for a single flight. Since we are working with big data and thousands of flights, one can imagine how the dimensionality increases and correspondingly, the computational effort required. Resampling can be used to re-define the points of each flight, that means, reducing the number of entries by means of extrapolation. On the other hand, this operation can be performed on the other sense, by interpolating points in the event of a flight presenting very few measurement points.
- **Creation of aggregates:** This is not a technique itself but a required transformation for some case-studies, for instance if we require the total amount of fuel consumed for one flight we will need to aggregate the consumed fuel for each segment.
- **Data splitting:** Some variables may be split into different variables, for instance we can extract the day of the week from a date and thus, create a new variable.
- **Filtering:** Noisy data requires cleaning by means of smoothing techniques such as: Kalman Filter, Median Gaussian Filter, moving mean, etc.

3.1.3 Load

The load stage is committed to save the transformed data into the selected storage platform. During the load process, it is key to have homogeneous data splitting definitions. In other words, all files need to be stored following the same partition, that being year, month, day partition for instance. And the format of the files shall be the same so it will ease the access and data usage for all users. In SafeOPS, the chosen storage platform for all datasets is the cloud infrastructure AWS, specifically AWS S3.

3.2 Data catalogue

For a more detailed description of the data (e.g. data provider, format or information contained) the reader is referred to the deliverable D2.1 (SafeOPS D2.1). In the table below, we provide a description centred on the **quality assessment and processing requirements** of the data initially selected for the project. It is important to remark that several transformation techniques will be required for every dataset such as: data integrity, missing values identification, unit homogenization, formatting of variables and key homogenization. Moreover, for large files containing information from large geographical areas (e.g. weather GRIB files and ADS-B files) it is important to select data according to different case-studies in order to reduce the dimensionality of the required datasets. The same applies to datasets, containing many variables. Thereinwewill need to identify the relevant ones, depending on the case-study and drop the rest. More information of the initial set of features selected for the case studies can be found in section 5.

Table 1: SafeOPS data catalogue

Data source	Data provider	Description	Time Frame available	Geographical availability	Quality assessment	Processing required
ADS-B	OpenSky network	Surveillance data that relies on aircraft broadcasting their identity, position and other information derived from on board systems (GNSS etc.)	20/4/2018 - ongoing	ECAC area	• Noisy data • Sampling period low (≈ 10 seconds) • Limited trajectory information	• Data Integrity • Out of range measurements • Outliers removal and missing values identification. • Units homogenization • Formatting of variables • Keys homogenization • Filtering • Resampling
FDM/QAR	Iberia and Pegasus	Aircraft performance data gathered from sensor inputs and stores the data in a bit data stream for recording onto Quick Access Recorder device.	Jun 2017 - Oct 2018 (Iberia) Jun 2017 - Apr 2019 (Pegasus)	Iberia's network and Pegasus's network	• Sampling period usually very low (≈ 1 second) • Parameters recorded with different sampling rate • Generally, the information is trustworthy and quality is overall good • Some variables have missing values • Limited airport coverage (airlines network) • Limited temporal coverage • Subjected to DPA	• De-identification • Data Integrity • Missing values identification • Units homogenization • Formatting of variables • Keys homogenization • Resampling • Removal of empty variables

Data source	Data provider	Description	Time Frame available	Geographical availability	Quality assessment	Processing required
METAR	Iowa state university	Routine aviation weather report of actual observed conditions at an airport or near one (e.g. wind, horizontal visibility, cloud coverage, QNH…).	2018 - 2021	Europe - Large and Medium Airports	• Sampling period high (30 minutes) • Missing values often • In METAR files there are often dynamic variables filled with the same value • Limited to on-ground measurements (less precise)	• Data Integrity • Out of range measurements • Outliers removal and missing values identification. • Units homogenization • Formatting of variables • Keys homogenization • Feature selection

3.3 Data verification and validation

Data verification is the process that starts after data extraction. Data verification is performed in order to determine whether the data has been properly extracted, keeping its integrity. Data loss shall be identified and relieved. Moreover, data verification is also performed to assess the data format and check whether all the extracted data follows the established formatting and partitioning.

On the other hand, after data has been extracted, verified and cleaned it needs to be validated. That means ensuring the quality of the data after the cleaning process has occurred. Some of the activities involved in **data validation** the following ones:

- **Data type**: All variables must be compliant with their data type, that means, if a variable is an integer, it shall only use characters from 0 to 9.
- **Range**: A flight might present points with an altitude lower than zero, which would be out of range data as no flights cannot fly underground.
- **Uniqueness check:** When dealing with unique variables, such as flight id, it is important to validated that there are no repeated values, as that would mean corrupted data in the dataset.
- **Consistency check**: This task is performed to ensure the consistency of the data. For instance, a flight cannot land before it has departed. In the case of go-arounds, a flight cannot have a go-around after it has successfully landed.

On the other hand, labelling needs to be validated. In the particular case of SafeOPS where go-arounds are to be detected and then labelled, it is very important to ensure this labelling is correct and can be trusted. This verification is to be done by means of two criteria:

- **Visual verification**: Flights in which a go-around is detected are to be displayed with a 3D trajectory and the data scientist and experts can visually check whether a go-around has actually happened.
- **Advisory board:** ATCO's in the advisory board know the average of how many go-arounds happen in a given airport. This data can be cross-validated with the number of go-arounds detected by the go-around detector used in SafeOPS.

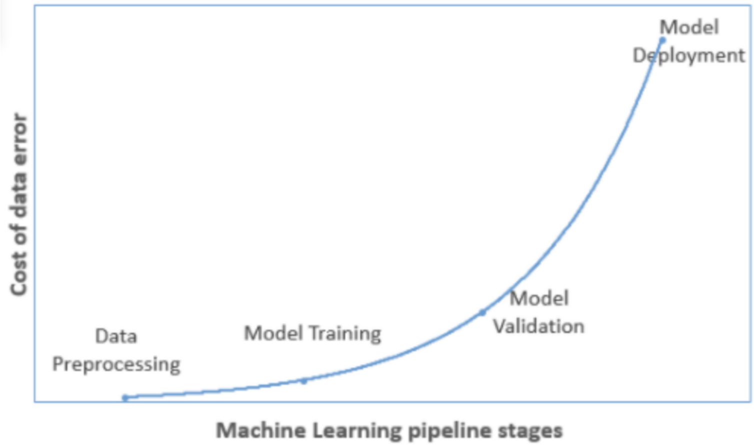


Figure 6: Cost of data error in the different machine learning pipeline stages (analyticsvidhya)

In the particular case of **Machine Learning models**, they are very sensitive to poor data quality, as a model learning from poor data could easily converge to inadequate solutions. Moreover, it is important to have a robust data validation and verification process that ensures quality data in the early stages of the whole process, see Figure 6. Otherwise, the corrupted data could easily propagate to the machine learning models and deployment thus requiring to re-do all the process once the error is identified.

3.4 Feature engineering

We can describe **Feature engineering** as the process of selecting and transforming the variables available in the raw data in order to create a predictive model using machine learning. This process involves a combination of data analysis and expert judgement. Although usually found within pre-processing it is thought appropriate to explain briefly in this section.

By "**feature**" in the context of predictive modelling, we are referring to a predictor variable. Feature engineering is the term, used for the process by which we create predictors so that a predictive model can be created. The dataset that will eventually be used in the prediction models consists of a matrix made up of rows and columns. Each of the rows of the matrix is an observation or a record and the different columns are the features that describe each record. Although many of the fields in the raw data can be used directly to train a model, it is often necessary to define additional features (feature engineering) created from the original data. These should aim to provide information that best helps to predict the desired records. There are several steps which will be **iterated** throughout the development of a machine learning solution:

- The first step in the feature engineering process is to **identify all the desired relevant predictor variables** that should be included in the model. The identification of these variables should not be limited to available data alone and should also involve consultation with **operational experts**, **users** as well as **relevant literature**. In SafeOPS, we have covered this step with the different workshops we have conducted with the ATCOs.
- The second step is the **creation** of the features themselves. This is where the raw data is manipulated to try to get the variables that have been defined and that are to be included in the model. During this stage, factors that need to be considered include the type of machine learning model (linear models vs non-linear models) or the ease of interpretation.

Finally, once all the desired features are extracted from the data, they have to be tested with the performance of the model. This is known as Feature selection. This refers to the decision about which features should be included in a model. Although at first glance it may seem obvious to include all available features in the model and have it automatically select the appropriate ones, this is not always possible, as sometimes the model cannot or is not designed to do so, and sometimes including all predictors can cause the model to generate unwanted or uninformative associations between them. It should be evaluated which ones are most helpful to the prediction and depending on the results, some should be removed.

4 ML Case studies

4.1 Problem definition

SafeOPS focuses on predictions of go-arounds and how these predictions can be used by Air Traffic Controllers (ATCOs) for their decision making. SafeOPS' deliverable D2.1 (available at https://safeops.eu/ or https://cordis.europa.eu/project/id/892919/results) laid out the scope of SafeOPS in defining scenarios, use cases and requirements for the project's concept and the concurrent developments.

The mission of work package 4, which is divided into Tasks 4.1 and 4.2, is the development of a machine learning solution, that can be used for time in advance prediction of go arounds, as described in the scenarios if D2.1 (SafeOPS D2.1), for further validating the concept in Task 2.2, later in the project. This deliverable focuses on the development of the data pipeline and an initial machine learning solution, whereas Task 4.2 will investigate several machine learning techniques focusing on interpretability for the defined use cases. The requirements relevant for work package 4 can be found in the functional requirements section of D2.1, specifically in subsection 3.5.1.4 - Big Data and Machine Learning Requirements and subsection 3.5.1.3 - Timing of Predictions.

Based on the requirements, D2.1 laid out the technical problem definition for the big data and machine learning related tasks in detail in D2.1 - section 4, which covers the topics of:

- **Data Acquisition**
    - including description of possible data sources with description
- **Data Processing Pipeline including**
    - Data Preparation
    - Data Cleaning
- **Model Training and Validation**

These aspects have again been theoretically described in the previous sections, establishing the foundation of the practical work done in Task 4.1. To further specify and plan the practical work, case studies have been defined for Work Package 4, which shall capture the operational aspects, defined in D2.1.

4.2 ML case study criteria

Having understood the problem from an operational perspective, the next critical step is to properly **translate it into a problem in ML terms**. Without a proper definition, even using the most powerful algorithms available, the predictive model results may be of no use to solve the original operational problem. When defining a new problem to be tackled with machine learning, a framework can be established whereby three simple questions are asked to establish the key parts of the problem and understand whether machine learning is appropriate or not. These questions are:

- **What is the problem? -** The first step is to establish and understand the original operational problem to which machine learning is intended to be applied. This part has already been covered previously in the project and can be found in D2.1 (SafeOPS D2.1)

- **Why does the problem need to be solved? -** Once the operational scenario is understood, the motivation and potential benefits behind applying machine learning must be established. Again, this part has already been dealt with earlier in the project and can be found in D2.1 (SafeOPS D2.1).
- **How to solve the problem? -** Define the parameters of a ML solution that responds to the operational problems identified.

To answer this last question and to define the different ML case studies proposed in SafeOPS, five areas have been established. These 5 areas are:

- **Expected/desired output**: Define what is the expected or desired outcome of the model.
- **Prediction horizon**: Set constraints on the possible prediction horizon within which the model will operate.
- **Geographical horizon**: Define the possible geographic constraints within which the model will operate.
- **Data available**: What datasets are available to be used by the model.
- **Labelling**: How the event to be predicted is to be identified.

4.3 Proposed ML case studies

The following proposed ML case studies will be presented in the context of the SafeOPS project. It is important to emphasise that it is not claimed that during the development of the project a solution will be found for all of them, but that they are a list of scenarios where machine learning techniques could be applied. Although the first two case studies are the ones that best fit the initial problem posed in the project proposal and therefore will be the main focus, we will try to explore as many as possible, being also flexible to the results and feedback received during the development.

4.3.1 ML Case study - Predictive scenario 1 (ML_CS_01)

- **Expected/desired output**: Binary classification problem (Go-around/No Go-around prediction). In addition, if possible, additional information on performance, such as the confidence of the prediction or feature importance.
- **Prediction horizon**: Distance (e.g. Nautical miles) from the runway threshold. In this case this distance will have to be defined as a trade-off between a high enough distance for the ATCOs to have time to react and not high enough to make prediction impossible. Based on the feedback received from the workshops with the ATCOs the minimum distance at which prediction would be useful is around **4NM** from the threshold. ATCOs mentioned that up to 4NM from threshold they still have time to make operational changes (e.g. do not authorise a take-off on the same runway or instruct a GA to the approaching aircraft). After 4NM all clearances and/or indications have to have been given and in this space nothing should be instructed (except in case of emergency) and the prediction could only be used as warning and to increase situational awareness.
- **Geographical horizon**: Specific airports (e.g EDDM and EDDF).
- **Data available**: ADS-B and METAR. FDM can be used in the exploration phase but not during the prediction. Features will be extracted from the prediction point up to 10NM from threshold in 0.5NM intervals (e.g., 4NM, 4.5NM, 5NM, 5.5NM, 6NM…)
- **Labelling**: Using ADS-B. FDM could be used as a form of validation.

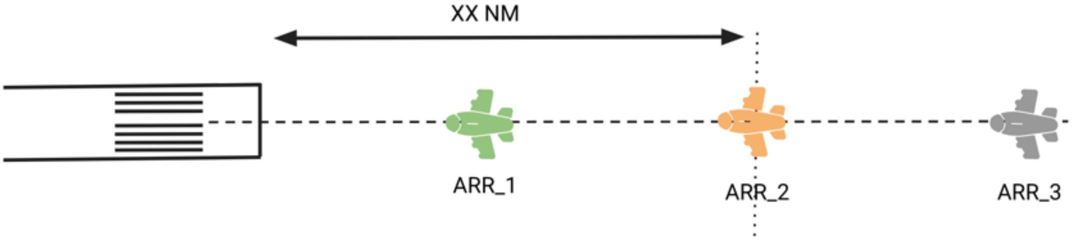


Figure 7: ML_CS_01 operational scenario

4.3.2 ML Case study - Predictive scenario 2 (ML_CS_02)

- **Expected/desired output**: Binary classification problem (Go-around/No Go-around prediction). In addition, if possible, additional information on performance, such as the confidence of the prediction or feature importance.
- **Prediction horizon**: Periodic time step within a predefined prediction window. In this case, there is no single prediction point, but the prediction is made according to the time step. For example, a monitoring area can be established from the threshold to 10NM and when an aircraft is within this area, the go-around prediction of the approach is updated every 10 seconds.
- **Geographical horizon**: Specific airports (e.g EDDM and EDDF).
- **Data available**: ADS-B and METAR. FDM can be used in the exploration phase but not during the prediction. Features would be extracted for every prediction point at the established prediction steps.
- **Labelling**: Using ADS-B. FDM could be used as a form of validation.

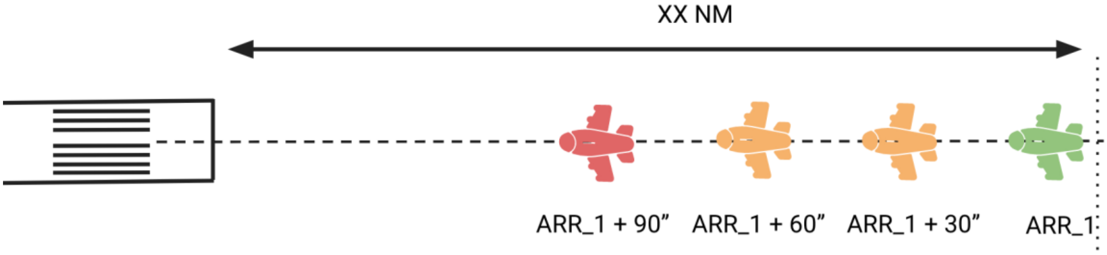


Figure 8: ML_CS_02 operational scenario

4.3.3 ML Case study - Predictive scenario 3 (ML_CS_03)

- **Expected/desired output**: Detection and identification of abnormal approaches
- **Prediction horizon**: N/A
- **Geographical horizon**: Iberia and Pegasus network
- **Data available**: All data sources (ADS-B, METAR and FDM)
- **Labelling**: N/A (unsupervised approach)

Co-funded by the European Union

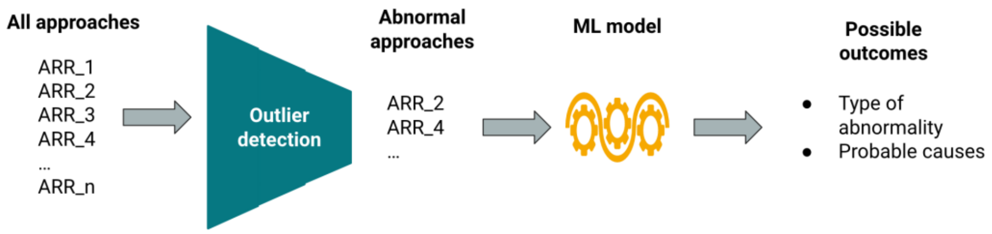


Figure 9: ML_CS_03 proposed workflow

4.3.4 ML Case study - Real-time detection (ML_CS_04)

- **Expected/desired output**: Flagging an initiated Go-around
- **Prediction horizon**: Real-time
- **Geographical horizon**: Any airport
- **Data available**: At the moment of detection only ADS-B
- **Labelling**: Using ADS-B

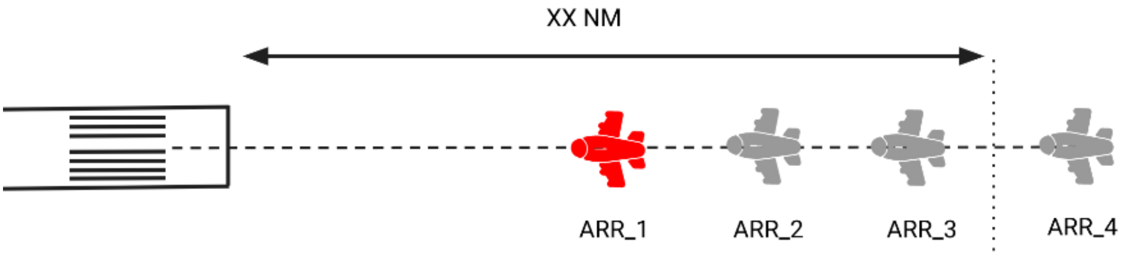


Figure 10: ML_CS_04 operational scenario

5 Predictive modelling

5.1 Machine Learning workflow

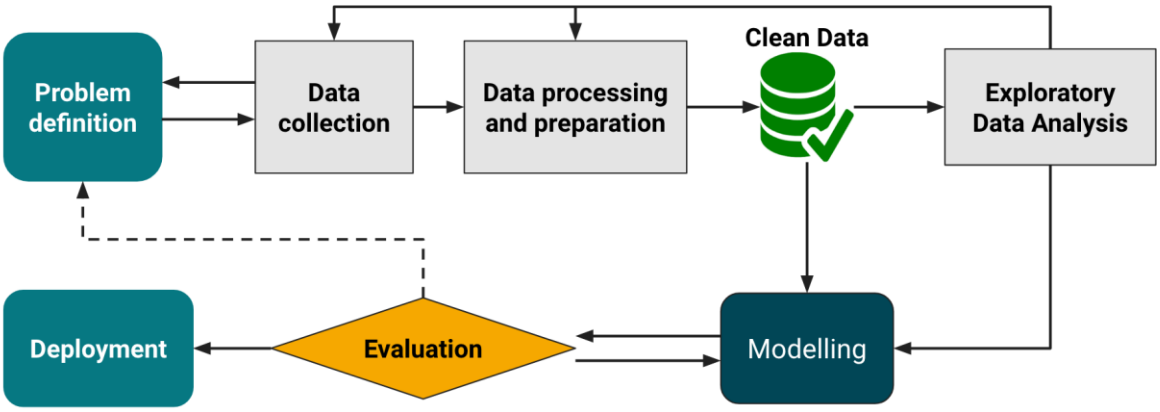


Figure 11: Basic Machine Learning workflow

Each project is unique and the workflow used often depends on the problem to be addressed and the solution sought. However, one can always distinguish a number of key tasks that every project must tackle. An example of a workflow for a data-driven project is shown in the Figure 11. This workflow is inspired by the **Cross-Industry Standard Process for Data Mining (CRISP-DM)** developed in 1996. We can distinguish 5 main phases:

1. **Problem definition + Data collection**: The first and most important step, as we have already mentioned, is to define and **understand the problem** to be solved from an operational perspective. In this phase, an assessment of the current situation should be carried out as well as establish what are the d**esired outcomes of the project**. In SafeOPS this has been done and can be found in the deliverable (SafeOPS D2.1). In addition, the objectives of data-driven project should be determined and a project plan should be drawn up to achieve the objectives. Finally, once the operational problem is understood and the objectives are established, the initial list of available data sources can be established and one can **start with the data collection**.
2. **Data processing and preparation + Exploratory Data Analysis**: After the problem definition and initial data collection, the second phase is where work with the data begins. The first step is to understand the data to be worked with. To do this, an initial exploration and description of the data is carried out (see (SafeOPS D2.1)). This is followed by a verification phase on the **quality of the data** (see section 3.2). After this, you can start with the preparation of the data by first selecting those data that you have decided are of the highest quality and most effective for the defined problem. Then proceed to clean the data using the **ETL pipelines** described in section 3.1. Once the data is cleaned, the final datasets are constructed by **merging** the different data sources and generating the considered **aggregations**. **Feature engineering** is also performed in this phase, generating new parameters by combining others. With this, what you could call "**Clean Data**" are obtained, which is the datasets ready to be used in the training of the prediction models.

3. **Modelling**: With the "Clean Data" dataset ready, one can proceed to the modelling phase. To do this, first you **select the modelling technique(s)** that best suits the needs of the problem. This should be done by documenting the modelling techniques to be used as well as the established **modelling assumptions**. Before training the model, a procedure or mechanism must be established to test the quality and validity of the model. After this, the selected model(s) can be trained. Finally, you should **assess and evaluate** the results of the models according to your domain-knowledge and the success criteria previously established.
4. **Evaluation:** The previous evaluation steps dealt with more technical factors of model performance such as the accuracy and generality of the model. This phase should assess the degree to which the model meets the **operational objectives** and try to determine if there are any reasons why the model is not performing as desired. A review of the development process should be conducted, summarising the main process activities and highlighting activities that may have been omitted or those that should be repeated. **Possible next steps** should also be identified, listing the reasons for and against each option. A validation plan will be developed in SafeOPS to carry out this assessment from an operational perspective.
5. **Deployment**: Although outside the scope of this project, the final step of the ML workflow would be the deployment of the solution in a real-world scenario. In the deployment phase, and based on the results of the assessment phase, the relevant **deployment strategy** will be determined. A **monitoring and maintenance plan** should also be established outlining the necessary steps and the appropriate monitoring and maintenance strategy. Finally, all the work done, the results obtained and the lessons learned should be **documented**.
6. **Iteration:** Although this is not a particular step, the need to iterate throughout the complete duration of the project should be emphasised. It is never possible to successfully complete a data-driven project in a single pass. New ideas must be constantly tested and process steps changed (e.g. extract new variables, rebuild and validate the model, realise that the parameters of the learning algorithm are not the best for the new feature set, find an inefficient subset of variables, etc.).

5.2 Go-around Labelling and Feature Definition

5.2.1 Processing Stages

The second part of the Machine Learning workflow described in the previous section constitutes the processing of data from raw data to a labelled data set with corresponding features utilized for the development of data models. In this section the processing stages are briefly described. An overview of the processing flow is depicted in Figure 12 and will be described in more detail in the following paragraphs.

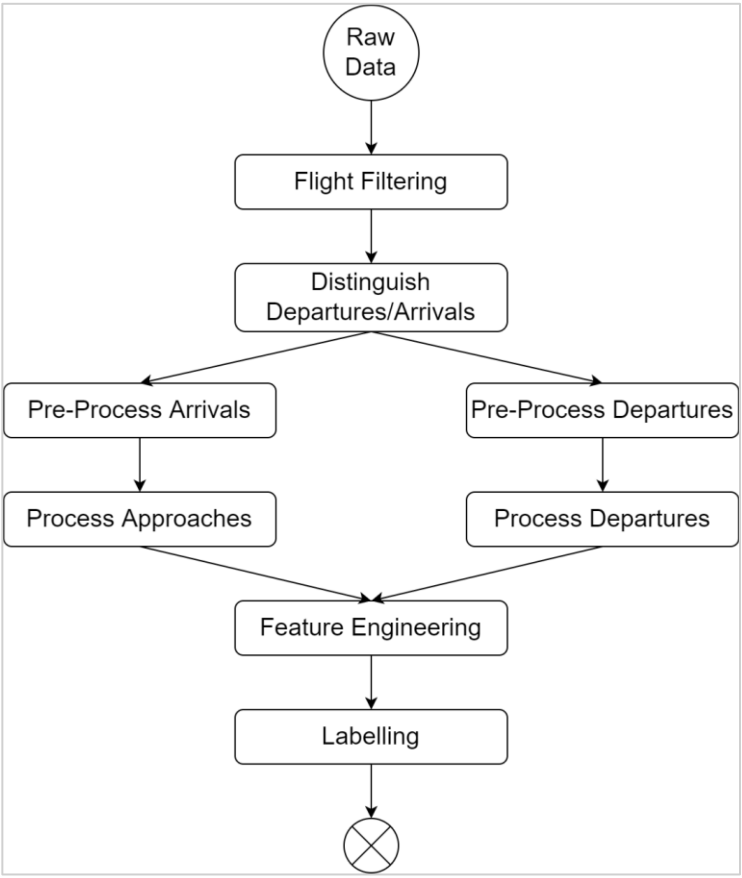


Figure 12: Data Processing Flow for ADS-B Data

The first step in the processing flow is to distinguish single flights out of the raw data set. The raw data contains all flights sending ADS-B messages within a defined area around airports of interest and are sorted by timestamp of each ADS-B message, not by flight. Single flights are filtered by the date and the callsign. This assumes that a single callsign is used only once per day which applies to commercial airline operations. With the data covering terminal areas of airports, data of single flights contain either the approach or the departure, not the complete flight trajectory. The next two steps after filtering for single flight data are applied separately to approaches and departures.

During the Pre-Processing of approaches, parameter units are transformed to SI units. Afterwards, data errors are removed. Here the pre-processing focuses on outlier removal and erroneous data points at the end of each approach. The two major errors at the end of recorded approaches are identical data in multiple data rows or follow up departures under the previous call sign. Especially the data error treatment at the end of each approach avoids erroneously detected runways because both errors lead to false track information. Part of the error treatment is checking the "On Ground" flag for plausibility. This check utilizes an engineered height above airport level to verify the "On Ground" flag and if discrepancies are detected to change the indication of the flag according to the height above airport level. The "On Ground" flag is the main indication to detect a landing in the ADS-B data. At the end of the pre-processing the separate flight phases contained in the flight data are detected. The flight phase detection is crucial for labelling the data based on identified Go-Around and described in more detail in the following section 5.2.2. The pre-processing of departures contains the same steps as for the approaches with minor differences in the specific algorithms.

Based on the flight phase detection data of single flights covering the approaches are divided into approach attempts, which splits the data in segments in case of a Go-Around and is the basis for the labelling. Here the collection of features for data models begins. For each approach attempt meta information about the flight (runway info, callsign, date, time) and specific time points (certain distances from threshold, certain heights above ground) are determined. Next, relevant weather information is drawn from METAR data and the approach type is added to the collection of features. Afterwards, features describing the approach performance are computed for each specific time point. More details about the features used are provided in section 5.2.3.

5.2.2 Flight Phase Detection

The basis of labelling flights with respect to go-arounds is a flight phase detection algorithm, that for a given flight (ADS-B or QAR) attributes the current flight phase to each index in the timeseries. Due to differences in QAR and ADS-B data regarding available parameters and resolution, two versions of the labeling algorithm are developed for SafeOPS. Both follow a similar logic, however the QAR version can resolve a higher number of different flight phases than the ADS-B version. This, however, is no problem, since SafeOPS is primarily interested in the approach, final approach, landing and go-around flight phases which both versions can detect. The flight phase detection algorithm for FDM data was initially developed for SafeClouds.eu, a predecessor project of SafeOPS. It is described in the SafeClouds.eu deliverable 4.3 (SafeClouds D4.3) which can be downloaded from the official European Commission EU research results hompage CORDIS https://cordis.europa.eu/project/id/724100/results. In the following subsections, the FDM version for the flight phase detection from SafeClouds.eu is revised. Additionally, the ADS-B version, adapted for SafeOPS is described.

5.2.2.1 Flight Data Monitoring / Quick Access Recorder Data

For FDM / QAR data, a flight phase detection based on a state machine shown in Figure 13 is used. This algorithm was described in the SafeClouds.eu deliverable D4.3 (SafeClouds D4.3) and is the basis of the go-around labelling. In this context, a specific flight phase is a state. The idea of the state machine is to define a set of possible states or flight phases that can be transitioned into from a current state/flight phase. For example, from the flight phase "Climb" it could be transitioned into "Cruise" or "Descent". For both options a logic is defined which describes the necessary behaviour of the flight data to be categorized as either one. For the "Cruise" the flight data needs to show no change in altitude (H=) and no vertical speed (VS=). Three different categories of behaviour are used. The first category are direct booleans from the flight data (e.g. parking brake flag). The second category are trends where timeseries are smoothed with a moving average and divided in a positive, negative or no trend. The last category are simple thresholds for certain time series. The transitions between the different phases in Figure 13 are abbreviated as follows:

- Booleans
    - P: is parking
    - A: is airborne
    - R: reversers out
    - F: flaps set
    - ¬: not true

Co-funded by the European Union

- Continuous time series used for trends and thresholds:
    - E: engine fan speed
    - H: pressure altitude
    - VS: vertical speed
    - V: airspeed
    - G: ground speed

- Trend definition:
    - =: trend remains constant / rate is zero
    - +: trend increases / positive rate
    - -: trend decreases / negative rate

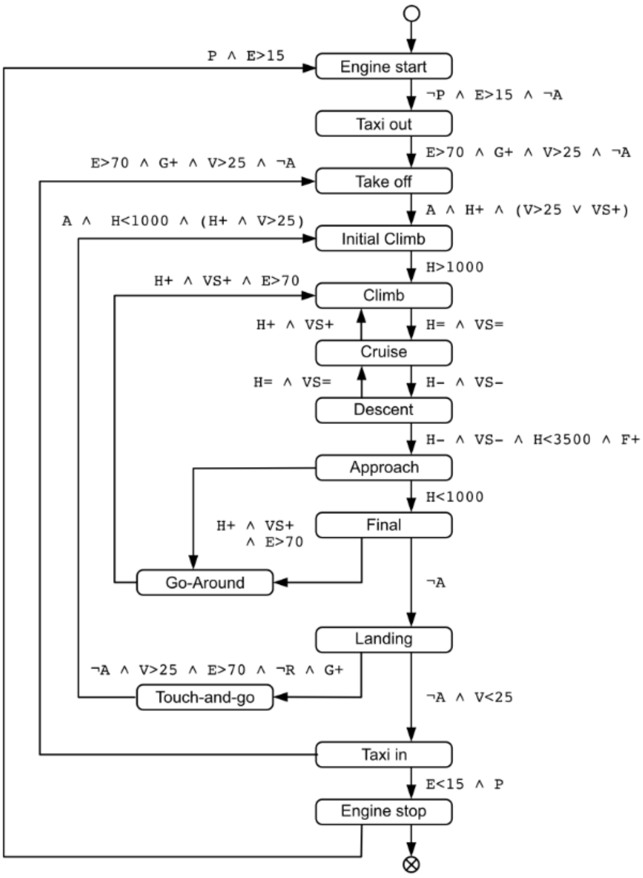


Figure 13: Flight phase detection logic for FDM data (SafeClouds D4.3)

A visualized example of the algorithm applied to FDM data is presented in Figure 14. The upper part of the figure illustrates the evolution of the flight phases of one example flight over the index of the flight data time series. The lower part of the illustration shows the pressure altitude. The colour coding illustrates the flight phase for the given index of the timeseries. From the profile of the pressure altitude, one can observe the first and second landing attempt, where the first attempt is labelled as go-around and the second as landing.

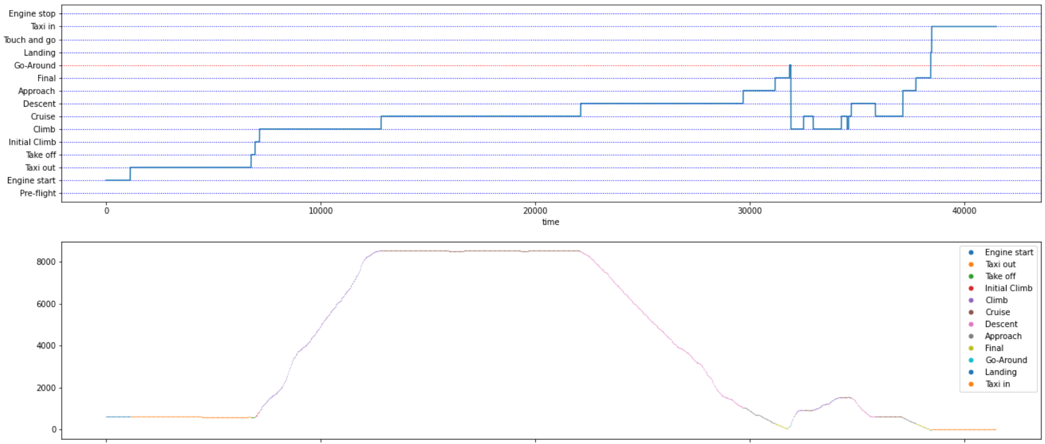


Figure 14: Flight phase detection for a single FDM flight

Based on the illustrated labelling of flight phases, grouping and slicing one flight into the relevant phases of Approach, Final Approach, Landing and Go-around is possible.

5.2.2.2 ADS-B Data Labelling

The ADS-B flight phase detection is a modified version of the flight phase detection described above, adjusted for the ADS-B data utilized in SafeOPS. Given the fewer parameters available in the ADS-B data, the ADS-B version has simplified transition conditions, compared with the QAR version. The state machine used to detect flight phases in ADS-B data follows the scheme illustrated in Figure 15. The same abbreviations as in Figure 13 are used. Compared with the QAR version, the ADS-B version has a shortcut from the Pre-Flight phase directly into the Descent phase, as the ADS-B data is directly filtered for terminal areas of the airports relevant for SafeOPS and flight data either contains Approaches or Departures. The state machine also copes with flight data ending before the Cruise phase.

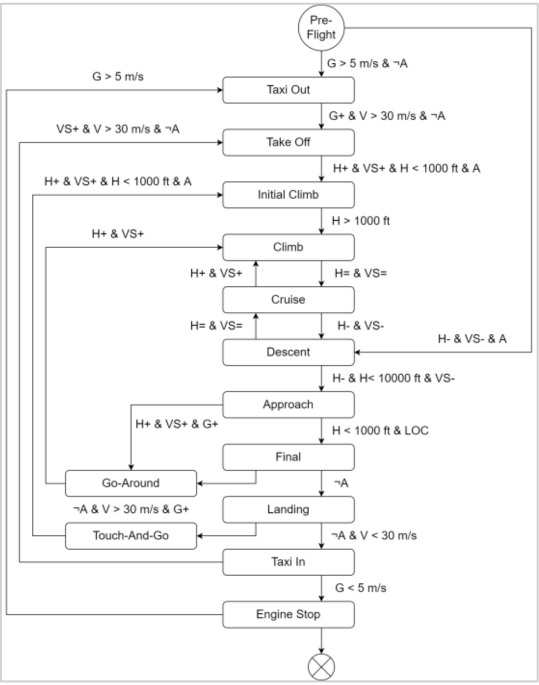


Figure 15: Flight phase detection logic for ADS-B data

As depicted in Figure 15, the Go-Around phase can be transitioned into from the Approach phase and the Final phase. The state transition is defined with the three parameters pressure altitude, vertical speed and ground speed. The pressure altitude and the ground speed need to show a positive trend, the vertical speed need so be positive. The relevant parameters for a Go-around detection are illustrated in Figure 16 below.

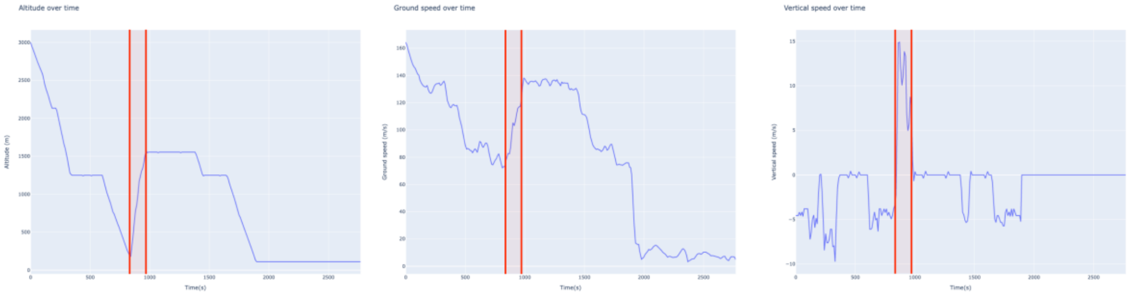


Figure 16: Go-around detection based on trend analysis (altitude, ground speed and vertical speed)

All three parameters in Figure 16 clearly show a Go-around was flown. However, not all Go-arounds can be detected with such clear behaviour of the data, especially if there is a Go-around early in the final approach with rather small changes in parameters. Therefore, thresholds categorizing the trend into positive and zero need to be chosen carefully. For the given flight, Figure 17 shows the missed approach procedure with the trajectory flown.

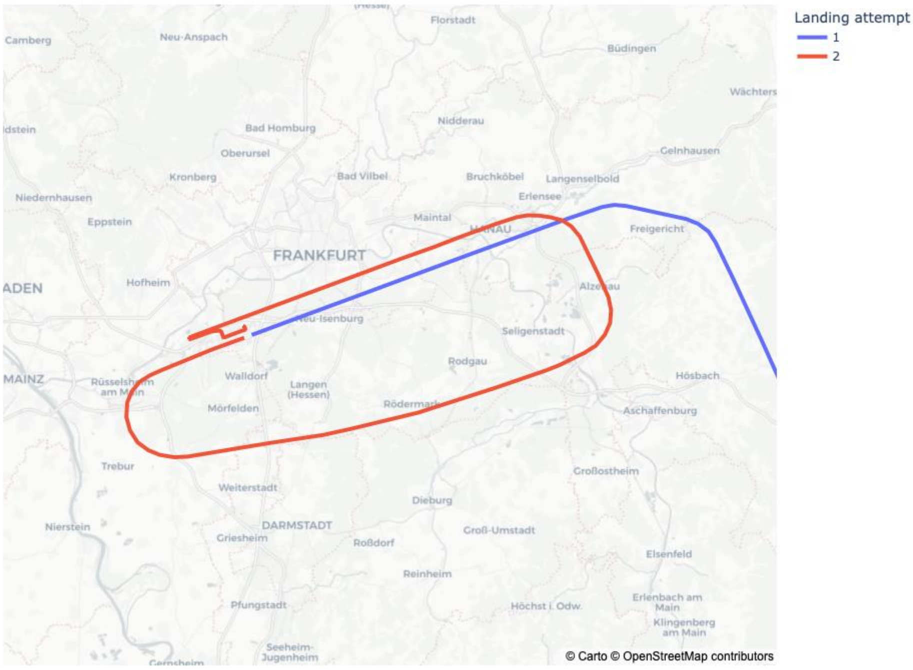


Figure 17: Visual confirmation of go-around procedure

5.2.3 Feature definition

In Table 2 you can find a summary of the set of features that have been defined to be used in the predictive models of the **ML_CS_01** case study. These features have been grouped into four feature types according to their nature. In addition, we can distinguish two feature sources. "**Available in data**" includes features that may have undergone some transformation (e.g. change of units, remove outliers) but are considered to be extracted directly from the data sources. "**Engineered feature**" includes those features that have been constructed through the transformation and combination of primary features.

Table 2: Description of features considered for the prediction model

Feature type	Feature name	Sampling	Source	Description
Flight information	Callsign	Static	Available in data	Flight callsign (e.g. DLH94U)
	ICAO24		Available in data	Aircraft unique 24-bit identifier (e.g. 3c4d6c)
	WTC		Engineered feature	Aircraft Wake Turbulence Category
	Approach attempt		Engineered feature	Flight approach attempt
	Hour		Available in data	Hour of the day
	Day		Available in data	Day of the week
	Week		Available in data	Week of the year
Weather data	Wind speed	Nearest available METAR report	Available in data	-
	Wind direction		Available in data	-
	Temperature		Available in data	-
	Visibility		Available in data	-
	Approach type		Engineered feature	IMC or VMC
	Dew point temperature		Available in data	Temperature below which the water will condense
	Ceiling height		Engineered feature	Based on the lowest clouds that cover more than half of the sky relative to the ground
Approach performance	Runway ID	Distance from the threshold	Engineered feature	Approached runway ID
	Specific energy level		Engineered feature	Aircraft specific energy level during the approach
	Ground speed		Available in data	Aircraft ground speed
	Vertical speed		Available in data	Descent vertical rate
	Vertical speed variance		Engineered feature	Descent vertical rate variance (window ±30s around time point)
	Track		Available in data	Aircraft track
	Track variance		Engineered feature	Aircraft track variance (window ±30s around time point)
	Altitude		Available in data	Aircraft altitude
	Track/Runway Bearing deviation		Engineered feature	Angular Deviation between aircraft track and runway bearing
	Centerline deviation		Engineered feature	Angular Deviation of aircraft position from runway centerline
Airport information	Total go-arounds	Time horizons (previous 10, 30 and 60 minutes)	Engineered feature	Total number of previous go-arounds at the airport
	Runway go-arounds		Engineered feature	Total number of previous go-arounds at the approaching runway
	Departures		Engineered feature	Total number of previous departures at the approaching runway
	Arrivals		Engineered feature	Total number of previous arrivals at the approaching runway
	Last departure time	Closest available flight information	Engineered feature	Time difference with previous departure at the approaching runway
	Last arrival time		Engineered feature	Time difference with previous approach at the approaching runway
	Last departure WTC		Engineered feature	WTC of the previous departure at the approaching runway
	Last arrival WTC		Engineered feature	WTC of the previous arrival at the approaching runway

For the feature type "**Approach Performance**" we want to briefly provide more details in the engineering of the features, especially for the detection of the runway ID and the specific energy. For the **runway ID** detection, it is crucial to choose the right time point in the timeseries. For the approach phases this is either the detected Touch Down or the last point before a Go-Around. If the airport is not provided, it is chosen from our database based on the minimum distance from the position at the chosen timepoint. For each runway at the airport, the difference between true track of the aircraft and true bearing of the runway as well as the angular difference between the true bearing of the runway and the true track between the aircraft position and the threshold position are computed. The runway with the minimum sum of both differences is chosen. Both differences are also computed for the defined sampling for the features "**Track/Runway Bearing deviation**" and "**Centerline deviation**".

The "**Specific energy level**" is the sum of the specific potential energy ($h_{AAL} \cdot g$) and the specific kinetic energy ($0.5V_{GS}^2$). This eliminates the need to have the mass available. However, the mass still has an influence on the ground speed due to higher approach speeds for higher landing weights.

5.3 EDA (Exploratory Data Analysis)

In this section we will proceed to carry out the **EDA (Exploratory Data Analysis)** of the defined features. Although this will focus on case study **ML_CS_01**, which is the one that will be explored and presented in this deliverable. It is considered that the conclusions obtained will be of equal validity and help for the rest of the case studies. EDA involves using **summary statistics** and **data visualization** techniques to analyse and identify possible trends in data sets. The primary objective of the EDA phase is to help the analyst gain **useful insights** into a data set.

After processing the selected data sources through the developed **automated data processing pipeline** a clean dataset is obtained containing the identified arrivals per airport, with the corresponding go-around labelling as well as the previously defined set of features. Due to the variant quality mainly of the **ADS-B data source** and the total coverage some arrivals are discarded during processing due to errors in the data such as excessive number of outliers, on ground segment missing or too few data points during the approach phase. Although this loss of information is less than desirable, the number of traces with errors is usually minimal and a sufficiently high volume of examples is maintained to proceed with exploration and the predictive modelling phase. For example, although the difference in traffic between **EDDF** and **EDDM** is not very substantial, some difference in the amount of data recovered can be seen mainly due to the worse coverage in EDDM, but still the volume is considered sufficient to go ahead with its use. Table 3 shows the total number of departures, approaches and go-arounds identified in the study period.

Table 3: Total number of departures, approaches and go-arounds

Airport	Year	Number of days	Number of departures	Number of approaches	Number of go-arounds
EDDF	2018	221	121852	125317	422
	2019	365	214541	216358	650
	2020	60	31468	29180	246
Totals		**646**	**367861**	**370855**	**1318**

Airport	Year	Number of days	Number of departures	Number of approaches	Number of go-arounds
EDDM	2018	221	96361	86059	289
	2019	365	146919	118588	399
	2020	60	21138	14841	85
Totals		**646**	**264418**	**219488**	**773**

Tables 4 and 5 show the total number of approaches and go-arounds broken down by runway at the airports EDDF and EDDM airports. In the case of EDDF, we can see that runway 07C presents considerably lower usage than the other runways, which was expected, but a considerable high go-around ratio. This may be due to the fact that with three parallel runways so close together it is sometimes difficult to correctly label the approached runway. Although aware of this possible error, as there are many other features being used in the predictive model, and given that the model must be robust to possible errors, it has been decided to leave the data as presented.

Table 4: EDDF - Total number of approaches and go-arounds per runway

EDDF	07L	07C	07R	25L	25C	25R
Number of approaches	68589	4441	62132	93075	42994	99624
Number of go-arounds	237	162	185	247	82	405

Table 5: EDDM - Total number of arrivals and go-arounds per runway

EDDF	08L	08R	26L	26R
Number of approaches	39704	49080	70431	60273
Number of go-arounds	147	179	223	224

For the case study **ML_CS_01**, and based on the feedback obtained by the ATCOs, it was decided to sample the data based on the distance to the runway's threshold. Four possible prediction points were established (**2NM, 4NM, 6NM and 8NM**). Based on the feedback from the ATCOs, the most interesting for them would be the point around **4 NM** which will be used for the models proposed in this case study. Closer to this point (e.g. 2NM), although probably more accurate predictions could be obtained, the reaction time is considered to be too small and of little use. Although it might be of more interest to be able to predict beyond 8NM, achieving an acceptable **prediction beyond 8NM is considered improbable** based on the existing literature (Dhief, Imen, et al. 2021) stating the complexity of the go-around and the influence of a large number of factors that may not be reflected in the data (e.g. pilots mental framework). Based on the sampling by distance for the **EDDF** we obtain that approx. **98% of go-arounds are initiated after the 2NM** mark and only **1% start before the 8NM mark**. Similarly, for **EDDM**, **98% of go-arounds are initiated after 2NM** mark and only **1% start before the 8NM mark**. Again, based on the literature (Proud, Simon Richard 2020), these results are similar to those experienced at other airports.

Co-funded by the European Union

The following is an exploration and analysis of the four main features types (Flight information, Weather data, Approach performance and Airport information). The objective will be to better understand the r**elationship between the features and the target variable** (Go-arounds).

5.3.1 Flight information

We start the analysis of the features of the "Flight information" group by looking at how the go-arounds are distributed according to **the day of the week**. As can be seen in Figure 18 for both EDDM and EDDF, the distribution of **go-arounds per 1000 approaches (GA_per_1000)** is quite continuous for the different days, only suffering some decreases on Saturdays or Sundays (days 6 and 7). This, a priori, indicates that this feature alone should be considered as a weak feature as it does not seem to provide much correlation to the target variable (go-arounds).

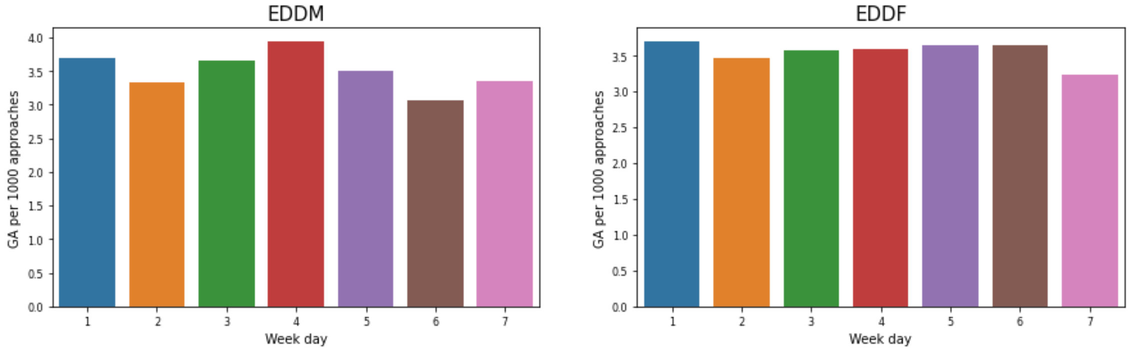


Figure 18: Week day vs Go-arounds per 1000 approaches - (EDDM and EDDF)

Continuing with the analysis we look at the relationship between the week of the year and GA_per_1000, see Figure 19. Contrary to the previous case, here there is not a uniform distribution rather we can see how the number of GA_per_1000 varies significantly in some cases depending on the week of the year. In both airports we can see how the first weeks of the year present a higher number of GA_per_1000. This may be due to the fact that they coincide with the weeks of the year with the most adverse weather conditions, see Figure 20. Also noteworthy in the case of EDDM is an unusual peak that occurs in week 30. We could consider that this feature as a possible strong feature as it has a higher correlation than the previous one.

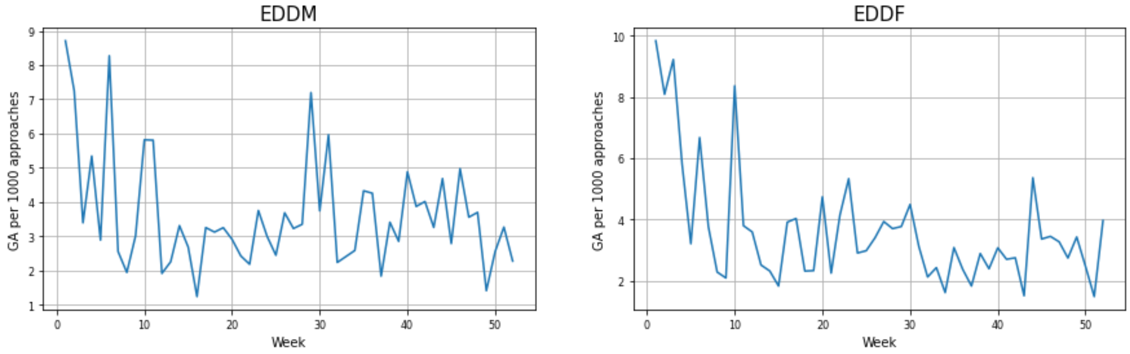


Figure 19: Week vs Go-arounds per 1000 approaches - (EDDM and EDDF)

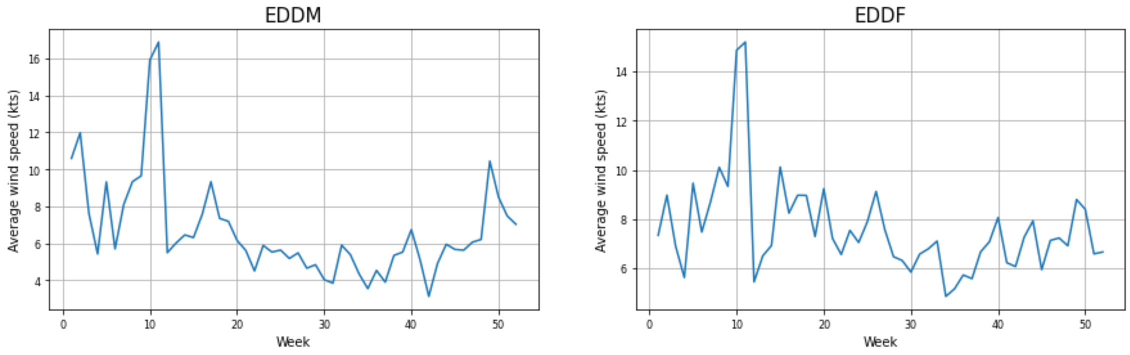


Figure 20: Week vs Average wind speed (kts) - (EDDM and EDDF)

Analysing now the **Wake Turbulence Category (WTC)** of the aircraft, we can see that aircraft of type Light (L) have the highest average GA_per_1000, see Figure 21. This may be due to the fact that these aircraft are more susceptible to weather conditions as well as being more affected by the surrounding traffic. Although it is worth pointing out that the total number of L-WTC-category, both for EDDM and EDDF, is extremely low (<1%). For both airports the **majority WTC category is M being for EDDM 82%** of all approaches and **72% for EDDF**. Also, highlight the L/M category, which in both cases have less than 10 approaches in the whole period studied. In general, these are usually special category aircraft and could be used as a filter to eliminate them from the dataset to be used in the training of the models. Finally, mention should be made of the category None. It has not been possible in all cases to derive the WTC from the information present in the data. Those aircraft that could not be obtained have been identified as None. It has been decided to keep them in the dataset because, as mentioned above, the model must be robust to operate with possible errors or in cases where the approaching aircraft is new and for which not all the information is available.

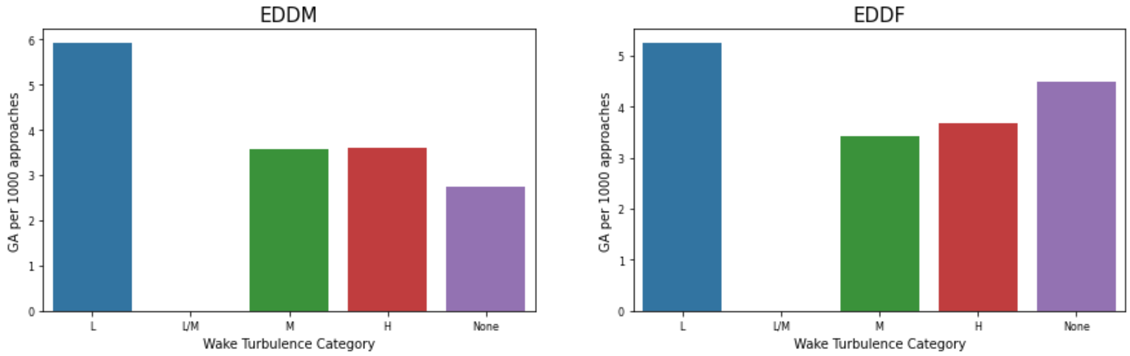


Figure 21: Wake Turbulence Category vs Go-around per 1000 approaches - (EDDM and EDDF)

Finally, to finish with the "Flight information" feature group, we look at the **relationship between the Callsign and go-around events**. As usually aircrafts from the same airline and flying a specific route get assigned the same callsign. EDDM present 4931 unique callsigns while EDDF 5542. Because of this, the information provided by the callsign goes far beyond the code itself. Through it, information can be inferred that is otherwise difficult to obtain. For example, the familiarity of a pilot with the airport, the total flight time or even peculiarities in the performance of pilots of a certain airline due to their training. In Figure 22 all callsigns are represented on the x-axis and on the y-axis the ratio of go-around per approach. Although it is difficult to see anything very clearly, the most important thing is that

Co-funded by the European Union

different spikes can be distinguished in the graph indicating that certain callsigns have a higher level of go-arounds than others. Although this may reinforce the belief in the usefulness of this feature in the prediction model due to the fact that it is a categorical variable that will have to be encoded and the varying number of different samples it presents, we will most likely have to drop it from the final training dataset. This is because categorical features with high cardinality, such as the callsign feature, can introduce noise in the data and thus make it harder to train a model and make it to perform poorly.

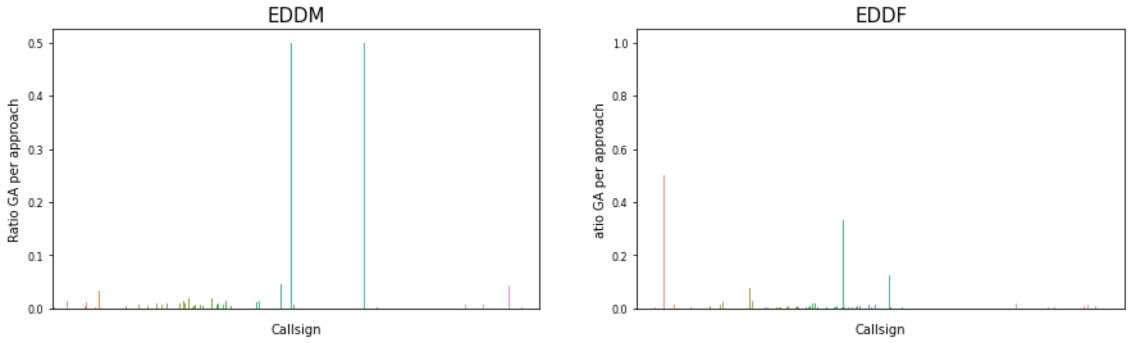


Figure 22: Callsign vs Ratio of go-around per approach - (EDDM and EDDF)

5.3.2 Weather data

Analysis performed on 2019 data from the EDDF station.

Table 6: Percentage of NaN's in METAR data

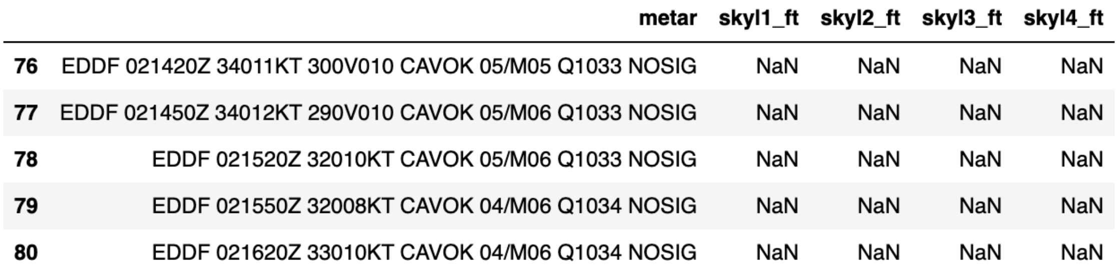
Variable	% of NaNs
station	0.000000
valid	0.000000
temperature_C	0.000000
dewpoint_C	0.000000
relHumidity_per	0.000000
winddirection_deg	12.812018
windspeed_kts	0.000000
precipitation_mm	0.000000
QNH_hPa	0.000000
mslp	100.000000
visibility_m	0.000000
gust_kts	97.880848
skyc1	41.132119
skyc2	69.577883
skyc3	91.557663
skyc4	99.868624

Variable	% of NaNs
skyl1_ft	42.245959
skyl2_ft	69.577883
skyl3_ft	91.557663
skyl4_ft	99.868624
wxcodes	80.362141
ice_accretion_1hr	100.000000
ice_accretion_3hr	100.000000
ice_accretion_6hr	100.000000
peak_wind_gust_kts	100.000000
peak_wind_drct_deg	100.000000
peak_wind_time	100.000000
feltTemperature_C	0.000000
metar	0.000000
snowdepth	100.000000
date	0.000000
utc	0.000000

Regarding the sky cover features which are indeed of interest for the go-arounds there are important remarks. There are eight sky cover features:

- Four regarding the sky coverage (skyc1, skyc2, skyc3, skyc4), these variables indicate the amount of sky covered following an octal metric, going from few clouds (FEW) if 1-2/8 of the sky is covered, scattered (SCT) with 3-4/8 of coverage, broken (BKN) which indicated 5-7/8 of sky covered and overcast (OVC), implying a total coverage of the sky
- Four regarding the height at which the sky coverage is measured (skyl1_ft, skyl2_ft, skyl3_ft, skyl4_ft).

A first measure if performed for sky level 1, if the measure indicates that the sky is clear (few or scattered), a second measure is performed for sky level 2, and so on until reaching a sky level at which the coverage is broken or overcast. These measures could be used to obtain the ceiling height, which is the height at which there is still visibility, however, as previously analysed the sky level 1 variables present around 42% of missing values, thus a very high amount of missing information, however, when taking a deeper look on what message accompanies all measures having missing values for all sky cover variables, we can see that it appears along with the word CAVOK (Ceiling and Visibility OK), which means that there are no clouds below 5000 feet above aerodrome level (AAL). This information has been contrasted with the METAR service provider.



	metar	skyl1_ft	skyl2_ft	skyl3_ft	skyl4_ft
76	EDDF 021420Z 34011KT 300V010 CAVOK 05/M05 Q1033 NOSIG	NaN	NaN	NaN	NaN
77	EDDF 021450Z 34012KT 290V010 CAVOK 05/M06 Q1033 NOSIG	NaN	NaN	NaN	NaN
78	EDDF 021520Z 32010KT CAVOK 05/M06 Q1033 NOSIG	NaN	NaN	NaN	NaN
79	EDDF 021550Z 32008KT CAVOK 04/M06 Q1034 NOSIG	NaN	NaN	NaN	NaN
80	EDDF 021620Z 33010KT CAVOK 04/M06 Q1034 NOSIG	NaN	NaN	NaN	NaN

Figure 23: METAR sky coverage

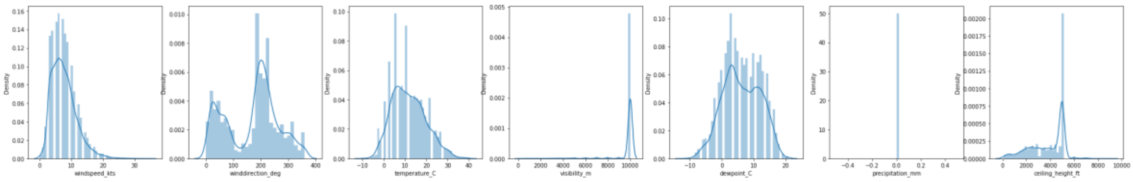


Figure 24: Distribution of the proposed features: wind speed, wind direction, temperature, visibility, dewPoint and precipitation

It can be appreciated how precipitation is always 0, so it shall not be used as features. On the other hand visibility has a poor distribution as it is measured with very low sensitivity, yet it may be a valuable feature, as, even if most of the times it indicates good visibility, it may be linked with go-around.

Also, taking a look into a pair plot of the METAR data we can see how all the selected five features are correlated to each other. It can be appreciated how there exist no linear correlations between any of them, which justifies the use of them all in the machine learning problem.

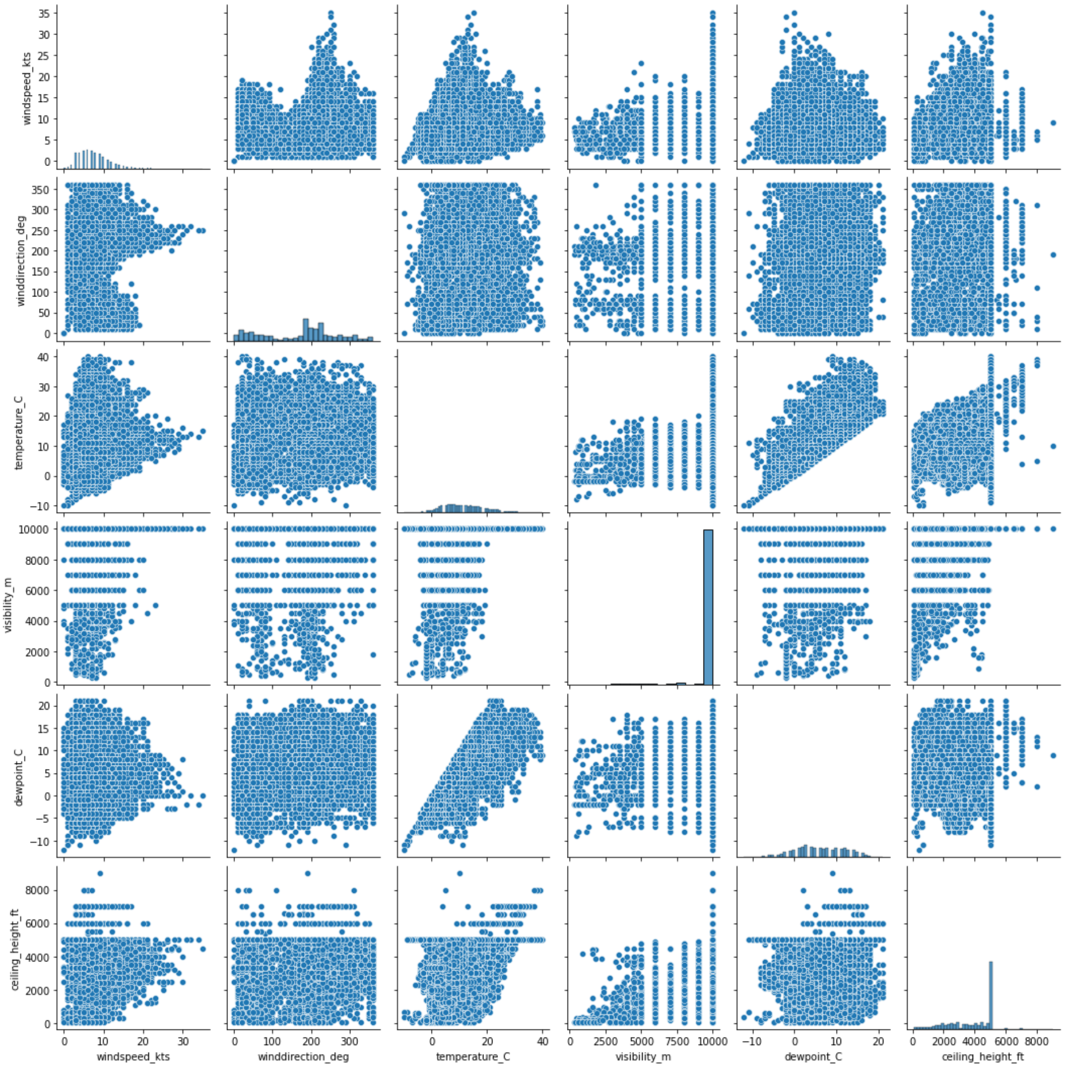


Figure 25: Pairplot of the proposed features: wind speed, wind direction, temperature, visibility, dewPoint and precipitation

5.3.3 Approach performance

Next, we explore the "**Approach performance**" type features. These refer to the performance of the aircraft in the different parts of the approach. It is intended to allow the model to understand possible variations in the performance of an aircraft when it finally performs a go-around compared to those which finally land.

We start by looking at the **ground speed (m/s)** distribution at 4 NM, see Figure 26. Before we start with the analysis, one important thing to point out is that these figures represent the distribution density. When looking at the figures, always keep in mind that this is done relative to the total number of examples in each class, emphasising again the disparity between the normal and go-around approaches (1:1000 approx.). It can be seen how the distribution of ground speed for those approaches that do not perform a go-around (blue curve) is very narrow, being centred around 70 m/s. In the case of the go-around approaches, although the main peak is also centred around this value, we can see how the distribution is much wider, with another peak for higher ground speed values. This indicates that although in general the majority of go-around approaches have similar ground speed to non-go-around approaches, a significant number tend to have higher ground speeds than non-go-around aircraft at the 4NM mark. This indicates that this variable could be useful for the model when identifying some of these go-arounds. Representing these distributions allows us to identify possible outliers in the data (in this case unusually high or low ground speeds) as a way to validate the processing of the data as well as informing us before the training phase of the model that there may be some approaches that need to be removed.

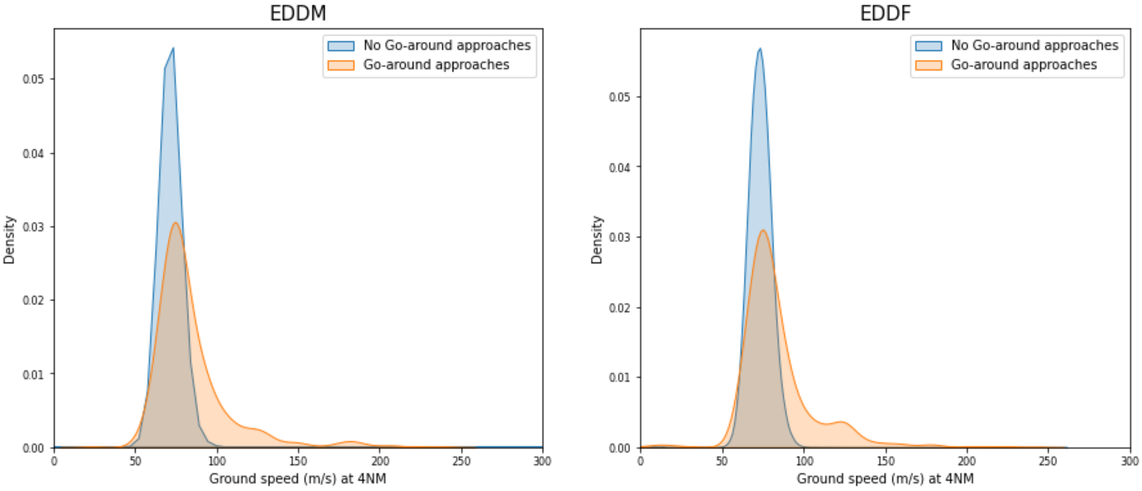


Figure 26: Ground speed (m/s) at 4NM - (EDDM and EDDF)

Turning now to the **vertical velocity (m/s)**, see Figure 27. Similar to the previous case we can see how those approaches without go-around have a very narrow distribution with most of them presenting a negative vertical velocity around 5 m/s. In the case of the approaches with go-around we can see that the distribution is again wider, in some cases even presenting positive values of vertical velocity or higher negative values than the approaches without go around. This indicates that this variable could be useful for the model when identifying some of these go-arounds.

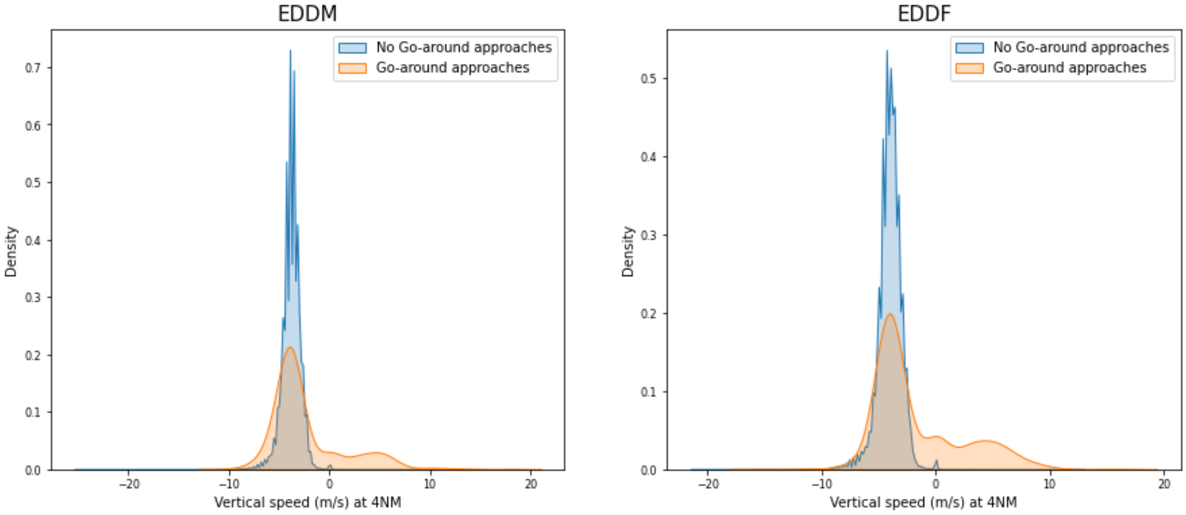


Figure 27: Vertical speed (m/s) at 4NM - (EDDM and EDDF)

In the case of the **altitude (m),** again very similar characteristics to the previous cases can be seen, see Figure 28. We can see how the approaches without go-around have a very narrow distribution with most of them presenting an altitude below 500m. In the case of the go-around approaches we can see that the distribution is again wider, centred like the non-go-around approaches but with a higher proportion of approaches with higher to lower altitude values. This indicates that this variable could be useful for the model in identifying some of these approaches.

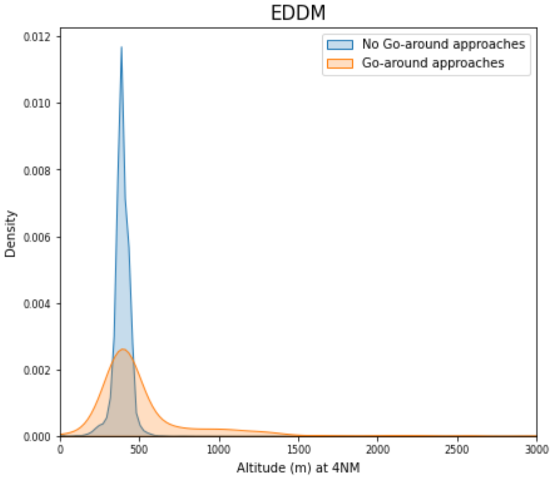


Figure 28: Altitude (m) at 4NM - (EDDM and EDDF)

Finally, we will look at the differences in the distribution of the **Centreline deviation (radians)** between the go-around and non-go-around approaches, see Figure 29. This is where we can see a major difference between the two distributions. As in the other cases analysed, the distribution of the non-go-around approaches presents a very narrow distribution where the flights show hardly any deviation. The only thing worth mentioning is two small spikes in both EDDM and EDDF. Although there is no clear initial explanation, in the case of EDDF this may be due to those approaches that perform a "swing" landing. In the case of EDDM it may be due to a similar situation. In the case of go-around approaches we can see that the distribution in this case is relatively flat. This indicates that for this feature the difference between the cases with go-around and without go-around is larger making it one of the most predictive possible features.

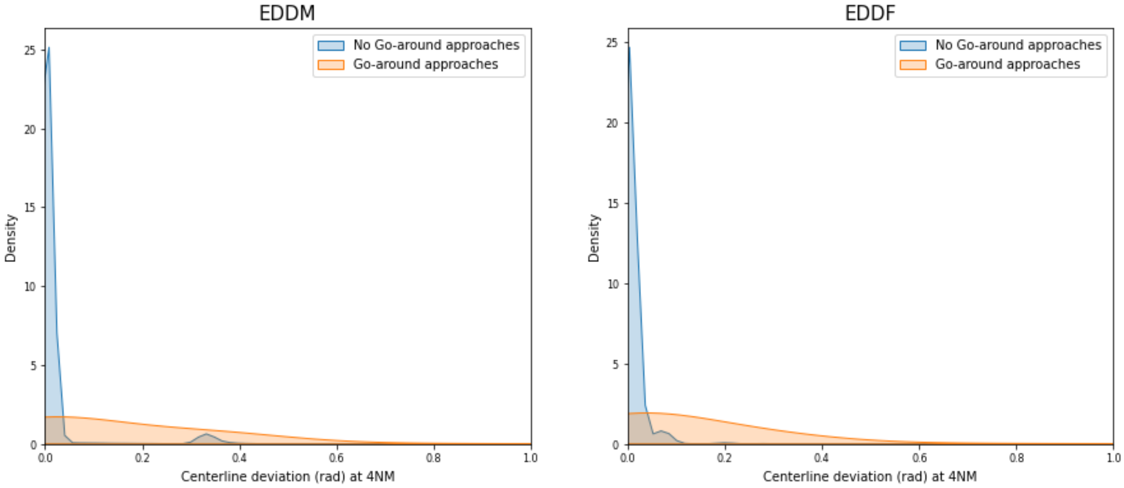


Figure 29: Centerline deviation (rad) at 4NM - (EDDM and EDDF)

5.3.4 Airport information

Finishing the EDA with the analysis of "**Airport information**" type features, we begin by looking at the relationship between the go-around prior to an approach. Looking at the relationship of go-around ratio (GA_per_1000) to previous go-arounds on the runway being approached we see that in the cases where there has been a previous go-around in the previous 10 minutes the ratio of go-arounds increases, see Figure 30. Although it should be noted that the vast majority of the approaches (both go-around and non-go-around) do not have a go-around in the previous 10 minutes.

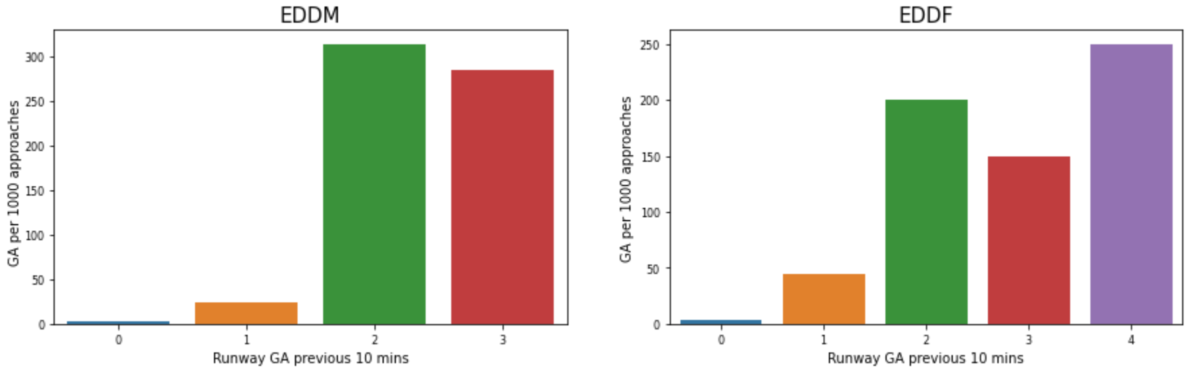


Figure 30: Number of previous Go-around in approached runway (10 mins) vs Go-around per 1000 approaches - (EDDM and EDDF)

Next, we look at the relationship of go-around ratio and number of arrivals on the same runway. In Figure 31 we can see how the highest go-around ratio occurs when in the previous 10 mins there has been no approach on the same runway. The ratio decreases as the number of arrivals increases but maintains a nearly flat distribution. This suggests that this feature may have little predictive power.

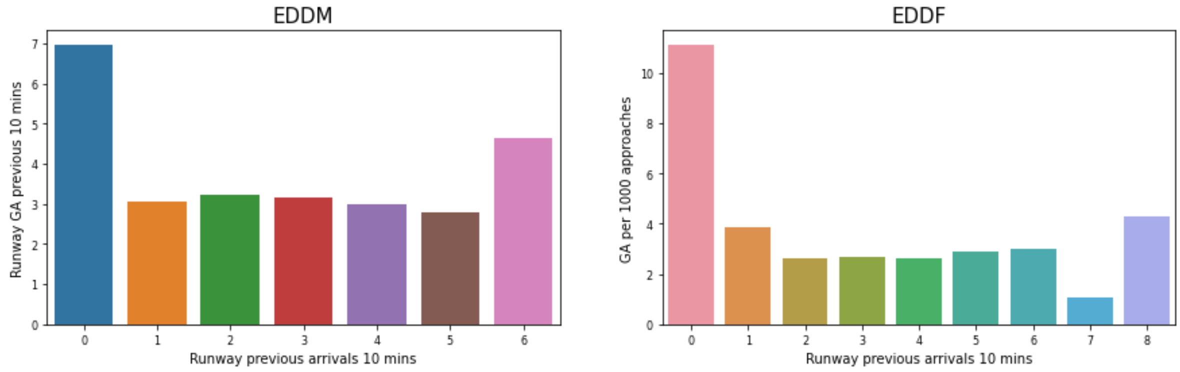


Figure 31: Number of previous arrivals in approached runway (10 mins) vs Go-around per 1000 approaches - (EDDM and EDDF)

Finally, in the tables below we see the ratio of aircraft WTC and previous arrival WTC at both airports. We can see how in both cases, due to the fact that most of the approaches are M or H aircraft, the largest overlap occurs between these two categories. Although it has been shown above that the WTC of the aircraft may have some influence on the prediction due to the small variation in overlap, it is expected that this variable will only have minor influence on the prediction.

Table 7: EDDM Aircraft WTC vs Previous arrival WTC

EDDM		Previous arrival WTC			
		L	L/M	M	H
Aircraft WTC	L	0	0	2	0
	L/M	0	0	0	0
	M	2	0	514	42
	H	0	0	39	17
	None	1	0	37	1

Table 8: EDDF Aircraft WTC vs Previous arrival WTC

EDDF		Previous arrival WTC			
		L	L/M	M	H
Aircraft WTC	L	0	0	0	1
	L/M	0	0	1	0
	M	0	0	613	134
	H	0	0	155	105
	None	0	0	50	11

5.4 Predictive results

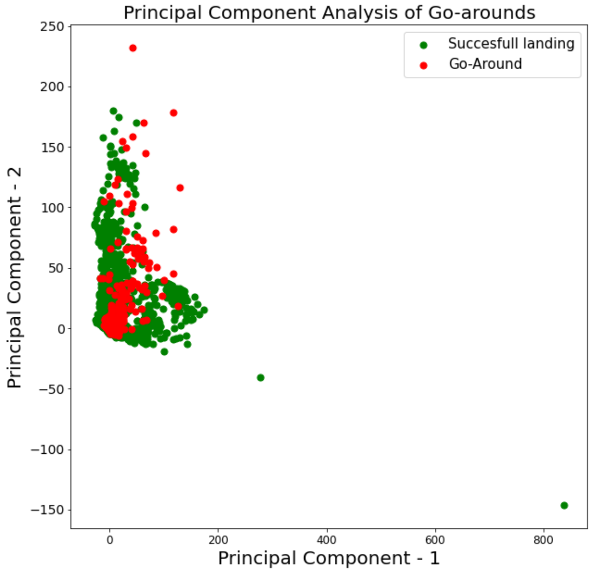
The prediction point for the first caste study (**ML_CS_01**) was set, as previously discussed, at **4NM from the runway threshold**. This means that, although features were computed for points closer to the threshold (e.g. 2NM or 3NM), these were not included in the final training or testing datasets. This is necessary in order to ensure the validity of the model results by avoiding the introduction of information in the training or validation phase that would not be available at the time of prediction in a real operational scenario.

5.4.1 Data transformations prior to training

Before training the models, some final filtering and transformations in the established dataset needs to be carried out. For example, thanks to the EDA phase, it has been found that after processing the trajectories, a small number of flights still show **outliers** such as excessively high or low velocities. Prior to training, filters can be applied to erase these approximations completely so so these erroneous points do not influence the model during the predictions due to these erroneous points. The only thing to be careful when applying these filters is when defining what is a possible error of an outlier. Although an outlier may at first glance also appear to be an erroneous value, care must be taken as it may not be an erroneous value and may only be a special situation from which the model must also be able to learn. That is why the applied filters should be a bit lax and let some seemingly abnormal outlier values pass. Additional techniques can be applied to reduce the influence of these so-called outliers, as we will see. For this case study (ML_CS_01), aircraft with extremely unusual ground speed, vertical speed or altitude values at the 4 nm mark will be eliminated and not used for training and testing.

In addition, columns with unwanted features for the model such as airport (models are trained for a specific airport) or full date were removed from the final dataset. This left a final dataset with a total of **149 features** for modelling. Continuing with the pre-training transformations, the numerical features were normalised. Many machine learning algorithms do not work well if the orders of magnitude between features vary considerably. **Standard scaling** was used which consists of subtracting the mean of the feature from each piece of data and dividing by the standard deviation. This method is more robust to outliers than variable scaling which compresses the input data within limits, usually the maximum and minimum. Similar to the numerical features, some models only support numerical features and thus encoding of categorical variables is necessary. There are various ways of encoding categorical features (e.g. ordinal encoding or one-hot encoding) and there is no optimal way and all have advantages and disadvantages. In our case we have opted to use the **Target encoding**. Target encoding is the process of replacing a categorical value with the mean of the target variable (Micci-Barreca, D. 2001). It is a simple and fast coding method that also has the advantage that it does not increase the dimensionality of the data set. The main disadvantage is that it depends on the target distribution, which means that the target coding requires careful validation, as it may be prone to overfitting. Also, after all the processing pipeline due to various small errors some examples may present empty values in some features (NaN). There are several techniques to impute value, based on the median value or most frequent value, or if the number of examples is sufficiently low, they can be directly removed if they do not affect the overall distribution of examples.

In Figure 32, 33 and 34, we present a **Principal Components Analysis (PCA)** visualisation. PCA is a dimensionality reduction technique that allows us to visualize all our features within a 2D plot (Jolliffe, I. T. 2002). The visualisation gives us two initial insights into the problem. First, it shows how classes cannot be linearly separable. Therefore, linear discriminant classifiers in principle do not seem to be good enough to solve the classification problem. Secondly, we can observe that the two classes overlap. Because of this distribution and taking into account the strong imbalance of the dataset, one can start to assume that identifying and isolating "class 1 = Go-arounds" will be a very complex machine learning challenge.



Co-funded by the European Union

Figure 32: EDDM - Principal Component Analysis (PCA) of Go-arounds

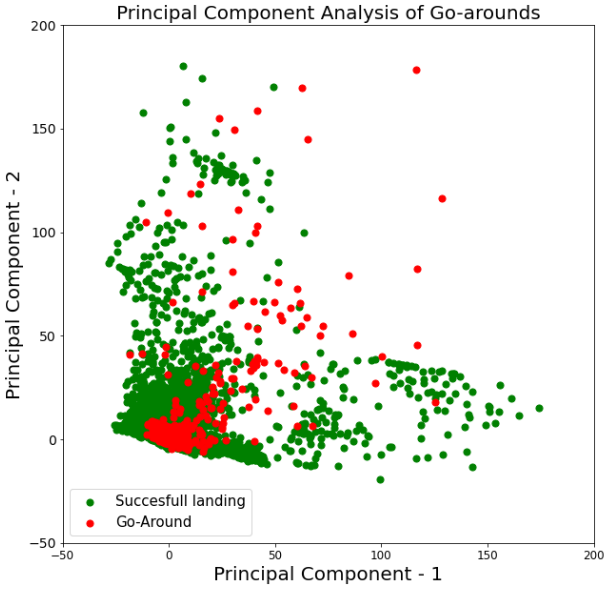


Figure 33: EDDM - Principal Component Analysis (PCA) of Go-arounds - (zoomed-in)

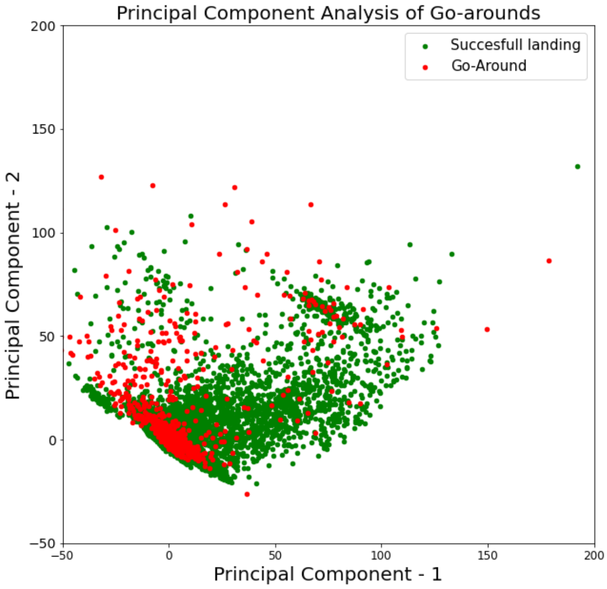


Figure 34: EDDF - Principal Component Analysis (PCA) of Go-arounds

Finally, and as we have already seen and discussed, we are dealing with a very **imbalanced classification problem**. In machine learning terms, a dataset is considered to be heavily imbalanced when the ratio between majority and minority class is greater than 1:100 (Krawczyk, B. 2016). In general, some classification models are designed to operate optimally on problems where the dataset is balanced or only slightly imbalanced. In these cases sometimes this requires the application of **resampling techniques**. Resampling is nothing more than a forced way to change the original ratio between the two classes. This resampling can be either **undersampling** (where the number of examples of the majority class is reduced) or **oversampling** (increasing the number of examples of the minority class). Resampling is not always a solution, as it will ultimately depend on the distribution of the data. Undersampling often introduces the problem that information is lost and thus may ultimately make it difficult for the model to correctly interpret the majority and minority class (underfitting). On the other hand, oversampling not only increases the learning time by increasing the number of samples, but also increases the likelihood of occurring overfitting. Although resampling techniques have been applied with varying degrees of success to the go-around problem (Dhief, Imen, et al. 2021), our initial tests show that they do not provide an advantage to all types of models. As we have seen in the PCA it is difficult to discern normal approximations from go-arounds and therefore using undersampling or oversampling in some cases does not improve the problem. As can be seen in Figure 35, where the **SMOTE** technique has been used to generate synthetic samples (Chawla, Nitesh V., et al 2002). As the overlap is very large, these new samples can increase the noise of the data and cause underfitting. In this case study, after testing various oversampling ratios, it has been found that the best results are obtained when the number of minority class values (go-arounds) is increased to around **5% of the number of majority class values**. In addition, **Penalised models** (e.g. ensemble models) can be used. Penalised classification models work by imposing an additional cost for misclassification errors in the minority class during training. These penalties can bias the model to pay more attention to this class. During the **benchmarking phase** several of the above techniques were used to combat data imbalance, finally after checking which ones contribute to improved performance and which ones do not, the results presented are those of the best performing combinations.

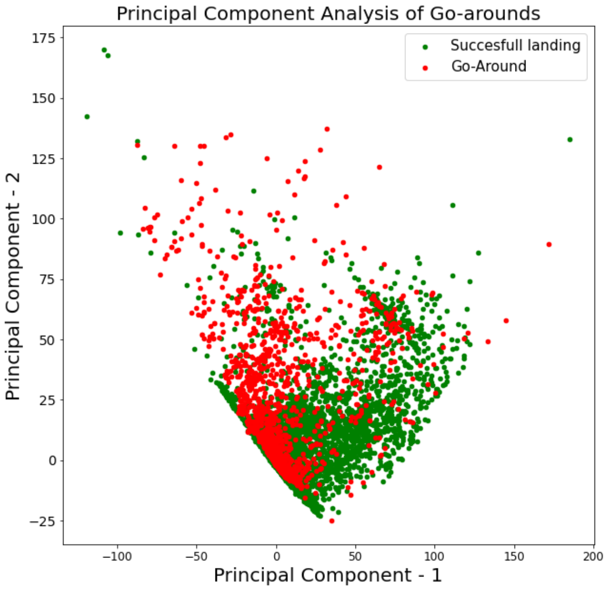


Figure 35: EDDF with SMOTE oversampling - Principal Component Analysis (PCA) of Go-arounds

5.4.2 Model testing and evaluation

In the section below, the results of the **benchmark study** of the nine most common models used in classification problems are described. The models were trained with some minor modification of the hyperparameters, as the aim is to obtain an initial result of the most promising model and provide a baseline for comparison with the model finally selected. Three different types of algorithms were proposed for this study:

- **Linear algorithms**: Naive Bayes and Logistic Regression.
- **Non-linear algorithms**: Decision Tree, K-Nearest Neighbours and Multi-layer Perceptron (artificial neural network).

Co-funded by the European Union

- **Ensemble Models**: Random forest, Adaptive boosting and Gradient boosting.

Although there are a multitude of other models and even more powerful or sophisticated ones, this selection was mainly based on their **interpretability** and **explainability** levels of the models. See Figure 36. It can be seen that although ensemble models, for example, are of low interpretability, due to their good performance and the existence of new techniques to understand how they work, they have remained in the analysis.

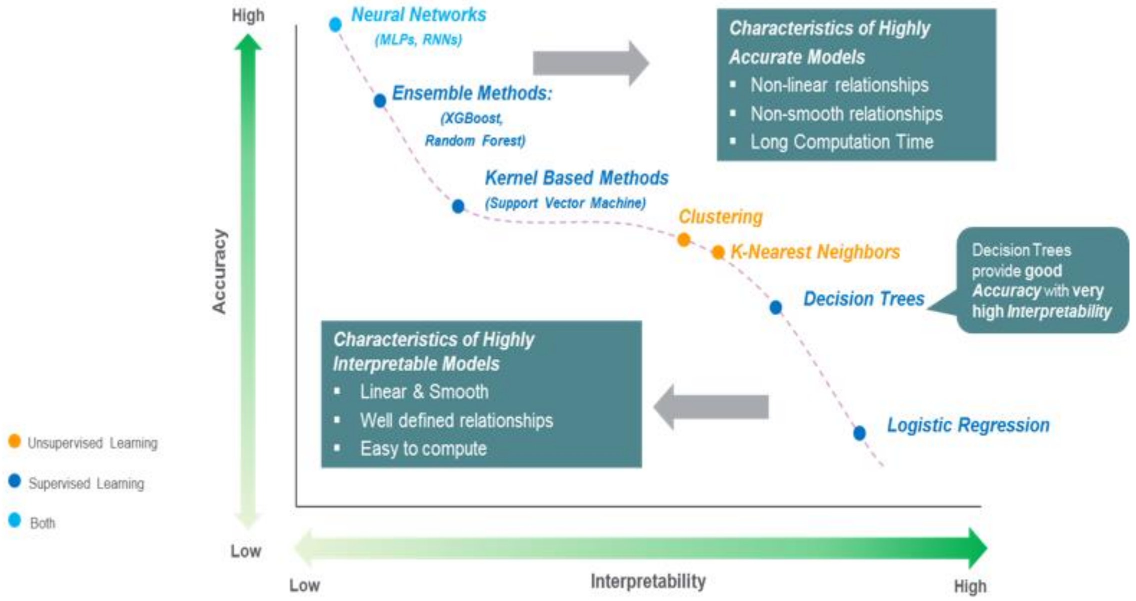


Figure 36: Interpretability vs Accuracy (Sharayu Rane)

The different models were trained and validated using the **k-fold cross-validation** technique. This process allows us to assess the performance of a model by splitting the data into several folds and dividing them into training and validation subsets. By applying k-fold cross-validation, we assess that our model fits correctly and we try to minimize **overfitting**. For each fold a new model is trained using the training subset and validated with the validation subset, obtaining different evaluation metrics for each model. A detailed description of cross-validation can be found in the deliverable D2.1 section 3.2.1 (SafeOPS D2.1). It is important to ensure that information from the validation dataset is not used during the training and cross-validation process mentioned. When this happens, it is known as "**data leakage**" and can cause overly optimistic results and therefore invalid predictive models to be created. Tables 9 and 10 shows the results by airport, obtained for each model with its corresponding **precision**, **recall**, **F1-score** and **ROC-AUC**. A detailed description of these metrics can be found in the deliverable D2.1 section 3.2.1 (SafeOPS D2.1).

Table 9: EDDM model benchmark study results

EDDM					
Model	Go-around	Precision	Recall	F1-score	AUC (ROC)
Naive Bayes	True	0.06	0.29	0.10	0.76
	False	0.99	0.98	0.99	
Logistic regression	True	0.67	0.15	0.25	0.80

EDDM					
	False	0.99	0.99	0.99	
K-Nearest Neighbours	True	0.84	0.12	0.21	0.64
	False	0.99	0.99	0.99	
Decision Tree	True	0.17	0.24	0.20	0.62
	False	0.99	0.99	0.99	
Multi-layer Perceptron	True	0.40	0.22	0.28	0.72
	False	0.99	0.99	0.99	
Random Forest	True	0.77	0.21	0.32	0.80
	False	0.99	0.99	0.99	
Adaptive Boosting	True	0.70	0.18	0.29	0.82
	False	0.99	0.99	0.99	
Gradient Boosting (XGBoost)	True	0.80	0.21	0.34	0.85
	False	0.99	0.99	0.99	
Gradient Boosting (LightGBM)	True	0.77	0.22	0.34	0.88
	False	0.99	0.99	0.99	

Table 10: EDDF model benchmark study results

EDDF					
Model	Go-around	Precision	Recall	F1-score	AUC (ROC)
Naive Bayes	True	0.07	0.42	0.13	0.84
	False	0.99	0.98	0.99	
Logistic regression	True	0.74	0.22	0.34	0.82
	False	0.99	0.99	0.99	
K-Nearest Neighbours	True	0.89	0.34	0.49	0.73
	False	0.99	0.99	0.99	
Decision Tree	True	0.32	0.41	0.36	0.71
	False	0.99	0.99	0.99	
Multi-layer Perceptron	True	0.60	0.38	0.47	0.78
	False	0.99	0.99	0.99	
Random Forest	True	0.84	0.44	0.58	0.87
	False	0.99	0.99	0.99	
Adaptive Boosting	True	0.62	0.23	0.34	0.88
	False	0.99	0.99	0.99	

EDDF					
Gradient Boosting *(XGBoost)*	True	0.90	0.40	0.57	0.89
	False	0.99	0.99	0.99	
Gradient Boosting *(LightGBM)*	True	0.84	0.43	0.57	0.90
	False	0.99	0.99	0.99	

As can be seen in the tables above, for both EDDM and EDDF the best performing models are the "**Ensemble models**". These models combine the prediction of multiple simpler "weaker" models (e.g. decision trees) to improve their overall performance. The main causes of error in machine learning models are due to noise, bias and variance. Ensemble models are especially good at minimising these factors because they are designed to improve the stability and accuracy of machine learning algorithms. There are two types of combinatorial models: **Bagging** and **Boosting**. Bagging models are based on creating random samples from the training data set and building a different model for each sample. The results of these models are combined using averaging or majority techniques. Boosting models use a sequential technique in which a model is first trained on the entire data set and subsequent models are built by fitting the residuals from the first model. In this way, observations that were erroneously predicted by the previous model are given more weight.

Of the Ensemble models tested, the one with the most complete performance were the **Gradient Boosting models**. There are several implementations of this type of model that are very similar, but may have slight variations. The main implementations are from the libraries: **Scikit-learn, XGBoost** and **LightGBM**. In this study, all three libraries were tested and similar performance results were obtained. It was finally decided to use the LightGBM implementation because, although the results with the other libraries were equivalent, it had much shorter training times than the others (up to 10 times less). It should also be noted that the LightGBM model benefited from applying **SMOTE** techniques but not from **penalty techniques**. This is because by applying different penalty costs the model overcorrected and increased the total number of go-arounds detected (recall) but at the cost of drastically reducing precision. Due to the specific nature of this problem where the cost of false warnings is very high it was decided that the performance that best suited the objectives of the project is where precision will be maximised by trying to obtain an acceptable recall.

We decided to try to optimise the go-around prediction capability of the LightGBM model as much as possible for both airports by optimizing their **hyperparameters**. The hyperparameters of a model are different adjustable parameters that must be chosen before training a model and that govern the training process itself. Hyperparameter tuning is often a delicate process because the performance of the model maybe highly dependent on the selected values. There are different techniques for hyperparameter tuning: **grid search**, **random search** or **Bayesian optimisation**. Although none of the technique guarantees optimal results, we decided to use the last option (Bayesian optimisation) through the **Optuna** library. Bayesian hyperparameter optimization works by building a probability model of the objective function and use it to select the most promising hyperparameters to evaluate the true objective function.

For the hyperparameter tuning we decided to split the total data set (per airport) into a ratio of **75%-25%**. This splitting ratio was initially random, although the distribution of the target variable (go-around/No go-around) in each partition was maintained. The 75% partition was then used to perform a cross-validation optimisation of the hyperparameters. Once the best hyperparameters were found, they were used to re-train a final model. Finally, the 25% partition was used to validate the final model performance. Although this procedure is not the most optimal, it sufficiently reduces the bias and provides reliable model performance information. Other procedures, such as nested cross-validation, can further reduce bias but at a high computational cost.

In the following tables we can see the results of the models for EDDM and for EDDF after performing the hyperparameter tuning.

Table 11: LightGBM performance metrics after hyperparameter tuning (EDDM)

EDDM					
Go-around	Precision	Recall	F1 score	ROC-AUC	PR-AUC
True	0.9111	0.2135	0.3460	0.9004	0.3521
False	0.9972	0.9999	0.9986		

Table 12: LightGBM performance metrics after hyperparameter tuning (EDDF)

EDDF					
Go-around	Precision	Recall	F1 score	ROC-AUC	PR-AUC
True	0.8974	0.4281	0.5797	0.9105	0.5285
False	0.9980	0.9998	0.9989		

After the hyperparameter tuning and the re-training of the models, we can appreciate a slight improvement in the models performance. The lack of a more significant improvement in the performance could be due to the fact that the optimal hyperparameters were not found using the bayesian optimization technique or that the model and/or the data had reached the maximum of their predictive capabilities. Even so, we can see a considerable performance of the models, especially considering the particularities of the operation to be predicted. The **confusion matrices** of the models for both airports are presented below. A Confusion matrix is simply a table that describes the performance of a classification model. Four different values can be found:

- **True Positive (TP)**: When the model predicts a "1" and the actual data is also a "1" → A Go-around is predicted and the flight contains a go-around.
- **True Negatives (TN)**: When the model predicts a "0" and the actual data is also a "0" → A flight that lands and was predicted to do so.
- **False Positive (FP)**: When the model predicts a "1" and the actual data is a "0" → A Go-around is predicted but it actually lands. Also known as **Type I Errors**.
- **False Negative (FN)**: When the model predicts a "0" and the actual data is a "1" → A flight is predicted to land but actually a go-around occurs. Also known as **Type II Errors**.

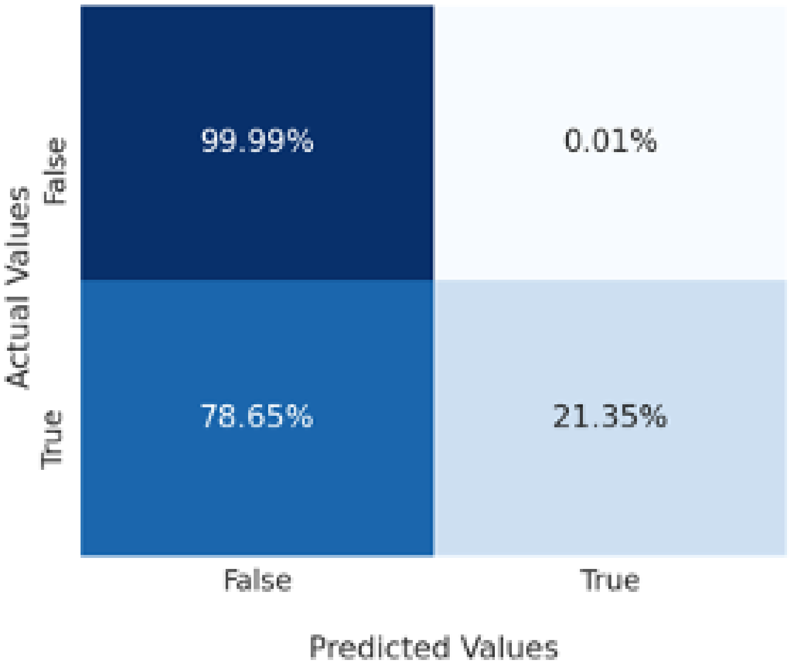


Figure 37: EDDM - Confusion Matrix

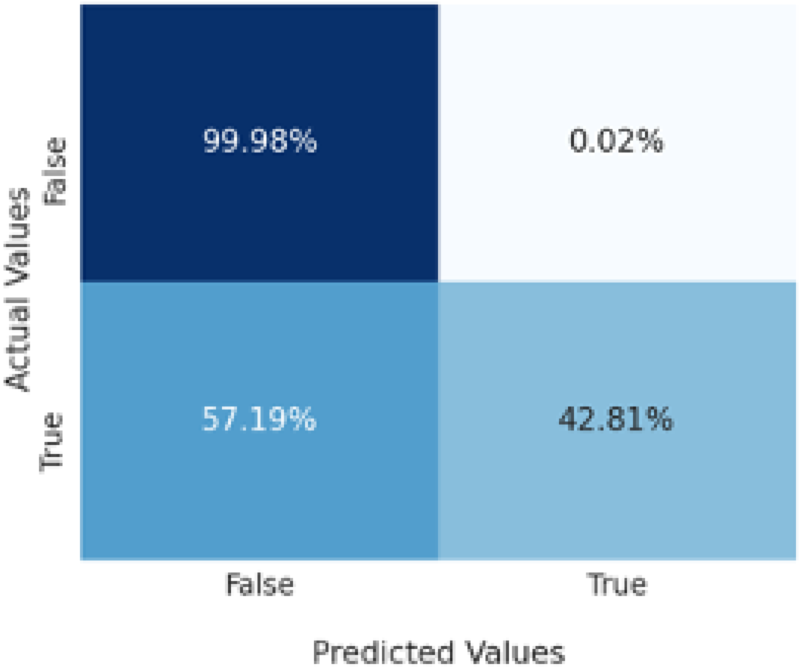


Figure 38: EDDF - Confusion Matrix

Figures 37 and 38 show the resulting confusion matrix for each of the models. These visualisations allow us to see a summary of the performance of the model with regards to total number of events present in the data. Reading from top to bottom and from left to right it can be seen in the case of EDDM that the "Actual Values – False" (Non-go-around approaches) the model identifies 99.99% of the events. This means that in the case of 10000 approaches of which 9965 are non-go-around events the model has correctly identified 9964 only incorrectly flagging one approach as go-around. If we go to the "Actual Values – True" (go-around events) following the EDDM case we see that the model this time only correctly identifies 21.35% of all go-arounds. Using the same 10000 approaches example where 35 would be go arounds the model only identifies correctly 7 and misidentifies 28.

The precision in the prediction of the negative class (No go-around) in both models is very high (99%). This was to be expected given the imbalance in the data discussed above. Also, for both models the prediction accuracy for the positive class (Go-around) is very high (90%) meaning that when the model classifies n approach as performing a go-around, the vast majority of the time it is correct. However, it can also be seen that the model is very selective in classifying the positive cases. In the case of EDDF, the model manages to predict slightly less than half of the approaches that execute a go-around (43%). In the case of EDDM, the model only manages to predict slightly less than a quarter of the go-around approaches (21%). This indicates that, although there is some level of predictability in the data used, almost half of the go-around approximations in the case of EDDF and three quarters in the case of EDDM present a very similar profile to the non-go-around approximations. This situation could be solved by using more data, although the imbalance is generally due to the rarity of the event and not to lack of information. It could also be solved by using more powerful deep learning algorithms, although these introduce more complexity, reduce the possible interpretability and do not ensure a better performance.

To better understand the performance of the classifier, the **Receive Operating Characteristics (ROC)** and **Precision-Recall (PR)** curve are presented below. The ROC curve is a graph showing the performance of a classification model at all possible classification thresholds. This curve represents two parameters: the True Positive Rate (TPR) or Recall and False Positive Rate (FPR) = FP / (FP + TN). The PR curve represents the precision and recall parameters on a curve. Very similar to the ROC curve, the PR curve is a graph showing the performance of a classification model at all possible classification thresholds. It can be seen that in both cases the ROC curve presents a very good result (**AUC ≈ 0.9**) but this result can be initially misleading. The ROC curve can be very useful for problems where the classes are equally distributed. But when the data is imbalanced it can lead to incorrect interpretations. This is because the information represented corresponds to the rate of true and false positives. The false positive rate when the data is imbalanced tends to be very small always due to the large number of negative observations. For all these reasons the PR curve is usually more useful in cases where there is an imbalance between classes.

In problems with imbalanced data such as this one, the minority (positive) class is usually of greater interest than the majority (negative) class. This is why the PR curve is usually a more appropriate metric for these problems. In the PR curve the only important class is the positive class because both precision and recall do not consider true negatives. The EDDF model present a higher PR-AUC (**0.53**) than the EDDM model (**0.35**). Although at first glance this may not seem like a very good result, it should be noted that, unlike the ROC curve, the base threshold of the PR curve depends on the proportion of positive samples in the data. In this case, because 99% of the data are present no go-arounds, the no skill threshold would be around 0.01. This means that in the case of EDDF the classifier the model is 50 times better than a random classifier and in the case of EDDM 35 times better. Furthermore, the PR curve also makes it easy to visualise the trade-off between accuracy and recall. Generally, in a classifier, if you want to increase one of these two metrics, this is achieved at the cost of worsening the other. This is why the user must decide beforehand how he/she prefers the classifier to perform and adjust the classification threshold accordingly.

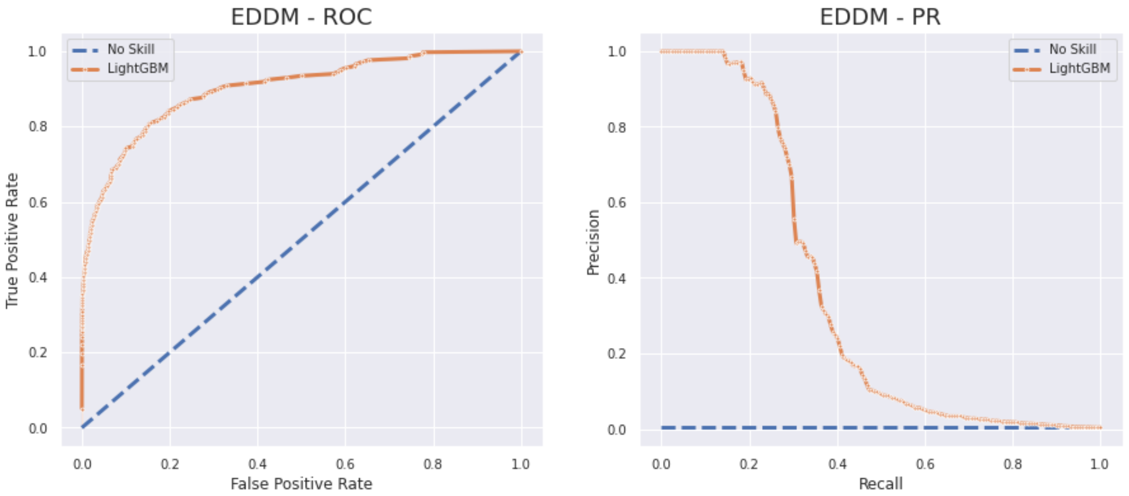


Figure 39: EDDM - Receive Operating Characteristics (ROC) and Precision-Recall (PR) Curves

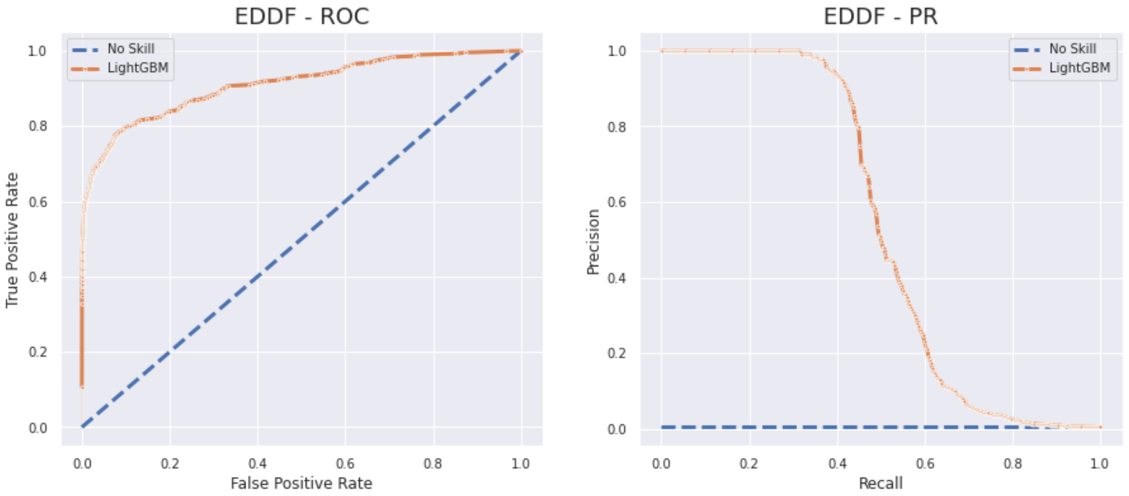


Figure 40: EDDF - Receive Operating Characteristics (ROC) and Precision-Recall (PR) Curves

In order to improve the performance of the predictive model developed for the first ML case study, there are different possible actions to be taken.

- **Getting more data**: Having more data is usually a good idea, although as go-arounds are a rare event the imbalance problem in the dataset would remain,

- **Treat missing and outlier data:** The unwanted presence of missing and outlier values in the training data often reduces the performance of a model. Modifying the current pre-processing stage could achieve a performance improvement.

- **Feature engineering and selection**: The feature engineering phase can also be further explored by developing new features that better reflects the operational scenario. In addition, a feature selection process can also be carried out, whereby features that do not add value to the prediction are eliminated.

- **The use of more powerful models** can also be explored, although usually at the cost of losing explainability and interpretability.

5.4.3 Model explainability

To conclude this section, an initial analysis will be made of the **interpretability** and **explainability** of the models developed. This analysis is not intended to be very exhaustive as the interpretability and expandability of the models will be developed later in the project and the results will be published in Deliverable 4.2. Even so, it is considered interesting to carry out this first analysis in order to have a better **understanding** of how the models work in order to optimise their performance in successive iterations of the same case study as well as being of interest for the other case studies proposed. The **SHAP (SHapley Additive exPlanations)** library will be used for this purpose. SHAP is a game theoretic approach to explain the output of any machine learning model (Lundberg, Scott M., et al. 2020). SHAP provides a visualisation tool that can be used for explaining the predictions made by a model through the computation of the contribution of each feature to the prediction.

Through SHAP we can easily visualise the **20 features** considered most important for each of the models.

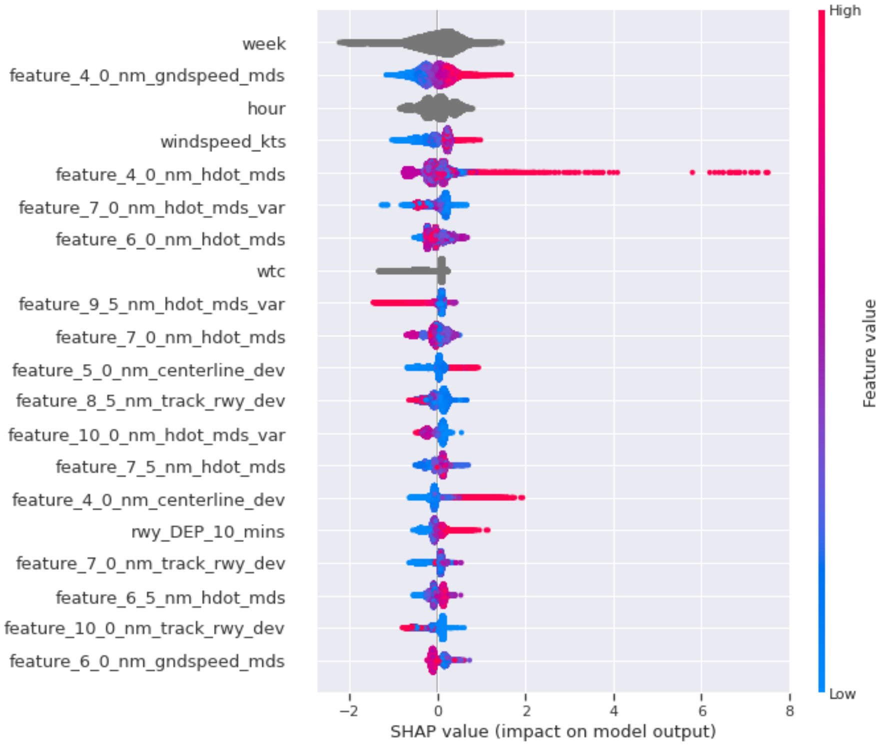


Figure 41: SHAP Feature importance (EDDM)

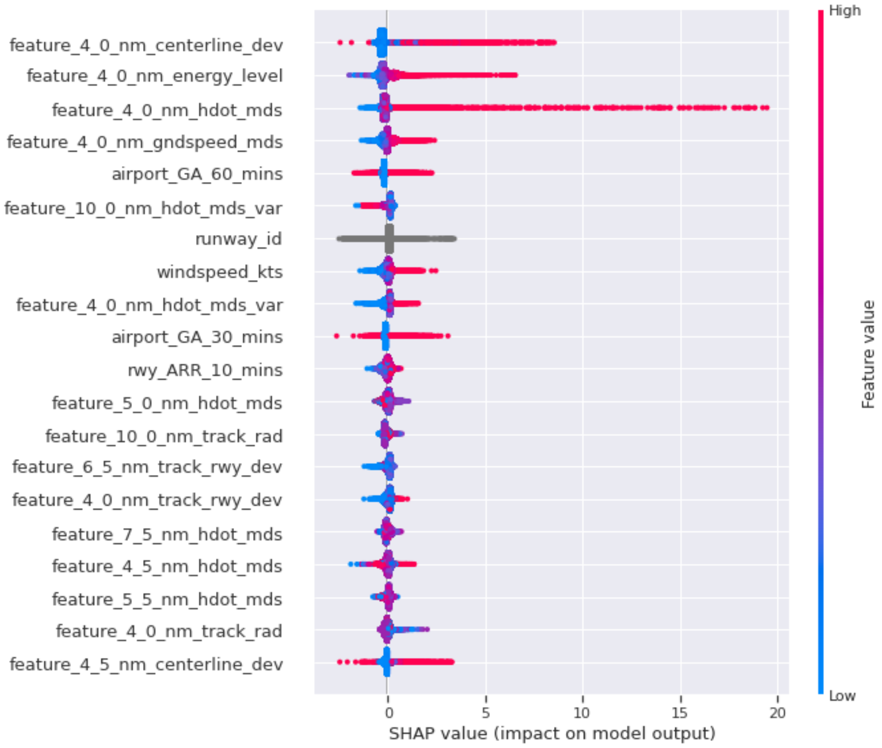


Figure 42: SHAP Feature importance (EDDF)

Figure 41 and 42 shows the SHAP "**Feature importance**" plot for the models developed for EDDM and for EDDF. In this figure we can see on the y-axis the **20 most important features** for the model ordered from top to bottom from most to least important. On the x-axis we can find the **SHAP value** where positive values indicate that they favour a positive prediction (go-around) and negative values favour an approach without go-around. Finally, we find a colour scale for each of the observations. This colour scale corresponds to the value of the feature for each approach. Higher feature values have a pinker colour and lower feature values a blue colour. The grey colour features are categorical features (e.g. WTC) and therefore do not show any colour as there is no intrinsic order in them.

It can be seen that the vast majority of the features most frequently used by the model had a certain level of correlation with the target variable as we have seen in the EDA section above. The first thing you notice is the difference in the most important features between one model and the other. Although there is not a big difference and most of them are aligned, it is curious to observe. The most important feature in the EDDM model is the "**week of the year**" followed by the **ground speed at 4NM**. In the case of the EDDF model the week of the year is not in the top 20 and the ground speed at 4NM is the 4th most important feature. In this case the most important features are "**centerline deviation at 4NM"** and "**energy level at 4NM**". In both cases the features of type "**Approach performance**" preponderate, although in both cases we can also see the importance of some meteorological features such as "**wind speed**". As mentioned above it is interesting in this kind of visualisations that we are not only able to see the feature importance ranking but also the effect of the value on the prediction. We can see how in both cases high "**vertical speed at 4NM**" or a high "**ground speed at 4NM**" favours the prediction of a go-around. Finally, we can see, especially in the EDDF model but also for EDDM, the importance of some "**Airport information**" type features such as "**total number of GA in the previous 60 mins**" or **"runway departures in the previous 10 mins**". Again, in this deliverable this is not intended to be an exhaustive analysis but a first insight into how the models are working. In the next deliverable, a deeper-dive in explainability and interpretability will be carried out, especially in order to merge the results of the predictive models with the WP3 **Risk Framework**.

6 Conclusions

In this deliverable a technical summary of the first phase of **Work Package 4 (WP4)** of the SafeOPS project has been presented. WP4 is dedicated to the development of all tasks related to the **technical development of a data-driven predictive solution** for the prediction of go-around situations in airport operations. The work carried out in this initial phase consisted of the development of the **technical infrastructure** necessary for the development of the solution as well as the definition of the possible **use cases** and the implementation of the **data processing pipelines** required for these. In addition, the final objective was the complete development of a **first AI/ML solution**, identifying insights from the collected data, performing a first model benchmark and obtaining an initial set of results to be used as a stepping stone in the continuous development of the solution in the project.

With respect to the objectives set out, it can be said that it has been possible to develop a first set of predictive models that, far from being perfect, have performed relatively well, especially considering the nature of go-around events and their relative scarcity (1-3 per 1000 approaches (Flight Safety Foundation 2017)). In both airports investigated (EDDM and EDDF), although the total number of go-arounds identified by the models is not high (21% and 43% respectively), it has been possible to obtain a go-around prediction precision of around 90%. This means that although the model is not able to identify all the go-arounds, those that it identifies are very likely to ocurre and the false alarm rate is very low. Again, although the recall rate was not particularly high, this was to be expected based on previous work made in the project (SafeOPS D2.1). Two main types of go-arounds were identified: **ATC induced go-arounds** and **Flight crew induced go-arounds**. Due to the limitations of data such as ADS-B it was speculated from the beginning of the project that go-arounds of the ATC induced type would be very difficult or impossible to predict as the cause of the go-around (e.g. runway is blocked or other traffic that requires immediate priority) would not be reflected in the data and therefore the model would not be able to identify these events correctly. This made the main realistic option of only being able to predict flight crew induced go-arounds causing the final number of viable go-arounds for prediction to be much lower than the 1-3 per 1000 approaches ratio. We have also been able to carry out an initial investigation into the main factors, determined by the models, that mainly influence the prediction of a go-around. Furthermore, as mentioned above, one of the objectives of the first phase of WP4 was to lay out the technical foundations (infrastructure + data pipelines). This now allows us to continue the technical development in an effective and efficient way and to propose new solutions based on the results and the possible feedback received. Therefore, a series of next steps for the second phase of WP4 can be established:

- Prepare **workshops** with ATCOs and airlines to present the initial results of the predictive models and to obtain useful **feedback** and suggestions for future developments.
- Develop the other ML case studies proposed (**ML_CS_02** and **ML_CS_03**). Explore their results, their predictive capacity and the explainability and interpretability of the models.
- Deepen the understanding of how the models work and expand on the **human interpretability** side of the solutions understanding and focusing on the correct ML Interpretation by the users.
- Coordinate the work done in **WP4** with **WP3,** strengthening the interaction between them, especially in the delivery the predictive results to be fed into the **Risk Framework (RF)**

7 References

[1] (SafeOPS D2.1) D2.1 Development of Use Cases, User Stories and Requirements: https://innaxis-comm.s3.eu-central-1.amazonaws.com/SafeOPS/SafeOPS_D2.1_v00.02.00.pdf

[2] (SafeOPS D3.1) D3.1 Risk framework: scope and SoA: https://innaxis-comm.s3.eu-central-1.amazonaws.com/SafeOPS/D3.1_Risk-framework-scope-and-SoA_V00_02_00.pdf

[3] (analyticsvidhya) Data Validation in Machine Learning is imperative, not optional. Analytics Vidhya https://www.analyticsvidhya.com/blog/2021/05/data-validation-in-machine-learning-is-imperative-not-optional/

[4] (Dhief, Imen, et al. 2021) "A Tree-based Machine Learning Model for Go-around Detection and Prediction." 11th SESAR Innovation Days

[5] (Proud, Simon Richard 2020) "Go-around detection using crowd-sourced ADS-B position data." Aerospace 7.2 (2020): 16.

[6] (SafeClouds) European project description in: https://cordis.europa.eu/project/rcn/206420/en

[7] (SafeClouds D4.3) D4.3 Predictive analytics and identification of unknown hazards https://cordis.europa.eu/project/id/724100/results

[8] (Jolliffe, I. T. 2002). Graphical representation of data using principal components. Principal component analysis, 78-110.

[9] (Micci-Barreca, D. 2001). A preprocessing scheme for high-cardinality categorical attributes in classification and prediction problems. ACM SIGKDD Explorations Newsletter, 3(1), 27-32.

[10] (Chawla, Nitesh V., et al 2002). "SMOTE: synthetic minority over-sampling technique." Journal of artificial intelligence research 16 (2002): 321-357.

[11] (Lundberg, Scott M., et al. 2020) "From local explanations to global understanding with explainable AI for trees." Nature machine intelligence 2.1 (2020): 56-67.

[12] (Krawczyk, B. 2016). Learning from imbalanced data: open challenges and future directions. Progress in Artificial Intelligence, 5(4), 221-232.

[13] (Flight Safety Foundation 2017) "Go-Around Decision-Making and Execution Project". Flight Safety Foundation. 2017

[14] (Sharayu Rane) "The balance: Accuracy vs. Interpretability". Sharayu Rane. towards data science

8 List of Abbreviations

ADS-B	Automatic Dependant Surveillance-Broadcast
AI	Artificial Intelligence
ATCO	Air Traffic Control Officer
AUC	Area Under the Curve
AWS	Amazon Web Services
BeSt	BeaconStack
CAVOK	Ceiling and Visibility OK
CORDIS	Community Research and Development Information Service
CRISP-DM	Cross-Industry Standard Process for Data Mining
DPA	Data Protection Agreement
ECAC	European Civil Aviation Conference
ECMWF	European Centre for Medium-Range Weather Forecasts
EDA	Exploratory Data Analysis
EDDF	Frankfurt airport
EDDM	Munich airport
ETL	Extract, Transform, Load
EU	European Union
FDM	Flight Data Monitoring
GA	Go-Around
ICAO	International Civil Aviation Organization
IMC	Instrument Meteorological Conditions
METAR	Meteorological Aerodrome Reports
ML	Machine Learning
ML_CS	Machine Learning Case Study
MSP	Multi-Side Platform
NM	Nautical Miles

OL	Operational Layer
PCA	Principal Components Analysis
PL	Predictive Layer
PR	Precision-Recall
QAR	Quick Access Recorder
RF	Risk Framework
ROC	Receiver Operating Characteristic
SaaS	Software as a Service
SDF	Smart Data Frames
SHAP	Shapley Additive Explanations
SI	International system of units
SMOTE	Synthetic Minority Oversampling Technique
VMC	Visual Meteorological Conditions
WP	Work Package
WTC	Wake Turbulence Category

-END OF DOCUMENT-