Modeling pre-tactical Air Traffic Flow Management decision processes with Machine Learning/Deep Learning approaches

IB-328-2022-3

Master Thesis
in the Field of *Machine Learning*
Studied Under *Mechatronics*
at
Technischen Universität Hamburg
Institut für Lufttransportsysteme

Sujay Suresh
Matriculation- Nr.: 21862615

First Examiner:	Prof. Dr.-Ing. Volker Gollnick
Second Examiner:	Prof. Dr.-Ing.Thorsten A. Kern
Supervisor:	Manuel Derra
Duration:	15.07.2021 - 14.02.2022

Hamburg, 14. February 2022

I declare in lieu of an oath that this document has been prepared independently and by my own hand.
Hamburg, On 14. February 2022

Signature

Task for the
Master Thesis for
Sujay Suresh

Matriculation No: 21862615

Modellierung von prä-taktischen Air Traffic Flow Management Entscheidungsprozessen mittels Machine Learning/Deep Learning Methoden

Modeling pre-tactical Air Traffic Flow Management decision processes with Machine Learning/Deep Learning approaches

Introduction

The rapid increase of European air traffic in recent decades (pre-Covid-19) has pushed airspace capacity to its limits. Air Traffic Flow Management (ATFM) in Europe aims to balance limited capacities with growing demand. One of the main constraining factors in this process are existing uncertainties in current prediction logic on a pre-tactical level. These lead to necessary capacity buffers, which result in reduced network efficiency. As increasing air traffic demand rates can be assumed in a post-Covid era, this will become a pressuring issue again in the long term.

Task Description

Within the scope of this master thesis, a structured analysis of a comprehensive set of flight plan data is to be carried out with the help of advanced analytical methods from the fields of Machine Learning and Deep Learning in order to draw conclusions about ATFM-decision processes in the pre-tactical phase regarding operational influence parameters. For this purpose, Sabre Airport Data Intelligence data as well as Eurocontrol’s RnD data should first be merged, parametrized and clustered in order to be able to subsequently feed them into the training of a specifically developed ML/DL model that is capable of predicting Operational Flight Plan trajectories. In the context of a large amount of parameterised data, it is important to identify those mission parameters that have a high impact on the prediction quality. For validation, the currently used PREDICT logic at Eurocontrol should be implemented conceptually and compared to the developed model. Furthermore, the model should be investigated for a possible increase in prediction-accuracy of planned airspace demand and therefore a reduction of uncertainties.

Nomenclature

Abbreviations

Abbreviations	**Full Forms**
ACC	Area Control Center
ADP	ATFCM Daily Plan
ANSP	Air Navigation Service Provider
AO	Aircraft operators
ATC	Air Traffic Control
ATFCM	Flow and Capacity Management
ATFM	Air Traffic Flow Management
AU	Airspace User
CASA	Computer Assisted Slot Allocation
CDM	Collaborative Decision-Making
CPU	Central Processing Unit
DBSCAN	Density-Based Spatial Clustering of Applications
DCB	Demand Capacity Balancing
DCNN	Deep Convolution Neural Network
DL	Deep Learning
ECAC	European Civil Aviation Conference
EDA	Exploratory Data Analysis
EDDF	Airport Frankfurt Am Main
EGLL	Airport London Heathrow
ETFMS	Enhanced Tactical Flow Management System
eps	Epsilon
ETO	Estimated Time Over
ETOT	Estimate Take-Off Time
FMP	Flow Management Position
FPL	Flight Plans
GPU	Graphics Processing Unit
HPC	High Performance Compute
IATA	International Air Transport Association
ICAO	International Civil Aviation Organization
IFPS	Initial Flight Plan Processing System
KNN	k-Nearest Neighbors
LSTM	Long Short Term Memory
minPts	Minimum Points
ML	Machine Learning
NM	Network Manager
NN	Neural Network
NMOC	Network Manager Operations Centre

NOP	Network Operations Plan
OD	Origin - Destination
OMBD	Airport Dubai International
RHEL	RedHat Enterprise Linux
SLURM	Simple Linux Utility for Resource Management
TU	Technische Universität
XGBoost	Extreme Extreme Gradient Boosting

Table Of Contents

1. Introduction

The first complete passenger aircraft, the Junkers F13, made its maiden flight in 1919, laying the groundwork for civil aviation. Since then, aviation's economic impact has steadily increased. Ticket prices have decreased due to technological advancements and competitive pressures, resulting in ever-increasing demand. Between the year 2013-2019 alone global number flight per years grew from 20% to 37.3% which is of 38.9 million. Due to COVID-19 this number dropped to 16.9 million by 2020 [1].

Following COVID, it is expected that air traffic will resume operation at full capacity, as it did during the 2019 [2]. A growing demand is being driven by the ever-increasing number of flights, which is driving higher safety and efficiency requirements and increased competitive pressure. In order to meet these needs, air traffic flow management (ATFM) was implemented in Europe, which performs the functions of strategic and tactical demand-capacity smoothing as a part of flight planning and ATFM network control.

ATFM in Europe aims to balance limited capacities growing demand. The number of en-route ATFM delays has doubled within one year from 2017 to 2018 [1]. This is an indication that European airspace is reaching its capacity limits. To avoid congestion of individual ATC airspaces and resulting traffic flow measures, a more accurate demand prediction is of interest. This problem has been addressed by the *Deutsche Flugsicherung* (DFS) declared in the context of the *NM User Forum* in January 2019, next to weather, as the most important task to be solved by the *Upper Area Control Center Karlsruhe* [3].

Post-COVID, due to an increase in air traffic demand rates the air traffic congestion will become a pressuring issue again in the long term. One of the main constraining factors is the uncertainty that exists in current prediction logic in the pre-tactical level. The current prediction logic is purely based on a fixed decision flow process and does not follow any relational procedure that takes patterns in historical data into account. The lack of uncertainty introduced by this prediction in the Flight Plans (FPL) forecast results in inefficient planning processes due to the need for bigger safety buffers [4].

This is an ideal application area for machine learning and deep learning algorithms because it is a complex problem with many variables. As a result, the work's goal emerges, as outlined below.

Objective

Within the scope of this master thesis, a structured analysis of a comprehensive set of flight plan data is to be carried out with the help of advanced analytical methods from the fields of Machine Learning and Deep Learning in order to draw conclusions about ATFM-decision processes in the pre-tactical phase regarding operational influence parameters. For this purpose, Sabre Airport Data Intelligence data as well as Eurocontrol's RnD data should first be merged, parametrized and clustered in order to be able to subsequently feed them into the training of a specifically developed ML/DL model that is capable of predicting Operational Flight Plan trajectories. In the context of a large amount of parameterised data, it is important to identify those mission parameters that have a high impact on the prediction quality. For validation, the currently used PREDICT logic at Eurocontrol should be implemented conceptually and compared to the developed model. Furthermore, the model should be investigated for a possible increase in prediction-accuracy of planned airspace demand and therefore a reduction of uncertainties

2. Fundamentals

2.1 Air Traffic Flow Management

Air Traffic Flow Management(ATFM) is "a service established with the objective of contributing to a safe, orderly and expeditious flow of air traffic by ensuring that *Air Traffic Control* (ATC) capacity is utilized to the maximum extent possible, and that the traffic volume is compatible with the capacities declared by the appropriate ATS authority" [5].

ATC is a service provided by *Air Traffic Controllers* who guides the aircraft over defined controlled airspace and also on the ground. The primary job is to ensure that there is no collision and congestion in the airspace for the safe operation of the aircraft.

From the previous chapter 1, for the year 2018, the aircraft delay has doubly increased. One of the main causes of aircraft delays is that the available ATC capacity was not able to regulate all of the demands of the flights safely. A demand-capacity imbalance is used in the aviation industry to describe this as a problem. ATFM's main goal is to design and implement actions to *Demand-Capacity Balancing* (DCB). Background information on ATFM will be explained in the coming sections of this chapter.

2.1.1 Definition and Stakeholders

There are two main goals of ATFM. The first goal is to optimize available capacity, while the second is to evade excessive demand. Capacity in ATFM refers to the number of flights that can be safely managed in a given airspace sector over a given period (typically the hourly rate) and also the number of flights that an airport can safely manage. *Air Navigation Service Providers* (ANSP) based on available workforce and experience address the ATFM capacity concerns. Demand is defined as the number of aircraft planned to enter a specific air sector during a specific period. The planned demand is determined prior to the flight using the filed flight plans[6].

The ATFM process involves several stakeholders. *Aircraft operators* (AOs), airports, and ANSPs are the primary stakeholders. It is relatively uncommon for these three party's objectives to be at odds. As a result, each of these stakeholders input must be considered for ATFM to make it a successful process. The European Commission created the job of *Network Manager* (NM) to administer this process and ensure transparency and fairness to all parties concerned, and EUROCONTROL was nominated for the position. The European Commission has entrusted the NM with centralized management of the European ATM network as well as the management of limited resources via a *Collaborative Decision-Making* (CDM) approach.

The acronym ATFCM stands for ATFM, but there are substantial distinctions between the two acronyms. The first and most significant point is that ATFCM can be thought of as a component of the ATFM process. While ATFM refers to the collaborative planning and decision-making process for guaranteeing maximum capacity utilization and traffic flow safety, ATFCM refers to the actions that must be implemented to achieve the ATFM's high-level objectives. Another distinction between the two abbreviations is that ATFM includes all four primary stakeholders (NM, AOs, airports, and ANSPs), ATFCM only includes the NM and ANSPs. The Flow Management Position (FMP) is a crucial function for ANSPs. As a local specialist in a specific area control center, the FMP assists the NM with its flow management duties (ACC). The distinction between the two acronyms is depicted clearly in Figure 2.1.

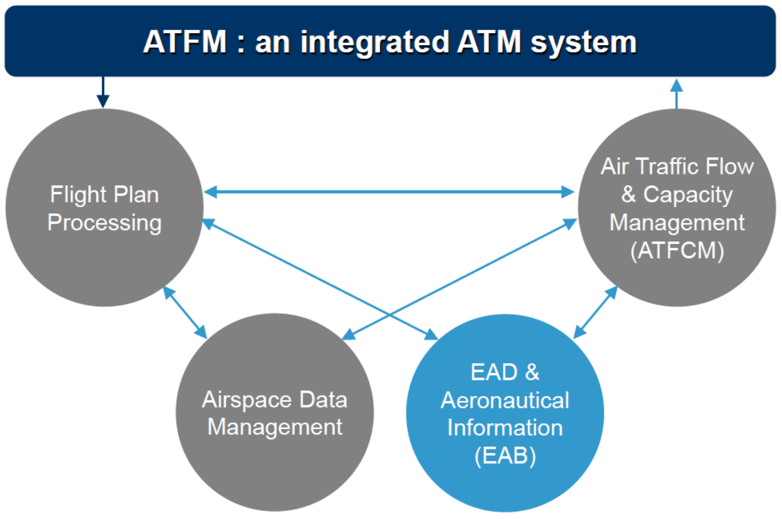

ATFM : an integrated ATM system
Flight Plan Processing
Air Traffic Flow & Capacity Management (ATFCM)
Airspace Data Management
EAD & Aeronautical Information (EAB)

Fig. 2.1: ATM integrated representation of ATFM process [7]

2.1.2 Air Traffic Flow and Capacity Management

This chapter describes *Air Traffic Flow and Capacity Management* (ATFCM) and its effects on capacity and delays in more detail.

ATFCM strives to ensure that available airspace and airport capacity is sufficient to satisfy traffic demand, and that after the most recent capacity possibilities have been exhausted, the demand is sufficient to meet the maximum amount of available space and capacity. Accordingly, flow procedures, such as the allocation of individual aircraft departure periods (slots), may be implemented in order to alleviate bottlenecks and eliminate safety concerns to the greatest extent possible. A constant stream of communication and information exchange with all of Europe's air traffic control units and aircraft operators is maintained throughout all of this activity [8].

ATFCM Phase

The ATFCM process according to *European Civil Aviation Conference* (ECAC) region will be carried out in four phases as shown in fig .

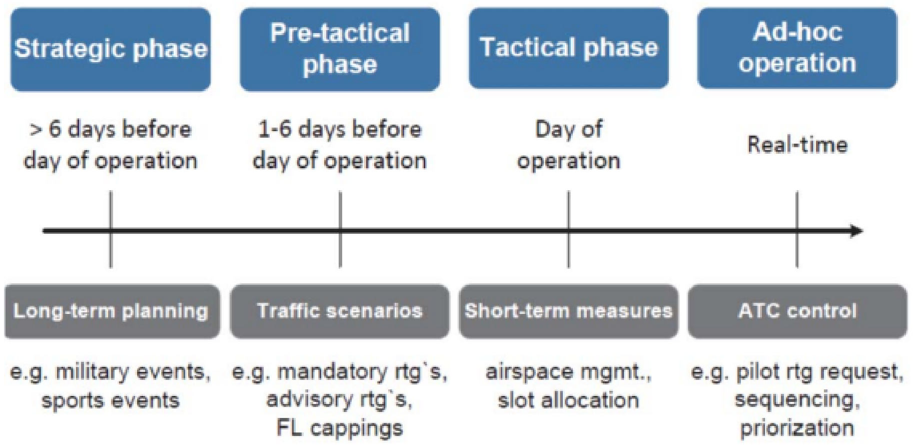

Strategic phase
Pre-tactical phase
Tactical phase
Ad-hoc operation
> 6 days before day of operation
1-6 days before day of operation
Day of operation
Real-time
Long-term planning
Traffic scenarios
Short-term measures
ATC control
e.g. military events, sports events
e.g. mandatory rtg`s, advisory rtg`s, FL cappings
airspace mgmt., slot allocation
e.g. pilot rtg request, sequencing, priorization

Fig. 2.2: ATFCM Phases [9]

- The **Strategic Phase** is carried out between 6 months to 6 days prior to the flight operations. In this phase, initial planning for the traffic volume is carried out. Events known in advance , which influence the demand are taken into account to identify the bottlenecks such as major demand/capacity imbalances to plan the *Network Operations Plan*(NOP).

- The **Pre-tactical phase** is carried out between 6 days to 1 day prior to the flight operations which makes revisions to the capacity plan that was prepared during the **strategic phase**. Initial ATFCM measures, such as time-limited regulations for routes and flight levels, are prepared for smooth demand. These are recorded in the *ATFCM Daily Plan* (ADP).

- In the **Tactical Phase** occurs on the day of operations and entails modifying the ADP in real time. This phase ensures that the strategic and pre-tactical phases' actions are sufficient to address the demand-capacity imbalances. Disruptions such as staffing issues, large meteorological events, crises and special events, unforeseen constraints in land or air infrastructure, etc. may necessitate adjusting the original plan. Providing reliable data allows for short-term forecasting, including the impact of any incident, and maximizes existing capacity without jeopardizing safety.

- **Post operational analysis** phase is the final phase in the ATFCM planning. While taking into consideration performance targets, this phase compares the predicted outcome (if applicable) with the actual measured outcome, which is typically measured in terms of delay and route extension. The formulation of best practices and/or lessons learned for improving upon certain operational processes and activities is the end result of this phase.

The main goal of dividing AFTCM planning into four phase is to address the DCB as early as possible to ensure smooth flight operation. A deeper look into DCB is explored in the next section.

2.1.3 Demand Capacity Balancing

To resolve demand-capacity imbalances, the NM and the affected FMP jointly consider possible solutions and choose the best one to implement. One of the requirements of this process is that all stakeholders' needs to be considered before a measure is implemented. The measures that are being considered to address demand-capacity issues, as well as their hierarchical order, will be stated in the following sections.

The initial steps to take are to optimize the available airspace capacity. This can take the form of sector management, which involves changing the layout fig 2.3 of a single sector by splitting it into smaller ones or collapsing smaller sectors into a single one. Another way to improve capacity is to allocate additional capacity. This can be accomplished by using holding patterns, minimizing traffic complexity, or coordinating airspace use with the military [8].

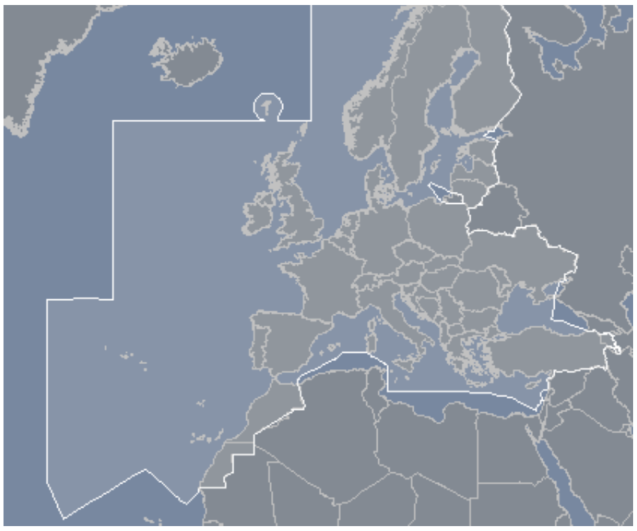


Fig. 2.3: European Airspace [10]

Whenever a class of solutions has been exhausted and a capacity mismatch still exists, the *Network Managers Operation Center* (NMOC) will seek to move demand into regions where capacity is still available. This can include traffic flow rerouting or flight level management, in which a portion of the flows is assigned to fly at different flight levels. Other solutions in this category include advance traffic that can departure slots sooner and tactical FMP interventions.

After the preceding approaches have failed to correct the imbalance, the only remaining alternative is to regulate airport demand. There are two basic strategies that can be used to achieve airport demand moderation. The first is ground holding, which requires all aircraft that meet certain conditions to remain on the ground until further notice. This type of precaution is typically used in the event of extreme weather, accidents, or to reduce long periods of in-flight wait time. Because the length of this holding time is determined by the occurrence that has produced it, the delay experienced by affected flights cannot be forecast. The third option, and the subject of this study, is to use ATFCM regulations. These regulations are used for a variety of purposes and are divided into 14 main categories in general [11]. Figure 2.4 depicts the various reasons for restricting demand, with the related total ATFM delay for the year 2020 for each cause.

A flight plan is required for all flights entering the NM area of operations. These plans are collected in the Integrated *Initial Flight Plan Processing System* (IFPS), which is then supplied to the *Enhanced Tactical Flow Management System* (ETFMS) along with the ANSPs declared capacity . The *Computer Assisted Slot Allocation* (CASA) system then uses these inputs. The NM initiates ATFCM restrictions for restricted areas where demand exceeds capacity after they have been identified. Flights that will be entering controlled areas are given an ATFM delay, which is a pre-departure delay on the ground. The slot allocation process is the process of attributing ATFM delays to particular aircraft.

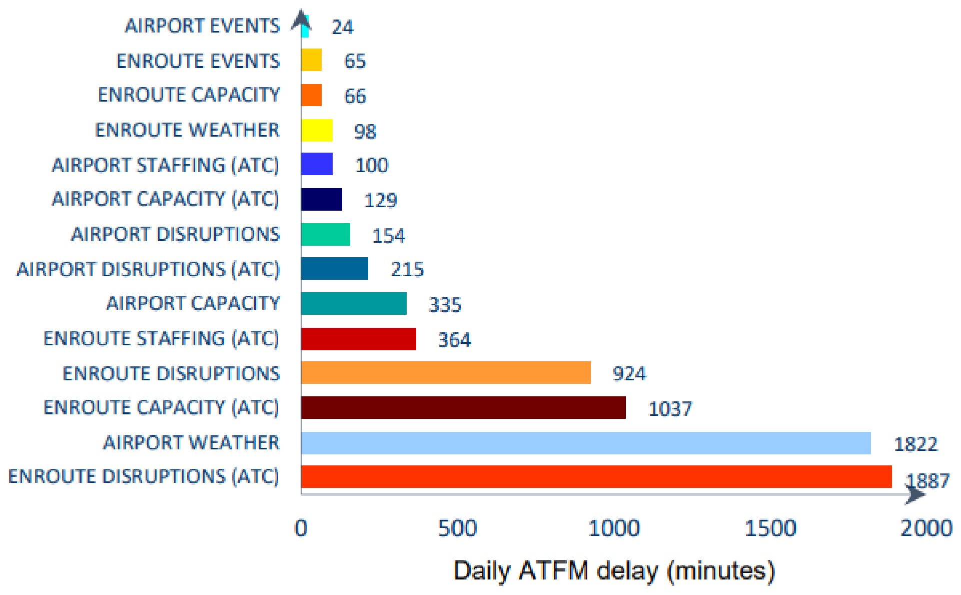


Fig. 2.4: Reason for AFTM Delays for the year 2020 [11]

Consider a flight that originates from A, terminates at B, and goes through seven controlled sectors to understand the slot allocation method. Based on the projected time to enter the sector, an *estimated time over* (ETO) is calculated for each sector. Assume that demand capacity imbalances exist in sectors four (in the middle of the route) and seven(at the destination). The sector's capacity determines slot times; for example, if the capacity is 35 flights per hour, a slot is available every two minutes to enter the sector. The first available slots are searched for both regulated sectors, and the most penalizing one is chosen. Consider that the initial ETO for sector four was 19:18, and the first available slot to join this sector is 19:22; the original ETO for sector seven was 19:51, and the first open slot is 19:57. In this example, the slot for entering sector five at 19:57 will be chosen since it is the more punishing, with a 6-minute penalty than the other slot's 4-minute delay. As a result, the flight's *estimated take-off time* (ETOT) will be moved back by 6 minutes, allowing it to enter the regulated area at the specified slot time. Finally, when multiple flights are evaluated for the procedure, the slots are assigned using the 'First Planned, First Served' approach [8]. "Flights should arrive over the regulated region in the same order (based on ETOs) as they would if there were no restriction," says this principle.

From the above example it can be inferred that the prediction of airspace capacities will help in managing the ATFM delays and also to plan for required ATC staff.This work will explain how *Machine Learning* may help estimate future airspace demand.

2.2 Machine Learning

The modern-day buzzword *Artificial Intelligence* dates to Greek and Egyptian mythology, where mechanical and artificial beings were referred to as artificial intelligence. In the academic world, the initial idea of artificial intelligence took birth at a workshop that was held in 1956 at Dartmouth college [12]. Artificial intelligence is a branch of computer science that uses algorithms to deal with machine intelligence, *Machine learning* is a sub-field of artificial intelligence that deals with IT systems that can recognize patterns and regularities in data and algorithms to produce solutions. In 1959, Arthur Samuel, in his paper [13] stated that “Machine Learning is the field of study that gives computers the ability to learn without being explicitly programmed”.

Raise in Machine learning has been possible due to the development in the computational capacity of GPUs and CPUs in recent years, making it possible for computers to handle more complicated problems more effectively than people. In general modern-day Machine Learning has two goals : One is to classify data based on models that have been trained on past data, and the other is to generate predictions about future outcomes based on these models. For instances, an example for a Machine Learning model detects and classifies malignant moles in images based on computer vision, reducing the workload for doctors. A machine learning system for stock trading, on the other hand, may advise the trader of probable future projections and thus support in the decision-making process [14].

2.2.1 Types of Machine Learning

Machine learning problems can be divided into three main groups namely *Supervised*, *Unsupervised* and *Reinforcement Learning*. Training of the data depends on the available data and the type of problem present.

Supervised Learning
The majority of the problems that exist fall under *Supervised Learning*. The input and output values will be known for the available data set. The output parameters indicate the prediction results, referred to as labels. Based on the problem, a model will be trained to produce a heuristic that will decide the output parameter corresponding to the trained input parameters. The recognition of handwritten numbers from the MNIST dataset [15] is an example of this. It is a collection of images depicting the numbers 1 through 9, each written by a different person. Each image was given a label, which is the appropriate number. As a result, this is an issue of supervised learning. The computer is provided visuals with associated numbers during training, as shown in Figure 2.5.

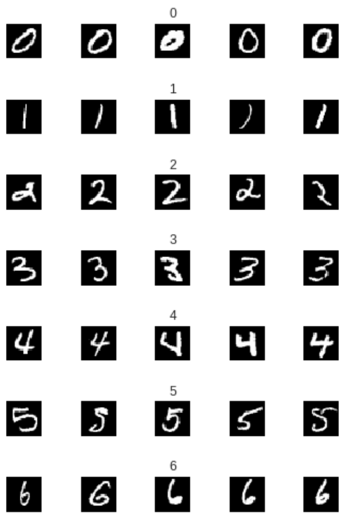


Fig. 2.5: Sample images from the MNIST dataset [16]

Unsupervised Learning
In *Unsupervised Learning* the output labels of the training data which are present in *Supervised Learning* will not be present. Hence this type of data set is called *Unlabeled Data*. The *Unlabeled data* is used to group data points (clustering) or to observe and detect the pattern in the data points to formulate meaningful interpretations. For example, *Unlabeled data* will be passed into an *Unsupervised Learning* algorithm to group the data in meaningful way as shown in Figure 2.6.

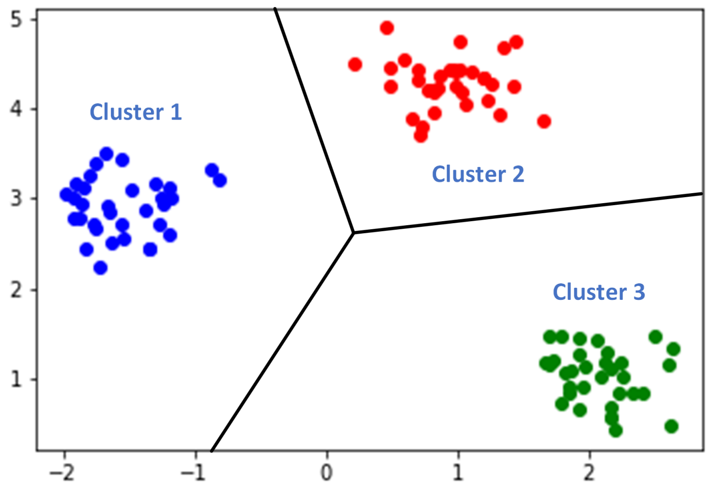


Fig. 2.6: Grouped data [17]

Figure 2.6 shows the schematic representation of the division of a data set into different groups(clusters). Again, color coding is done to identify the area of the group. The black lines show the grouped areas.

The training data in Supervised Learning problems must have output labels, which in many cases require a time-consuming and error-prone manual labeling process. *Unsupervised Learning* algorithms can often replace this process. The combination of Unsupervised Learning and Supervised Learning is called *Semi-Supervised Learning*, which yields almost similar accuracy as *Supervised Learning.*

Reinforcement Learning
Reinforcement Learning, in contrast to the other machine learning problems previously discussed, does not require any data to be collected beforehand. Instead, the data is trained and labeled in a simulated environment in many runs. One or more agents learn the strategy in order to increase the rewards (R) gained within the simulated environment (see Fig 2.7). In advance, the agent is not shown which action (A) will result in the most significant reward. This type of algorithm is used, for example, in games such as chess. The model plays a variety of simulated chess games and learns moves that eventually lead to victory over time. One reward could be the capture of an opponent's piece.

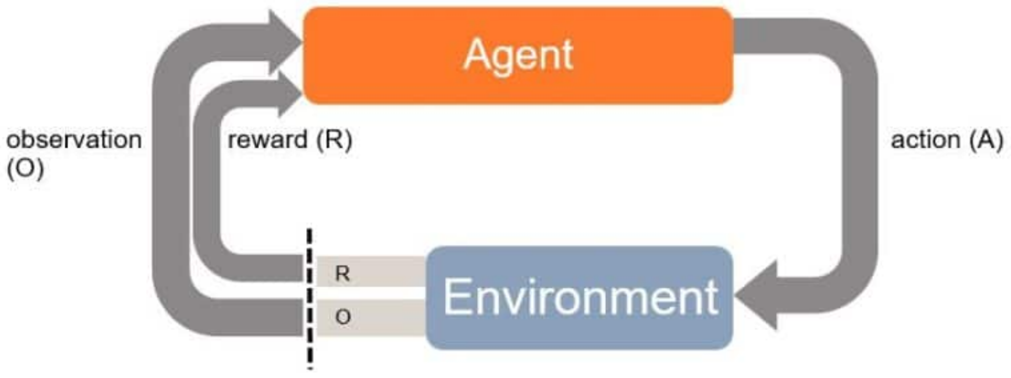


Fig. 2.7: Iterative Reinforcement Learning with one agent [18]

In this thesis study, to solve the problem *Semi-Surpervised Machine Learning* approach is taken based on the data that is present. It means the approaches will consist of both *Unsupervised* and *Supervised Learning.*

No free Lunch theorem [19] states that there is no such algorithm that optimizes the problem in the best way will perform worse on the rest of other optimized problems. Hence, to find the best-performing algorithm for the problem, different algorithms should be explored.

The appropriate algorithm will be considered based on the available data set and the problem. As mentioned above, this work uses custom machine learning involving both unsupervised and supervised techniques. First, in order to create the output labels for the Supervised Learning models, unsupervised algorithms such as clustering techniques will be explored, and later various supervised techniques will be explored such as *Random Forest*, *XG-Boost*, and *Lightgbm.*

Even though the majority of applications today are based solely on *Supervised Learning*, most of the data that is available around the world is unlabeled. As previously discussed, using *Unsupervised Learning*, the problem where the data are unlabeled will be labeled. As a result, in the following section, a thorough explanation of *Unsupervised Learning* will be provided.

2.2.2 Unsupervised Leaning Techniques

The computer scientist Yann LeCun famously said that "If intelligence was a cake, *Unsupervised Learning* would be the cake, *Supervised Learning* would be the icing on the cake, and Reinforcement Learning would be the cherry on the cake" [20]. To build any effective machine learning system in the real-world *Unsupervised Learning* makes an excellent foundation.

To understand the importance of using *Unsupervised Learning* in any intelligence system, have a closer look into one such example; consider an object tracking system as shown in Figure 2.8 such as a traffic cam, where the main goal is to classify different kinds of vehicles into different classes like cars, trucks, and bikes. It is easy to build a camera system that automatically creates thousands of pictures every day. This only creates a dataset without the associated output labels, which are required for Supervised Learning problems like object tracking. So, all the available algorithm under *Supervised Learning* requires labeled data, problems like object tracking. Human experts conducted the labeling process by going through every picture manually in the past. As it is known, any human-involved task would be cumbersome, tiresome, expensive, time-consuming, and error-prone.

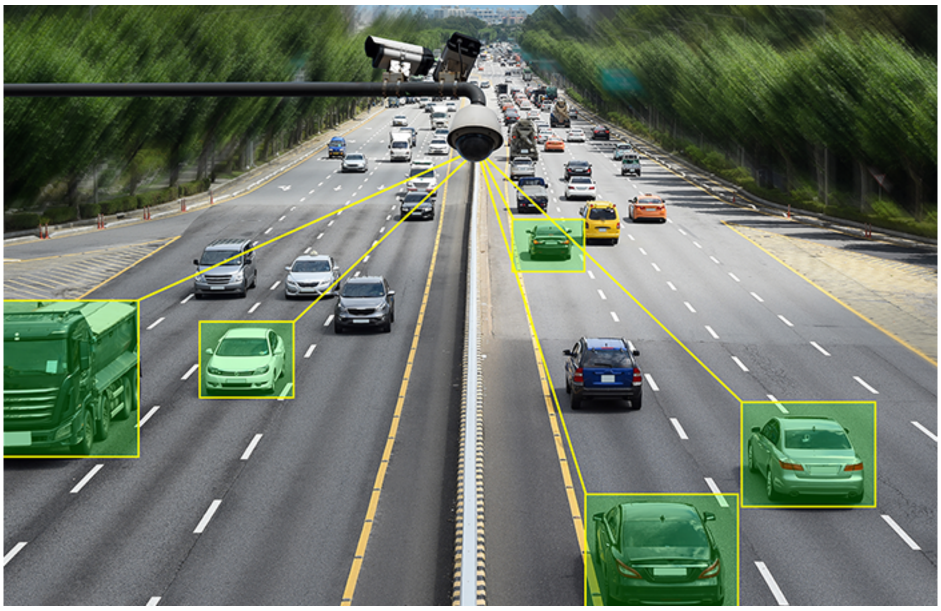


Fig. 2.8: Vehicle Tracking Cam [21]

This is where the *Unsupervised Learning* takes its advantage because *Unsupervised Learning* will label the data based on the similar characteristics, or features into different groups as shown in Fig 2.9 without any human intervention. Clustering is the type of *Unsupervised Learning* algorithm to label the available data set.

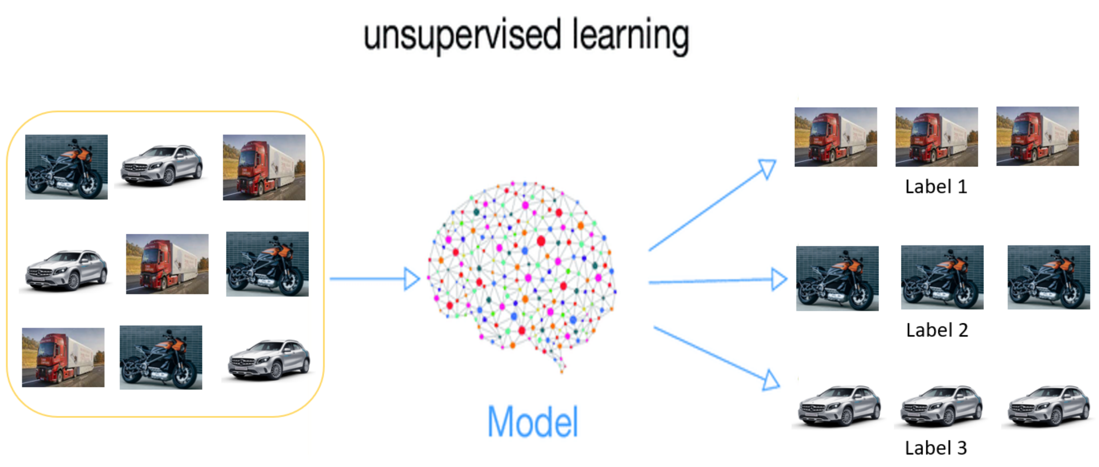


Fig. 2.9: Labelling of Data set using Unsupervised Learning

2.2.3 Clustering

Clustering is dividing data points into several groups so that data points in the same group are similar and dissimilar to other data points in other groups. It is, in essence, a collection of objects arranged according to their similarities and differences [22].

Because clustering is a subjective task, the number of approaches used to achieve this goal is numerous. Every methodology has a unique set of rules for determining the 'similarity' between data points, and each methodology is different. There are over 100 clustering algorithms that have been discovered. The most frequently used clustering algorithms are the following:

- **Connectivity models:** As the name implies, these models are based on the premise that data points that are closer to each other in data space exhibit greater similarity to each other than the data points that are further apart. These models can be built in one of the two ways. In the first approach, they begin by classifying all data points into separate clusters and then aggregating them as the distance between them decreases. In the second approach, all data points are classified as belonging to a single cluster, which is then partitioned as the distance between them increases. Furthermore, the selection of the distance function is entirely subjective. These models are easy to interpret, but they lack the scalability required to handle large datasets. Examples of these models include the hierarchical clustering algorithm and its variants.
- **Centroid models:** These are iterative clustering algorithms in which the notion of similarity is derived from the closeness of a data point to the centroid of the clusters. The *K-Means* clustering algorithm is a popular clustering algorithm that falls into this category. Because the number of clusters required at the end of these models must be specified beforehand, it is critical to have prior knowledge of the data set. These models run iteratively to find the local optima.

- **Distribution models:** These clustering models are based on the notion of how likely it is that all of the data points in a cluster belong to the same distribution as compared to all of the data points outside the cluster (For example: Normal, Gaussian). Overfitting is a common problem with these models. An example of one of these models is the Expectation-Maximization algorithm, which makes use of multivariate normal distributions to achieve the best possible result.
- **Density Models:** These models search the data space for areas with varying densities of data points in the data space. They isolates various density regions and assigns the data points contained within these regions to the same cluster. *DBSCAN* and *OPTICS* are two examples of density models that are frequently used.

Since the data set which is used in this thesis work is unlabeled and also hard to group into a known number of clusters,the Density Model *DBSCAN* is suitable as it creates clusters and labels itself

DBSCAN

Density-Based Spatial Clustering of Applications with Noise (DBSCAN) is a base algorithm for density-based clustering that is used in many applications. It can extract clusters of varying shapes and sizes from a large amount of data that contains noise and outliers, and it can do so in real time [23].

Density-Based Clustering is a concept that belongs to unsupervised learning methods that recognize unique groups/clusters in data. It is based on the idea that a clusters in data space is a contiguous region of high point density that is separated from other such clusters by contiguous regions of low point density and that a cluster in data space is a contiguous region of low point density that is separated from other such clusters.

The *DBSCAN* algorithm uses two parameters:

- **minPts:** The smallest number of points (a threshold) that must be clustered together in order for a region to be classified as dense.
- **eps(ϵ):** It is a distance measure that will be used to locate points that are in close proximity to any given point.

To better understand the described parameters, concepts called Density *Reachability* and *Density Connectivity* are discussed in the following:

- **Reachability:** In terms of density, a point is considered as reachable from another if it is located within a specified distance (eps) to the first.
- **Connectivity:** The transitivity-based chaining-approach, on the other hand, is used to determine whether or not points are located in a specific cluster. For example, where a->b indicates that b is in the vicinity of a, and where N is an outlier 2.10.

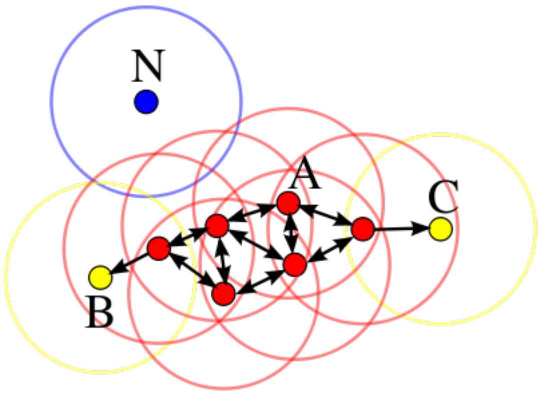


Fig. 2.10: DBSCAN algorithm behavior[23]

There are three types of points after the DBSCAN clustering is complete Fig.[23]:

- **Core:** This is a point that has at least m points within distance n from itself.
- **Border:** This is a point that has at least one Core point at a distance n.
- **Noise:** This is a point that is neither a Core nor a Border and it has less than m points within distance n from itself.

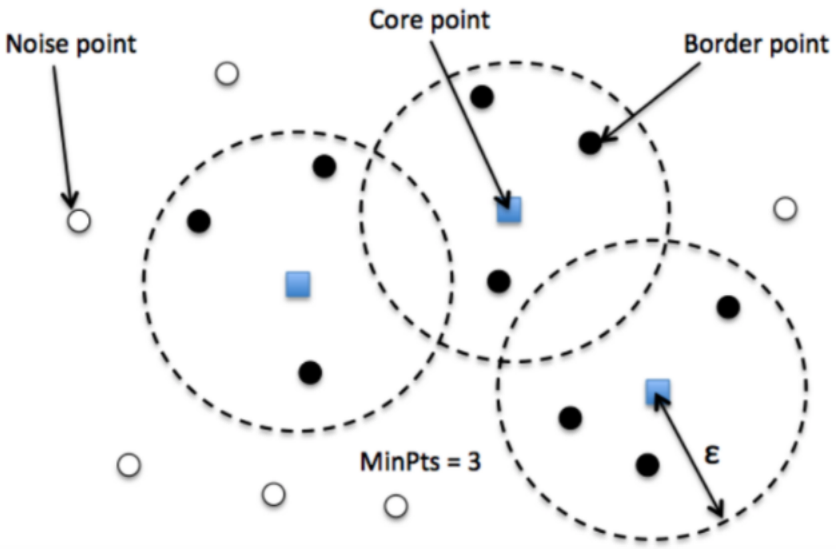


Fig. 2.11: DBSCAN parameters [23]

With this approach, unique clusters of data points will be formed based on the distance between the density of data points.

In the next section, *Supervised Learning* will be explained in more detail, as it is the next step in *Semi-supervised Learning*

2.2.4 Types of Supervised Learning Problems

All the problem that exists under the spectrum of *Supervised Learning* can be broadly categorised into two kinds, namely, Classification problem and Regression problem as shown in Figure. 2.12. In Supervised Learning, the model is trained to approximate a function that either separates data points into classes (Classification) or represents data points to output continuous values (Regression).

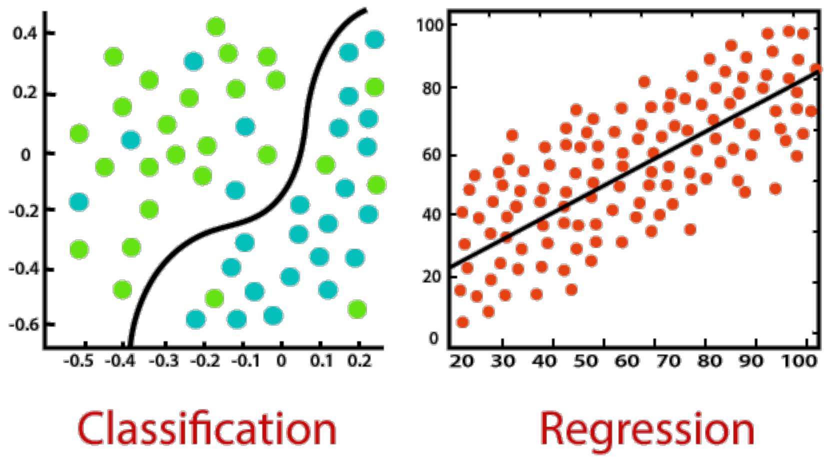


Fig. 2.12: 2-dimensional Classification and Regression problem [24]

Classification problems are further divided into 4 types:

- **1) Binary Classification:** In this type of problem there will only be two classes as possible output. Typically it can be said that there will be one ‘normal’ state and an ‘abnormal’ state. For example, incoming email belongs to ‘inbox’ or ‘spam’ folder.
- **2) Multiclass Classification:** When multiple classes are present, here there is no normal or abnormal state present unlike binary classification. Here labeling of the class will be more than two depending on the data set, the model may predict one among thousand faces or tens of thousand faces in a facial recognition system .
- **3) Multilabel Classification:** These are classification problems where each class has unique label assigned in each instance. For example, in facial recognition system the model has to recognize each individual face by going through the data set containing thousands of facial photos.
- **4) Imbalanced Classification:** Imbalanced classification tasks are those in which the number of examples in each class is unequally distributed among the classes[25].

Before diving into the explanation of the used Machine Learning algorithm, first the basic knowledge of the term ‘training’ should be known.

2.2.5 Training

Until now the word “Training” has come up a lot of time, it is time understand what actually training means with a simple example. Training in Machine Learning is similar to how one learns driving for the first time. At the beginning one will not know how to use brake or accelerator pedal, applying gears and even operating steering wheel. With constant practice one will become a good driver. Once the good driver starts adapting to real world conditions, he becomes an expert driver.Because the driver is trained to drive repetitively. The driving abilities have evolved as a result, the driving will be honed. This whole process is called Training.

To understand mathematically, let us consider a straight line formula,

$$y = m * x + b \tag{2.1}$$

Where:

y = is the value of the line (Output)
m = is the slope of that line
x = input value
c = is the y-intercept

The values that are available for adjusting or “training”, are slope of the line m and y-intercept c. In no other way the position of the line can be affected, as x is the input and y is the output.

$$Weights = \begin{bmatrix} m_{1,1} & m_{1,2} \\ m_{2,1} & m_{2,2} \\ m_{3,1} & m_{3,2} \end{bmatrix} \tag{2.2}$$

$$biases = \begin{bmatrix} c_{1,1} & c_{1,2} \\ c_{2,1} & c_{2,2} \\ c_{3,1} & c_{3,2} \end{bmatrix} \tag{2.3}$$

In Machine Learning, there will be m number of features, that are collected together and formed into a matrix, which is weights in “weight matrix”denoted by **W**. Similarly for c, together is called as “bias matrix” , denote by **b**.

The training process in Machine Learning involves initializing the random values for both W and b. Then attempt predicting the output. For the case where the prediction is poor, the model output is compared with required output and the value of W and b are adjusted for correct prediction. This whole process is known as model prediction as shown in Figure 2.13 .

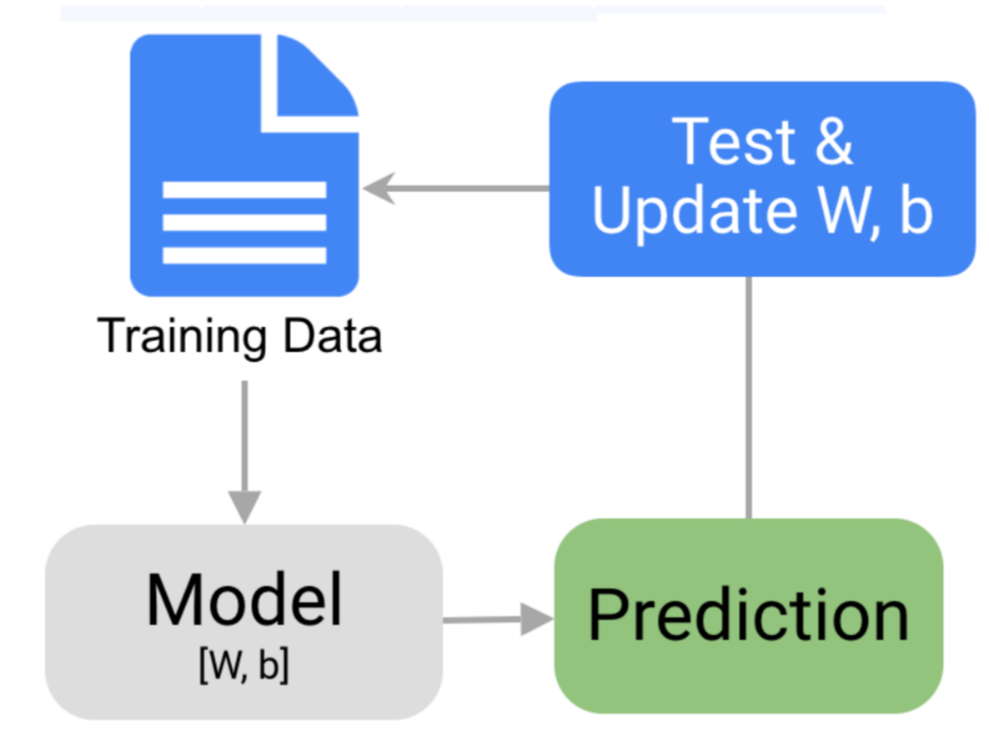


Fig. 2.13: Schematic illustration of model training in Machine Learning [26]

Once the model is trained, it is checked for the accuracy.The presence of overfitting can be detected when the accuracy of the training data is noticeably greater than that of the validation data. As illustrated on the right in Figure. 2.14, the algorithm maps the scatter caused by outliers in the training data.The model memorizes the training dataset and therefore does not generalize on unseen data. Due to this, the model's ability to generalize will be lessened. The opposite of overfitting is underfitting, which occurs when the data is insufficiently described by the model.

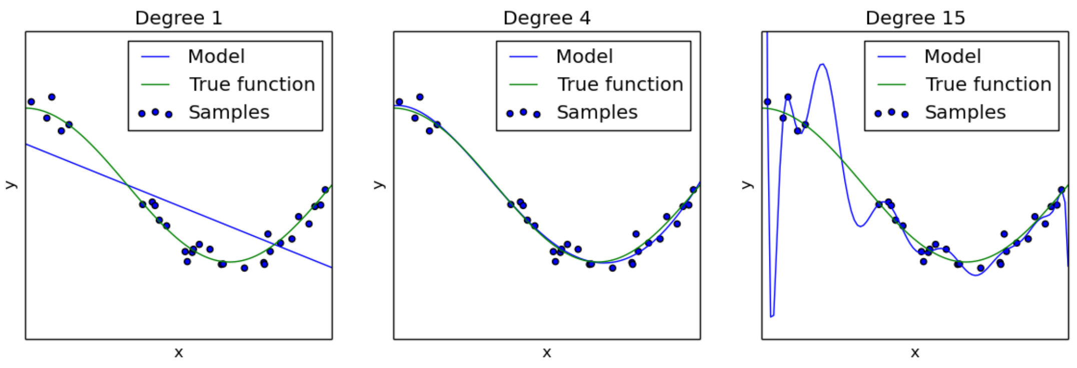


Fig. 2.14: Schematic illustration of underfitting (left), overfitting(right) and a robust model (center) for the mapping of a polynomial function [27]

2.2.6 Cost-Function

A metric (Cost Function) is used to determine the quality of the prediction during training. The difference between the predicted and actual values is used to calculate the error of the prediction. The model parameters are adjusted based on the size of the error. The cost function that is used is determined by the type of the problem.

The cross-entropy loss, also known as log loss, is a popular metric for supervised learning classifications.This metric evaluates the performance of a model whose output ranges from 0 to 1. The cost function for an expected true value of 1 is shown in Figure. 2.15. The log loss increases as the prediction gets closer to zero. This means that predictions that are incorrect and for which the algorithm is certain are penalized more heavily.

The log loss for binary classification will be calculated using the following formula:

$$L = -(y * \ln(p) + (1 - y) * \ln(1 - p)) \tag{2.4}$$

Where:

L = Log Loss
y = Binary indicator (0 or 1) whether class label c is correct for observation o
p = Predicted probability that observation o = class label c

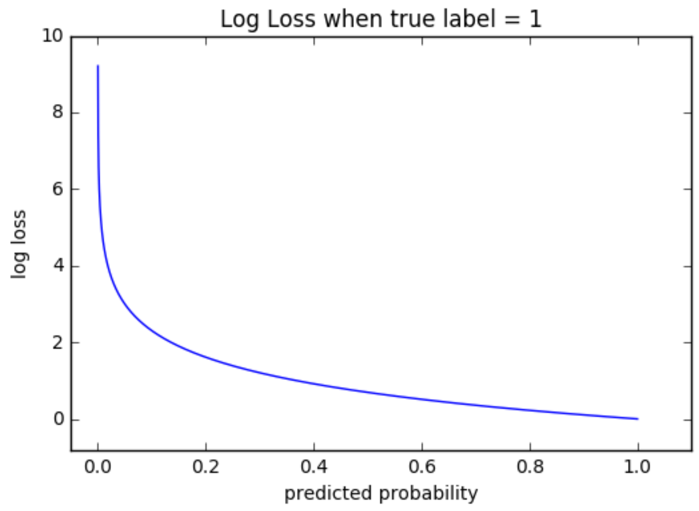


Fig. 2.15: The Log Loss metric for a function with an expected label of 1 [28]

If there are more than two labels (M>2) the following formula will be used:

$$L = -\sum_{c=1}^{M} y_{o,c} * \ln(p_{o,c}) \tag{2.5}$$

Where:

M = Number of classes
L = Log Loss
y = Binary indicator (0 or 1) whether class label c is correct for observation o
p = Predicted probability that observation o = class label c

2.2.7 Machine Learning Algorithms

In the field of Machine learning, there are numerous algorithms. The problem presented in this thesis is a multiclass-multilabel classification problem which belongs to *Supervised Learning*. In this section the most commonly used *Supervised learning* algorithm are presented.

Decision Tress will be explained briefly, before looking into the ensemble techniques like *Random Forest* and *Boosting algorithms.*

Decision Trees
When it comes to classification problems, *decision trees* are frequently used. In this algorithm, a hierarchy of if/else questions is generated, which ultimately leads to a decision. The goal is to get to the desired result with as few questions as possible. The first node is defined as the variable with the greatest amount of information gain. Each of the possible answers leads to the next question, which in turn provides the greatest amount of additional information gain. The term *"Leaf"* refers to the point at which a decision path through the *Decision Tree* comes to an end [29].

An example of a *Decision Tree* model is shown in Figure. 2.16. It is to distinguishes between four animals: bear, hawk, penguin, and dolphin. At first, it is asked if the animal has feathers. This bisects the decision space. Depending on the answer, the question "Can it fly?" or "Does it have fins?" is asked and a final result is found.

When creating the *Decision Tree*, the arbitrary combination possibilities of decision rules will lead to the creation of all branches to the end (*Leaf*). are created. This leads to a complex model and heavy *overfitting. Pre-Pruning* or Post-Pruning can be used to counteract this effect. With *pre-pruning*, the *decision tree* is constrained either by a maximum possible number of leaves or by a minimum number of data points per branch, depending on the case in question. *'Post-Pruning'* refers to an operation in which the *Decision Tree* is not constrained until after it has been created, which is described in detail below [30].

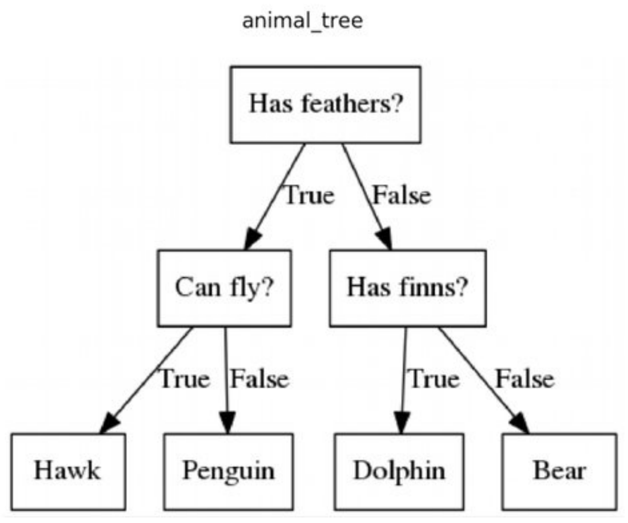


Fig. 2.16: Schematic representation of Decision Tree [30]

Decision Trees have two distinct advantages over other methods for decision making. The models are straightforward to visualize and comprehend. It is not necessary to scale the data, as it is with other algorithms. Due to small datasets the model will have low accuracy, which is one of the disadvantages. Deterministic *decision trees*, despite *pre-pruning*, are prone to *overfitting* and low generalizability.

Ensemble Learning

When complex questions are posed to random people, the aggregated answers usually outperform the expert solution. This is called the *wisdom of truth*. Similarly, the aggregated predictions group of predictors often get better predictions when compared to a single predictor is known as *Ensemble learning* [20].

Random Forest

Ensemble methods, which combine multiple Machine Learning models to achieve higher accuracy, are becoming more popular. Random Forest is an example of an algorithm that trains various decision trees (see Figure. 2.17). A single decision tree has a tendency to have the effect of overfitting for a portion of the data it is used to predict. By taking the results of multiple decision trees and averaging them, these effects are reduced. Random Forest is one of the most widely used machine learning algorithms as it combines the advantages of the decision tree with the advantages of Ensemble Learning techniques while minimizing the disadvantages of the Decision Tree.

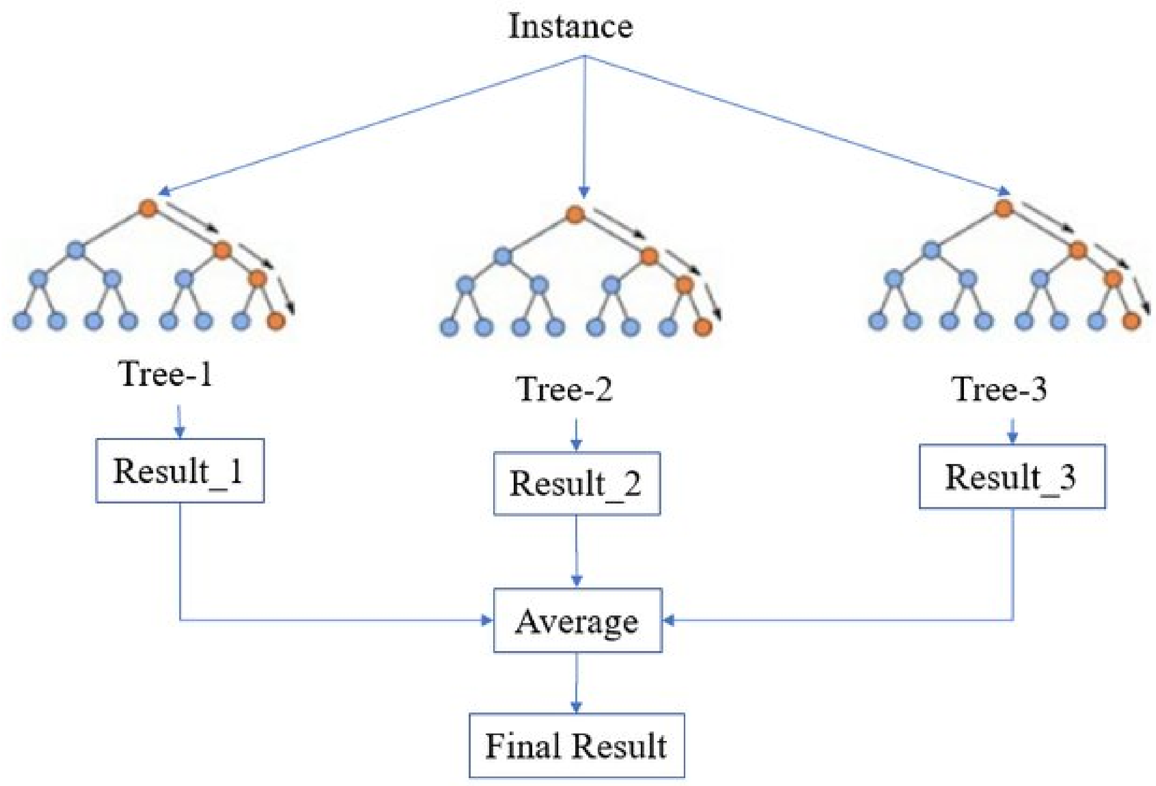


Fig. 2.17: Schematic representation of a Random Forest Model [31]

Gradient Boosting

In Machine learning, boosting is a technique for putting together an ensemble of data. It begins by fitting an initial model to the data (for example, a decision tree Figure.2.17). Then, a second model is developed with the goal of accurately predicting the cases in which the first model fails. It is anticipated that the combination of these two models will perform better than either model alone. The process of boosting is iteratively in a loop. Each successive model attempts to correct for the shortcomings of the combined boosted ensemble of all previous models.

Gradient Boosting is a type of boosting that is used to improve performance weak learners. It is based on the intuition of sequentially adding the predictors to an ensemble, to correct the previous predictor. This method tries to fit the predictor to the residual error [20], residuals are the losses incurred and will be calculated after each model predictions:

- **a)** The next target outcome of a case is high if a small change in the prediction for the case results in a large drop in error. Predictions from the new model that are close to their targets will help to reduce the error.
- **b)** If a small change in the prediction for a case results in no change in the error, then the next target outcome for the case is zero. Changing this prediction has no effect on the error.

Gradient Boosting is given its name because the target outcomes as shown in Figure.2.18. Each case is determined based on the gradient of the error relative to the prediction. Each new model takes a step in the direction of the least amount of prediction error possible in the space of possible predictions for each training case, which is called the prediction space [32].

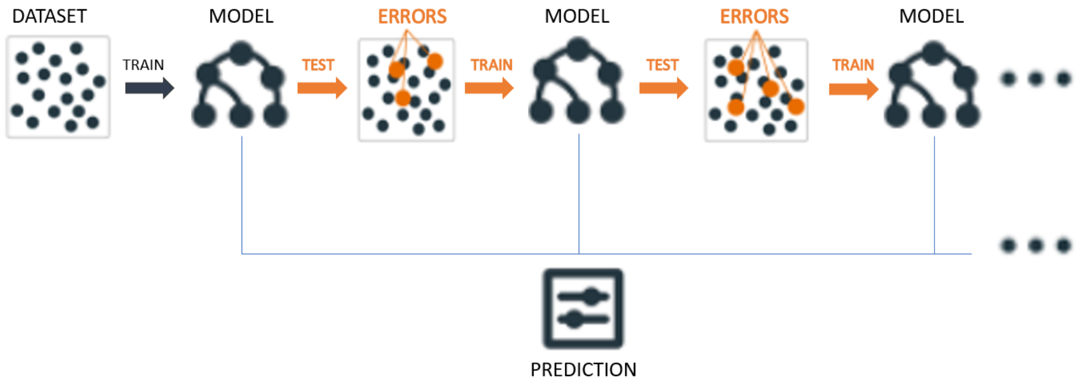


Fig. 2.18: Schematic representation of Gradient Boosting [33]

Though *Gradient Boosting* performs well on the unprocessed data and provides great flexibility that is, it can optimize on different loss function which makes the function fit very flexible. The disadvantage of using *Gradient Boosting* is, that it is computationally expensive and requires more than thousand (>1000) decision trees, which eventually leads to memory problem [33]. To minimize the disadvantage of *Gradient Boosting* various forms of algorithm have been derived.

XGBoost

XGBoost (eXtreme Gradient Boosting) is a scalable and improved version of the *Gradient Boosting Algorithm* designed for efficacy, computational speed, and model performance. It is an open-source library that is a part of the Distributed Machine Learning Community. *XGBoost* is a unique combination of software and hardware capabilities that is designed to improve the accuracy of existing boosting techniques in the shortest amount of time.

Each tree model in XGBoost strives to reduce the residual, as shown in Figure.2.19. The traditional GBDT only employs the first derivative of error information in its calculations. XGBoost performs the second-order Taylor expansion of the cost function, and it makes use of both the first and second derivatives of the cost function which results in efficient training of the model. In addition, the XGBoost tool offers a cost function that can be customized [34].

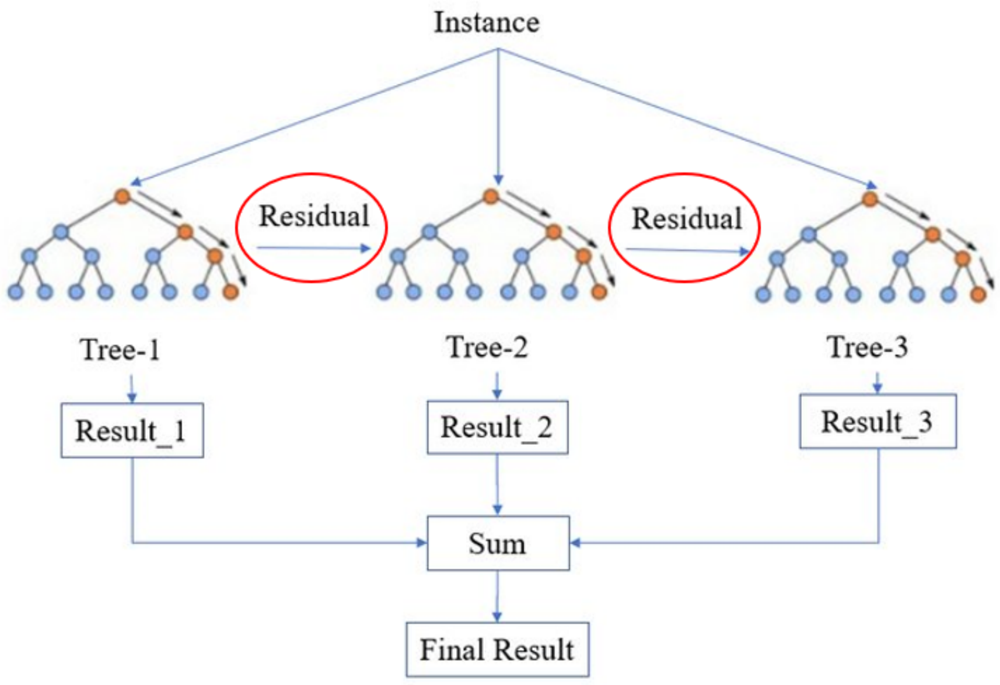


Fig. 2.19: Schematic representation of XGBoost [34]

LightGBM

LightGBM is the short form for *Light Gradient Boosting*, which uses *Gradient Boosting* in an open source library. It provides an efficient and effective implementations of the same.

The difference in *LightGBM* with other tree-based models is that the framework makes use of a leaf-wise tree growth algorithm, which is opposite to many other tree-based algorithms, which make use of depth-wise tree growth algorithms. Leaf-wise tree growth algorithms tend to converge more quickly than depth-wise tree growth algorithms towards the solution as shown in Figure.2.20. They are, however more prone to *Overfitting* than other gradient boosting models.

Whenever the categorical features are present in the dataset, categorical features has to be encoded into numerical form as the machine can only understand the data in numerical form. One-hot encoding one of the method that is used for encoding categorical features. If the dataset is large like the dataset which is used in this thesis work, encoding the categorical features is a tedious task and memory consuming.

The one-hot encoded features used in tree-based model, trees constructed will be grown in-depth and creates an imbalance in the model. This is where the advantage of *LightGBM* comes into use because this algorithm handles categorical data as it is without any need for encoding, through which a good accuracy will be achieved. [35].

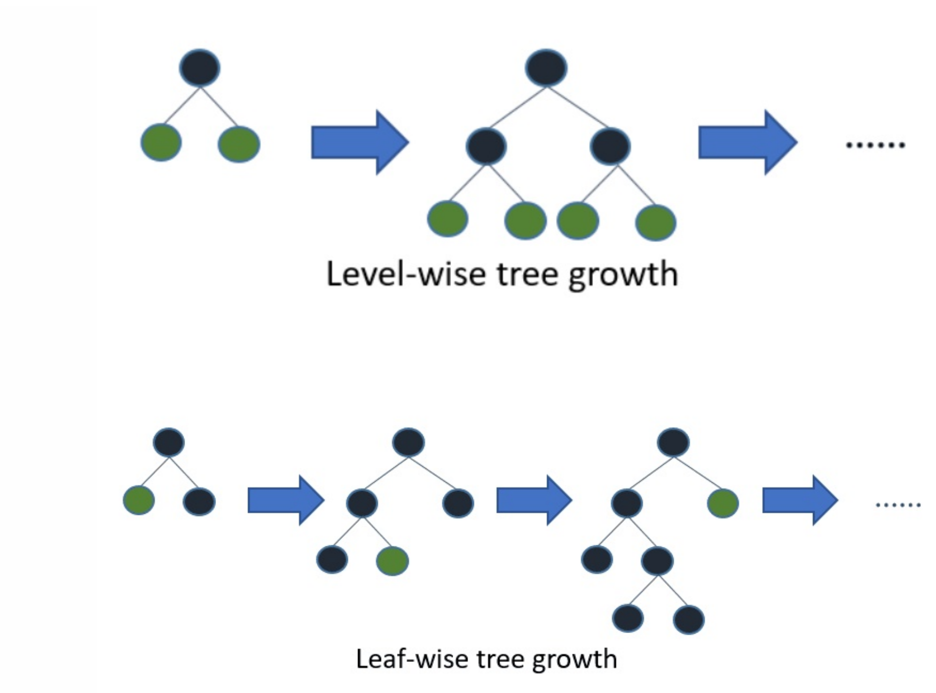


Fig. 2.20: Schematic representation of LightBGM [35]

2.2.8 Evaluation Criteria

In order to validate the trained Machine Learning models, they must be evaluated based on the validation dataset. The *Confusion Matrix* serves as the foundation for all of the different evaluation metrics for classification algorithms (truth matrix). The model predictions are compared to the actual values and added to the Confusion Matrix as True Positives, True Negatives, False Positives and False Negatives, depending on the actual value as well as the correctness of the prediction as shown in Figure 2.21.

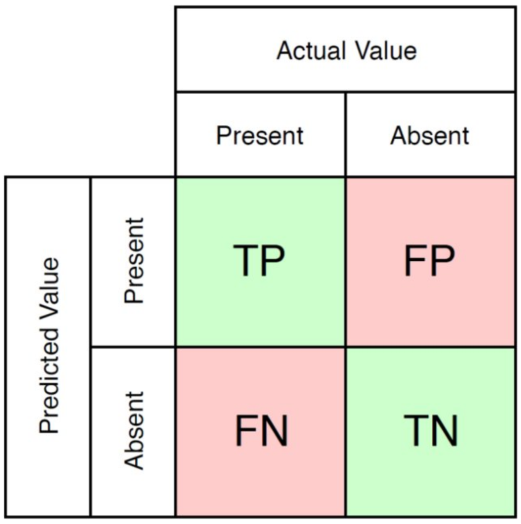


Fig. 2.21: Schematic representation of Confusion Matrix.[36]

A visit to the car mechanic and the mechanic's assessment of whether or not the car is working serves as an illustration of this point. The mechanic's opinion is either positive (classification as "working") or negative (classification as "not working"). As a result, there are four possible outcomes:

1. **True Positive**(TP): The car is not working and the mechanic is right in his decision
2. **False Positive**(FP): The car is not working and the mechanic has incorrectly classified it as working(Type I error)
3. **True Negative**(TN): The car is working and the mechanic is right in his decision
4. **False Negative**(FN): The car is working but the mechanic has incorrectly classified it as working(Type II error)

Precision

Precision is the ratio between the *True Positive(TP)* and all the positives. In the example, that would be identifying the car that is not working out of all cars working, mathematically it is given by the following equation 2.6:

$$Precision = \frac{TP}{TP + FP} \tag{2.6}$$

Recall

The recall is the measure of identifying the *True Positive(TP)*. Thus, for the cars which are actually not working, recall gives the information on how many cars are correctly identified as not working. Mathematically is it is given by the equation 2.8:

$$Precision = \frac{TP}{TP + FN} \tag{2.7}$$

Accuracy

The easiest and most popular metric for the evaluation criteria is to understand *Accuracy*. *Accuracy* is the ratio of the total number of correct predictions and the total number of predictions.

$$Precision = \frac{TP + TN}{TP + FP + TN + FN} \tag{2.8}$$

Intuitively, using accuracy as an evaluation metric makes sense because the distribution of values is balanced. The meaning of balanced here is that is an equal number of positives and negatives values are present. But in general, it is always recommended to check precision and recall.

In some scenarios where the achieved accuracy might be very high, but the precision or recall might be low, for the example mentioned earlier, one would like to avoid any situation where the car is not working, but the model classifies as working i.e. high recall. On the other hand, where the car is working, and the model predicts as not working, the mechanic has to avoid repairing the working car at any cost.

Similarly, while aiming for high precision to avoid the wrong car to be repaired, but the mechanic gets a lot of cars that are working to be repaired

It is always recommended to have high precision and high recall for trained model, for the accurate model. But ideally achieving the above is impossible as there will be always a trade off between the precision and recall when the dataset is imbalanced.

This is the main disadvantage of using accuracy as evaluation metric. Hence in this case one the best possible evaluation metric is *F1 Score*,which is explained in next section

F1 Score

So, while understanding the accuracy, one has to deduce that there should be trade-off between precision and recall. So F1 score one which considers Harmonic mean of precision and recall. The harmonic mean balance the trade-off between the precision and recall

F1 scores can be represented in the *Confusion matrix*.The F-Measure method provides a single score that takes into account both the concerns of precision and recall in a single numerical value.

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{2.9}$$

In order to achieve a good F1 score, one must have a low number of false positives and false negatives, which means you are correctly identifying real anomalies and are not disturbed by false alarms. When the F1 score is 1, the model is considered perfect; when the score is 0, the model is considered a complete failure [37].

2.3 The potential Machine Learning model for Pre-Tactical Phase In ATFCM

Due to the nature of high system complexity, aviation industry generates a large amount of data. Since it is difficult to find the underlying pattern present in the huge complex data manually, the aviation industry is an exciting area to implement Machine Learning solutions. The aviation industry is highly affected by COVID-19, and expert has predicted the traffic will not be returning to the level of 2019 until 2024 [38]. Despite the challenging phase happening in aviation, there has been a gradual increase in innovation in the aviation industry has been observed, especially in developing Machine Learning and Artificial Intelligence solutions [39]. Also, as the report form, James Nurton et al. [40] highlights, there is an increase of around 67% in the Machine Learning patents related to aviation between 2013-2016.

During these unprecedented times, the aviation industry has seamlessly made efforts to make stress-free travel to ensure COVID rules like disinfecting, providing protective gear to prevent the spread, and operating with limited seats to maintain social distancing. This has made the industry think to develop data-driven and more innovative solutions and approaches. Post-COVID, it is expected that there would be lots of changes in aviation industry. For example, in the field of area and fleet management operations, Machine Learning could predict the dynamic pricing of airline tickets based on the demands and delays [41]. Also, the second-largest share in terms of size and cost in the aviation industry is crew management. Predictive models can be used to predict the fatigue and the stress level on the crew due to constant change in time zone [41].

Other applications like safety and airplane maintenance monitor the health, which would help reduce the maintenance cost and run safer airlines. Also, due to COVID and social distancing measures, thermal imaging and temperature measurement are in place in airports. This gives lots of scopes to apply Machine Learning and data-driven approaches in the field of image processing and Computer Vision [42]. The other fields in which Machine Learning could be used are shown in Fig. 2.22.

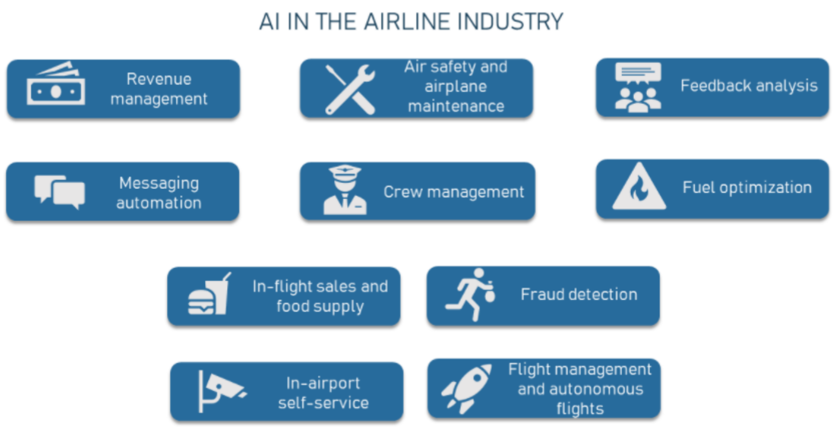

AI IN THE AIRLINE INDUSTRY
Revenue management
Air safety and airplane maintenance
Feedback analysis
Messaging automation
Crew management
Fuel optimization
In-flight sales and food supply
Fraud detection
In-airport self-service
Flight management and autonomous flights

Fig. 2.22: Application of Machine Learning in Aviation [41]

ATFM is an up-and-coming area for optimization because it contains a high proportion of decision support applications. As the data becomes large and highly complex, the decision-making becomes tougher. Hence Machine Learning has an enormous potential to improve the efficiency of ATFM. In the following section, the most relevant work in the fields of Machine Learning applied to ATFM will be discussed.

According to [39], there has not been an ATFM infrastructure that optimizes the trajectories efficiently. The optimized predictions trajectories would lead to more efficient planning of the route. As a result, there will be less congestion and fewer ATFM measures. This in turn will help in the reduction of carbon emission. Although there are many research which has focused on predicting the trajectories in tactical phase [43],[44], [45], pre-tactical phase has received less attention.

Even though the paper presented by Rasoul Sanaei et al. [46] discuss the prediction of flights delays using *Deep Convolution Neural Network*(DCNN) algorithm in the tactical phase. The paper highlights the dynamic behaviors present in the pre-tactical phase.

The pre-tactical forecast is so dynamic because the prediction is made using something called PREDICT software. The forecast is dynamic because the decision made by PREDICT is not taken by learning from the pattern present in the available data. Instead, the decision is purely based on the flow process. The lack of clarity regarding the implementation of PREDICT even though it is mentioned in the EUROCONTROL operation manual [47]. Due to this, there is a potential for improvement to predict the trajectories in pre-tactical phase.

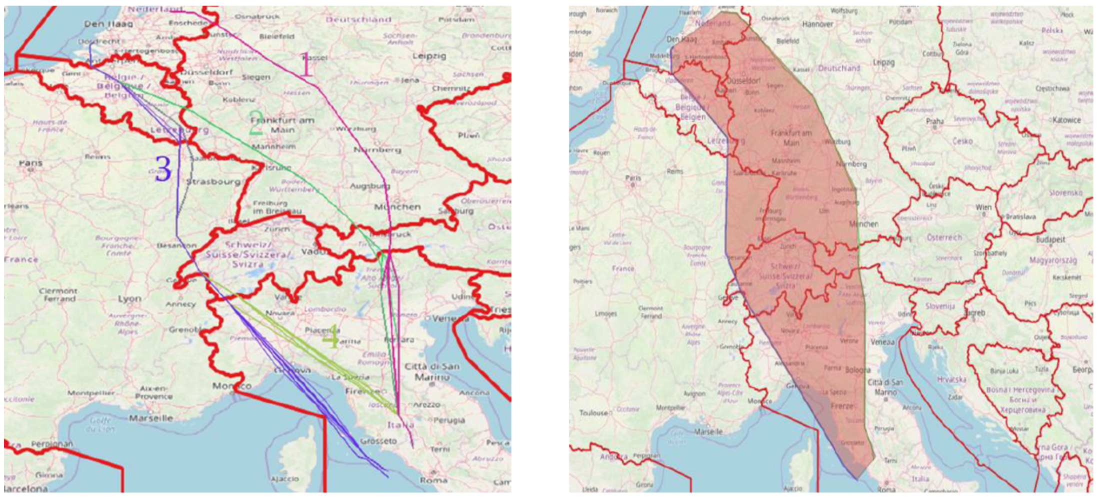


Fig. 2.23: Example for creating Clusters on area between routes. Trajectory Clusters between Rome - Amsterdam (right). Calculation of the area between two routes(left) [4]

The one paper discussing using Machine Learning applications in the pre-tactical prediction is present by Manuel Mateaos et al.[4] Due to the dynamic behavior and imbalance in the data present in the pre-tactical phase, the author has made use of the area between two routes to define the cluster, as shown in Fig 2.23. The prediction of the planned trajectory is carried out by making use of clusters formed for each OD pairs, day and time of flight, airline, and type of aircraft used.

Since there is a lack of attention on predicting trajectories in the pre-tactical phase, which could help optimize the planning of the route and balance demand and capacity. This thesis work addresses this issue, and the goal is to predict the planned trajectories in the pre-tactical phase of flight operation. Furthermore, the PREDICT logic used by EUROCONTROL will be implemented and used as a benchmark for the created Machine Learning model. The following chapter explains the methodology used in this thesis work.

3. Methodology

As mentioned in previous chapter 2.3 the problem, this thesis work intended to solve is, predicting the planned trajectories in pre-tactical phase. In order to make the prediction more accurate and help in ATFM decision, the tasks are divided into three parts:

1) Merging Sabre ADI data with the EUROCONTROL RnD database
2) Cluster similar planned trajectories for each *Origin-Destination*(OD) pair
3) Develop a methodology for predicting the the created cluster of similar planned trajectories

To understand the tasks mentioned, in the following sections each of tasks will be explained individually.

3.1 Database

The Basis of this thesis work is from the *EUROCONTROL RnD Data* and *Sabre Airport Market Intelligence* for three months, June, August, and November of the year 2016. The detailed insights into the dataset will be explained in the next section.

3.1.1 EUROCONTROL

The RnD dataset which is released by the *EUROCONTROL* for the research purposes are taken. The planned trajectories represented in the RnD dataset are based on the *Flight Plan* (FPL) which is usually available only after the pre-tactical phase. In this thesis work the clustered planned trajectories are expected prediction for the developed models. The most important **13** datafields used from the RnD dataset are shown in Table 3.1. The dataset includes the information of **2,453,627** flights for the mentioned three months. Detailed information on *EUROCONTROL RnD* data will be provided in appendix.

The parameter that is used for the flight include :- unique identifier for each flight(1), departure airport(2), corresponding latitude and longitude information for departure airport(3)(4) , arrival airport(5) and latitude and longitude of arrival airport (6)(7). The *Filed OFF BLOCK Time* is the information planned departure time(8) and *Filed Arrival Time* is for planned arrival time(9). Additional information regarding the type of aircraft(10), operator(11) and type of flight(13).

Nr.	Flight Data	Meaning
1	ECTRL ID	Unique numeric identifier of each flight
2	ADEP	ICAO code for departure airport
3	ADEP Latitude	Departure airport latitude
4	ADEP Longitude	Departure airport longitude
5	ADES	ICAO code for arrival airport
6	ADES Latitude	Arrival airport latitude
7	ADES Longitude	Arrival airport longitude
8	FILED OFF BLOCK TIME	Planned departure time
9	FILED ARRIVAL TIME	Planned arrival time
10	AC Type	ICAO code for Aircraft type
11	AC Operator	ICAO code for Aircraft operator
12	AC Registration	Aircraft registration
13	ICAO Flight Type	Type of flight

Table 3.1: EUROCONTROL Flight Data [48]

3.1.2 Sabre Airport Data Intelligence

Sabre Airport Data is market intelligence which provides planned flight information based on the *Flight Intentions* (FI) that are usually available before the pre-tactical phase. The dataset obtained for this thesis work consist of 12 flight-specific datatypes which looks as show in Table 3.2.The dataset includes the information of **8,738,841** flights for the mentioned three months. Detailed information on *Sabre Airport Data* is provided in appendix.

Similar to *EUROCONTROL Sabre Airport Data* has ATM-specific flight-parameters, but the representation is in different format as shown in Table 3.2. The number of flight information is large compared *EUROCONTROL*, due to the fact that the dataset obtained has information from all over the world. Before using the data to build *Machine Learning*, the data should be cleaned and prepared properly. A detailed explanation will be given in the next section on data preparation.

Nr.	Sabre Airport data	Meaning
1	flightname	Unique numeric identifier of each flight
2	Origin	IATA code for departure airport
3	Destination	IATA code for arrival airport
4	Airline	IATA code for Aircraft operator
5	DayOfDeparture	Planned Day of flight departure
6	DayOfArrival	Planned Day of flight arrival
7	TimeOfDeparture	Planned Time of flight departure
8	TimeOfArrival	Planned Time of flight departure
9	Equipment	IATA code for Aircraft type

Table 3.2: Sabre Airport Data [49]

3.2 Data Preparation and Parametrization

In order to use both datasets in the training process, they first need to be merged together. *EUROCONTROL* represents the flight data for European airspace, whereas the *Sabre Airport Data* contains the flight data from allover the world as mentioned in section 3.1.2. This thesis work is concentrated on solving problems in European airspace, the *Sabre Airport Data* should be represented in the form of *EUROCONTROL* and then combine both store as Bigdata. After analyzing both the dataset from *EUROCONTROL* and *Sabre Airport Data*. It is evident there is mismatch between dataset , the representation of flight data for same flight is not in the same format. Hence, in order to combine the data the following steps should be taken as shown in Fig 3.1 as follows:

1) The data present in the *Sabre Airport Data* contains the data in *IATA* code. In order to merge the datasets, the *IATA* codes should be converted into *ICAO* codes,which are present in the RnD dataset.

2) The conversion from *IATA* to ICAO codes in the Sabre Airport Data is carried out using a translation file containing the codes.

3) Finally to combine the data, for each the OD pair same airline and aircraft are filtered between both the dataset for the closest departure time. The reason for choosing the closest flight is because departure time differed between the two datasets for specific flights.

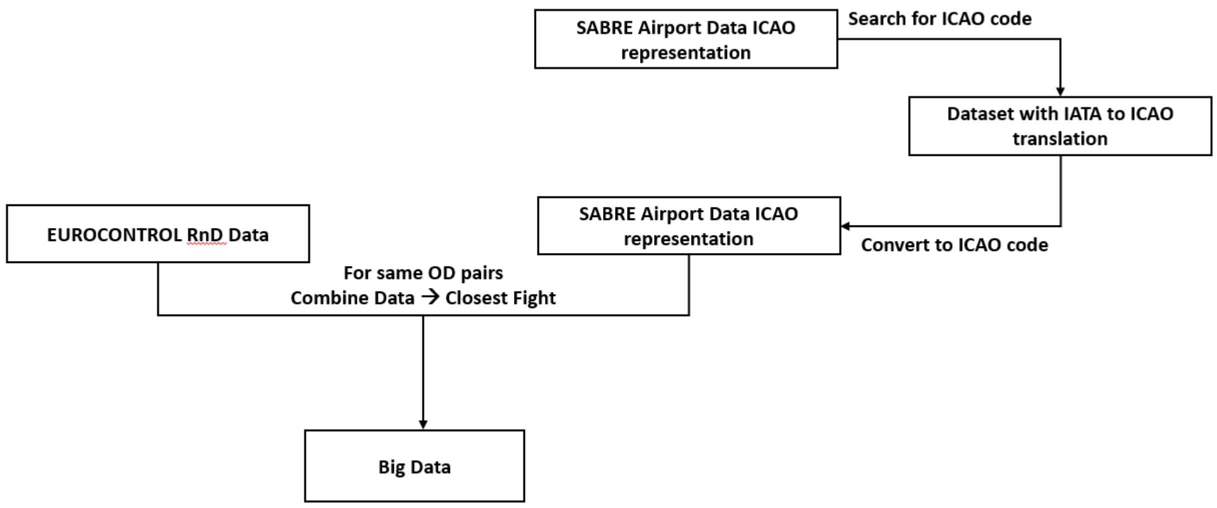

SABRE Airport Data ICAO representation
Search for ICAO code
Dataset with IATA to ICAO translation
EUROCONTROL RnD Data
SABRE Airport Data ICAO representation
Convert to ICAO code
For same OD pairs
Combine Data → Closest Fight
Big Data

Fig. 3.1: Block Diagram representing data merging process

Table 3.3. This table represents the chosen parameter to determine the trajectories for each OD pair.

Before building the *machine learning* model, the available dataset should be split into training data and validation data. Usually in practice 20% of the dataset is considered as validation dataset. The model is trained on the training data. The validation dataset is transformed according to training parameters, the validation

Nr.	Flight Data	example
1	ECTRL ID	197151726
2	ADEP	EDDP
3	ADEP Latitude	51.42389
4	ADEP Longitude	12.23639
5	ADES	LIME
6	ADES Latitude	45.66889
7	ADES Longitude	9.70028
8	FILED OFF BLOCK TIME	01-06-2016 00:00:00
9	FILED ARRIVAL TIME	01-06-2016 01:17:17
10	AC Type	A306
11	AC Operator	BCS

Table 3.3: Trajectory parameters for EUROCONTROL RnD

dataset has paramters similar to training dataset. This prevents information from the validation data from flowing into the training set and the model from already receiving it during training.

In Machine Learning, it is common to use a proportion of randomly chosen data entries as the validation set. The percentage used is an established value in the ML domain and serves a meaningful validation. The following chapter describes the data pre-processing using *Unsupervised Learning* in more detail.

3.3 Data Pre-Processing Using Unsupervised Learning

After the data preparation is completed and data is structured as per the requirement the next step is to pre-process the prepared data. The meaning of pre-processing of data is to transform the dataset which can be used by ML algorithms. In order to pre-process, one should explore the data to check for co-relation between the chosen parameters which is also known as *Exploratory Data Analysis*(EDA). After performing the EDA on the entire dataset similar to Table 3.3, it was found that there was no output parameter which can be used to predict **trajectory** using Machine Learning model. By the definition presented in section 2.2.2, it can be treated as an *Unsupervised Learning* problem.

As mentioned in section 2.2.2 one of the way to solve *Unsupervised Learning* problem is by using *clustering algorithms*. First and foremost thing when using most of the *clustering algorithms* is to define how many clusters that needs to formed ?. It is hard to determine the number of clusters that needs to be formed for the dataset which is being used in this thesis. Because, the trajectory prediction uses 4D input parameters namely OD pairs, latitude and longitude. There were around **16,304** OD pairs combinations present. The method used in this thesis work requires the trajectories to be predicted to be grouped by OD pairs. On summarizing the input, it has 4D parameters which is termed as multi-class as mentioned in section 2.2.4. The output parameter will have multiple labels to predict which is termed as multi-label problem mentioned in section 2.2.4.

The best choice of *clustering algorithms* in cases where the number of cluster is hard to determine manually would be *Density Based Spatial Clustering Algorithm with Noise*(DBSCAN) mentioned in section 2.2.3. This algorithm intelligently forms the clusters and labels automatically.

Before passing the dataset into the DBSCAN algorithm, for each flight trajectories points are required. This data is extracted from *EUROCONTROL* database knows as *Flights Points data*(FIR) as shown in Table 3.4. As the prepared dataset consists only the flights for European space, every flight present in the prepared dataset can be matched with unique ID present the FIR data to get the trajectory points for each flight.

Nr.	**Flight Data**	**Meaning**
1	ECTRL ID	Unique numeric identifier of each flight
2	Sequence Number	Numeric sequence number of the points crossed by the flight in chronological order
3	Time Over	Time (UTC) at which the point was crossed
4	Flight Level	Altitude in flight levels
5	Latitude	Latitude points on the route trajectory
6	Longitude	Longitude points on route trajectory

Table 3.4: EUROCONTROL Flight Points [48]

The dataset is grouped by OD pairs, that contains the set of planned trajectory points (latitude and longitude) and used as the input to DBSCAN algorithm to form the corresponding clusters automatically. By default the DBSCAN algorithm uses *Euclidean Distance* to calculate the distance between each datapoint. But since the trajectories consist of multiple datapoints a custom distance metric is used.

The area between two routes has been tested as a distance metric as an alternative to the traditional route distance metric as shown in Figure3.2. Where (A) is the origin airport and (B) is the destination airport. Figure3.2 show area calculation between two routes. For this particular OD pairs there were 1045 planned routes and area is calculated between each combination of routes. The main shortcoming of this approach was that the observed distance grows with the great circle distance as discovered by Manuel Mateos et. al. [4].

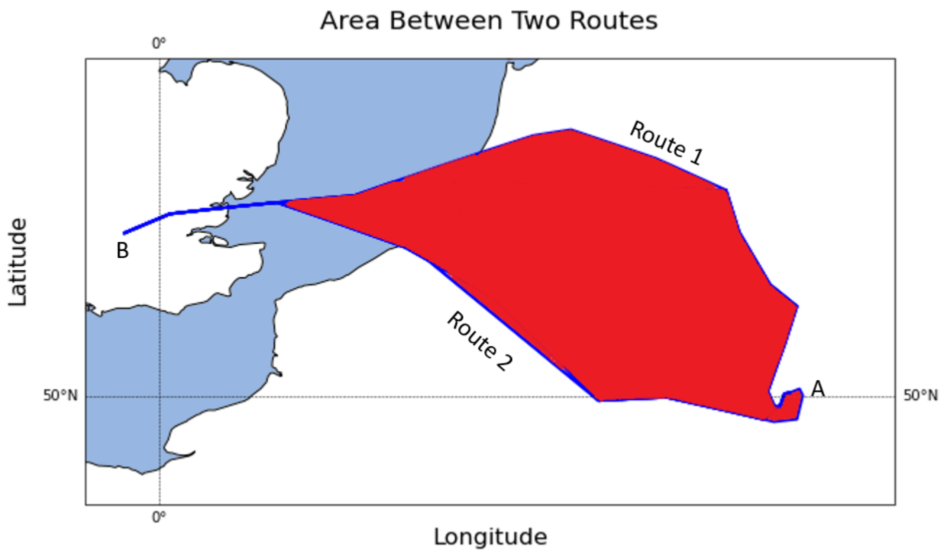


Fig. 3.2: Schematic representation of calculating area between two routes

This shortcomings is solved by normalized area between the two routes that have been taken into consideration as mentioned in [4]. The normalized area is calculated using the next equation:

$$route_{dist} = (\frac{total\_route_{dist}}{od\_pair_{dist}})^{\frac{3}{2}} \tag{3.1}$$

Where:

$total\_route_{dist}$ = area between two routes
od_pair_{dist} = haversine distance between the origin and destination airports

Clustering using DBSCAN was carried out using custom distance metric by making use of above equation.3.1. A minimum of 10 routes per cluster has been chosen, in order to avoid the creation of small clusters from which the Machine Learning algorithms cannot learn. Example clusters for the OD pair for Figure 3.2 is show in Figure 3.3.

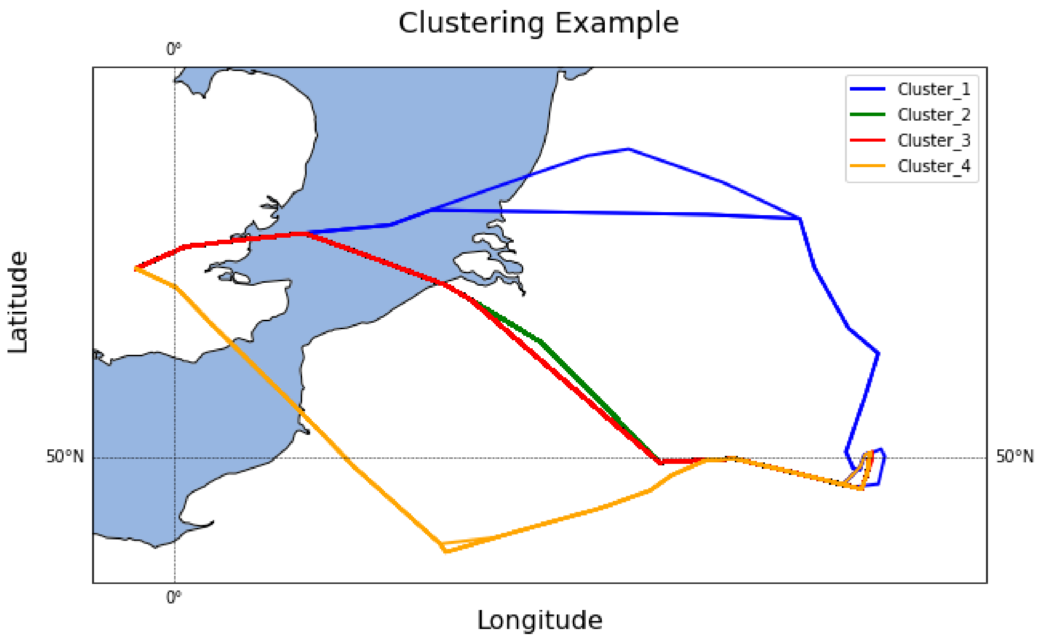


Fig. 3.3: An example for clustering using DBSCAN algorithm

As mentioned in Section (3.2), the dataset was split into training and validation dataset. The clustering will be carried out on the training dataset and then validation dataset will be transformed based on the determined clusters in the training dataset. This process involves checking for the trajectories in each cluster with the lowest mean area too all other trajectories for the specific cluster. Then labels from the training dataset is used to transform the validation dataset. In total there have been 21,199 clusters determined.

Once the cluster labels are created, this becomes the output parameter for predicting trajectories. The prediction can be done using *Supervised Learning*, as the dataset contains both input and output parameters.

3.4 Machine Learning Model

This chapter presents the Machine Learning model used for predicting the trajectories, and validating the model with Predict Logic

3.4.1 Supervised Machine Learning

To predict planned trajectories based on Flight Intentions. for the classification and prediction in this thesis work *Ensemble Learning* techniques like *Random Forest*, *XGBoost* and *Lightgbm* are selected. The reason for choosing the *Ensemble Learning* over popular *Neural Network*(NN) models, in general reduces the variance in prediction and also by definition from chapter 2.2.7 combines results from multiple model which leads to reduced generalization error [50]. Before passing the input parameters into the machine learning models, all the non-numerical which is also known as categorical data have to be encoded into numerical form. This is mainly because of the fact that the Machine Learning models can only understand the data in numerical format.

In order to compare with PREDICT Logic prediction, similar input parameters are chosen for all the above mentioned Machine Learning models and are show in Table 3.5. Except for the planned departure time, all the input parameters are categorical, hence these data should be converted into numerical data. The process followed to convert the data has been explained in the next section with a small example.

Nr.	Flight Data	Meaning	Type of Data
1	ADEP	Departure airport	Categorical
2	ADES	Arrival airport	Categorical
3	FILED OFF BLOCK TIME	Planned departure time	DateTime
4	AC Type	Aircraft type	Categorical
5	AC Operator	Aircraft operator	Categorical

Table 3.5: Input parameters

One-Hot Encoding
The categorical data is converted to numerical data by representing in **0's** and **1's**, this process in Machine learning is called *Encoding*. There are several types of encoding are available, but in this work *One-Hot Encoding* is used. The process involved in One-Hot Encoding can be explained by an example. Consider the column for *Departure Airport (ADEP)* shown in Table. 3.6, which has departure airport in ICAO code format. One-Hot Encoding, meaning each possible value in a column becomes a new column3.7. This way the categorical columns become binary columns and can be used by Machine Learning models.

From Table 3.5 it can be observed that *Date of Departure* is in *DateTime* format , the component in this parameter is encoded using *Cyclical Encoding* to convert into numerical data. In Cyclical Encoding the time is converted into sine and cosine components. The date part is one hot encoded for the seven days of week.

Nr.	ADEP
1	EDDP
2	KIAH
3	KSRQ

Table 3.6: Dataset representing departure airport before One Hot Encoding

Nr.	ADEP	ADEP_EDDP	ADEP_KIAH	ADEP_KSRQ
1	EDDP	1	0	0
2	KIAH	0	1	0
3	KSRQ	0	0	1

Table 3.7: Dataset representing departure airport after One Hot Encoding

After converting all input parameters to the numerical datatype, both input and output training data are fed into the Machine Learning model to train. Once the training is completed the validation data is fit to the trained model to predict the output which is cluster labels for each OD pairs. This gives the predicted trajectories for each OD pairs.

The model is trained for a different combination of training data size to observe the effects. The results are validated with EUROCONTROL *PREDICT* logic for the defined metric. In the next section detailed explanations regarding both the PREDICT logic and evaluation metric will be discussed. For a better overview of the data used, Figure 3.4 one of decision trees present in Random Forest are shown schematically.

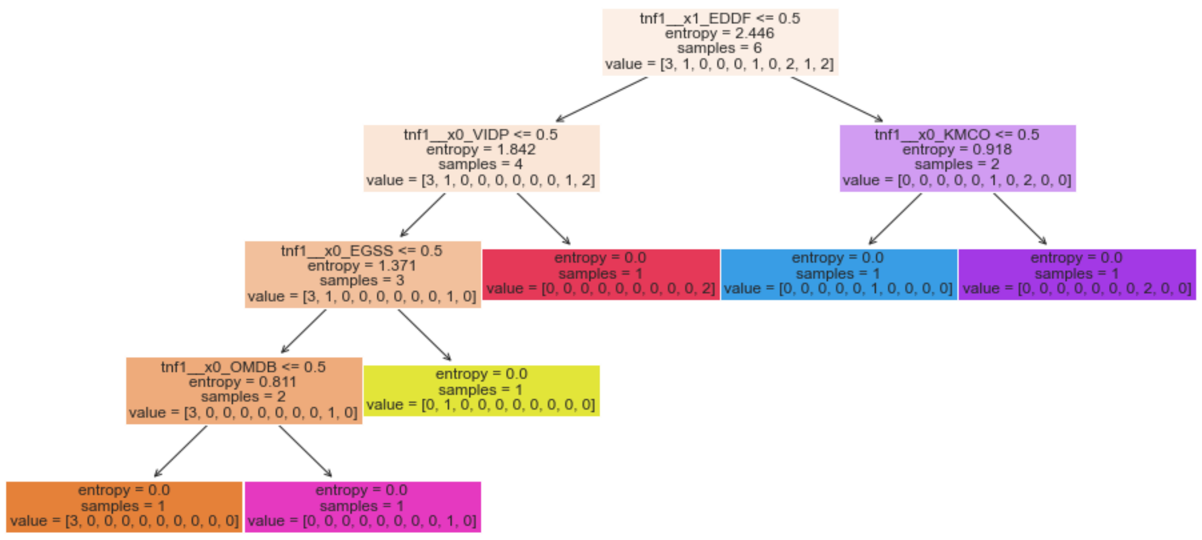

tnf1__x1_EDDF <= 0.5
entropy = 2.446
samples = 6
value = [3, 1, 0, 0, 0, 1, 0, 2, 1, 2]
tnf1__x0_VIDP <= 0.5
entropy = 1.842
samples = 4
value = [3, 1, 0, 0, 0, 0, 0, 0, 1, 2]
tnf1__x0_KMCO <= 0.5
entropy = 0.918
samples = 2
value = [0, 0, 0, 0, 0, 1, 0, 2, 0, 0]
tnf1__x0_EGSS <= 0.5
entropy = 1.371
samples = 3
value = [3, 1, 0, 0, 0, 0, 0, 0, 1, 0]
entropy = 0.0
samples = 1
value = [0, 0, 0, 0, 0, 0, 0, 0, 0, 2]
entropy = 0.0
samples = 1
value = [0, 0, 0, 0, 0, 1, 0, 0, 0, 0]
entropy = 0.0
samples = 1
value = [0, 0, 0, 0, 0, 0, 0, 2, 0, 0]
tnf1__x0_OMDB <= 0.5
entropy = 0.811
samples = 2
value = [3, 0, 0, 0, 0, 0, 0, 0, 1, 0]
entropy = 0.0
samples = 1
value = [0, 1, 0, 0, 0, 0, 0, 0, 0, 0]
entropy = 0.0
samples = 1
value = [3, 0, 0, 0, 0, 0, 0, 0, 0, 0]
entropy = 0.0
samples = 1
value = [0, 0, 0, 0, 0, 0, 0, 0, 1, 0]

Fig. 3.4: Schematic Representation one of decision tree in Random Forest

3.4.2 Validating Machine Learning Model Prediction with PREDICT Logic

PREDICT Logic

While the PREDICT tool is mentioned in a number of EUROCONTROL documents, the specifics of how it is implemented are not made public. What's more important is that the descriptions provided are insufficient for accurately reproducing the behavior of the tool. Following the recommendations of EUROCONTROL experts described by Mnauel Mateos et al in [4], the following workflow has been simulated to operate PREDICT Logic on the clusters created for OD pairs in merged dataset:

1) The tool searches for previous flights that had the same callsign as the current flight. If this is not possible, the flight operated by the same company at the closest time of the same day is chosen.

2) If there is no previous flight for the company available, the same operation is repeated regardless of the Aircraft operator(AC operator Table .3.5).

3) If no flight has yet met the requirements, the most recent FPL for the same OD pair is selected.

Validation

The most popular way of validating any Machine Learning model is to validate with model accuracy. Using accuracy as an evaluation metric is valid where the output labels are distributed evenly as mentioned in section 2.2.8. But it is not the case because for the *imbalanced dataset* how many output labels are present are not known for each OD pair. Using accuracy as the evaluation metric leads to more biased output as the accuracy evaluate for *true positives.*

Therefore, in such cases where the dataset is imbalanced, f1 score is the better evaluation metric mentioned in sec 2.2.8. Once the simulated data from PREDICT logic and prediction from Machine Learning model is obtained. f1 score for both the prediction are calculated by comparing with original output labels . Then, f1 scores of PREDICT and Machine Learning models are compared for better performance.

3.5 Computations

The computations of data preparation, model training and algorithm validation take place on the *High-Performance-Compute-Cluster Bombus* (HPC cluster) of the TUHH. This cluster, running under the *RedHat Enterprise Linux (RHEL) / CentOS 7* operating system uses a *SLURM batch system* to accept and process compute jobs. The Bombus HPC cluster consists of 281 compute nodes, five login nodes, and a parallel storage system with 350TB of storage. A total of 6600 CPU cores, 4 GPUs, and about 32TB of RAM are available for compute-intensive applications [51].

The computer code is written in the programming language *Python 3.9* on *Jupyter Notebook* environment. For the standardization of the data, training and validation of the machine learning models the framework *scikit-learn* is used [52]. For training and validation of *XG boost* is used [53] and similarly for *Light BGM* is used [54].

Fig. 3.5: Algorithmns and Frameworks [52],[53], [54]

4. Results and Discussion

In this chapter, the results of the thesis work is presented and discussed. These comprise evaluation of the merging process of the databases as well as the Semi-Supervised Machine Learning algorithm which will be compared to the implemented EUROCONTROL PREDICT logic.

4.1 Evaluation of Data Preparation

In this section, the data preparation and merging process is being evaluated. The complete datasets selected comprises *Sabre Airport Intelligence Data* and *EUROCONTROL R&D Data.* The datasets chosen is for June, August, and November 2016. The process of merging the dataset has been carried out in two stages. The results for each stage are explained below.

4.1.1 Translation Of IATA Codes to ICAO Codes

The flight information dataset obtained from the *Sabre Airport Intelligence Data* database is in IATA format as mentioned in section 3.1.2. In order to use dataset for Machine Learning model, merging of *Sabre Airport Intelligence Data* with the *EUROCONTROL RnD Data* has been carried out. The IATA codes present in *Sabre Airport Intelligence Data* need to be translated to corresponding ICAO codes to, in order to merge with *EUROCONTROL RnD Data* as mentioned in section 3.2.

Nr.	Sabre Airport data	Meaning
1	flightname	Unique numeric identifier of each flight
2	Origin	IATA code for departure airport
3	Destination	IATA code for arrival airport
4	Airline	IATA code for Aircraft operator
5	DayOfDeparture	Planned Day of flight departure
6	DayOfArrival	Planned Day of flight arrival
7	TimeOfDeparture	Planned Time of flight departure
8	TimeOfArrival	Planned Time of flight departure
9	Equipment	IATA code for Aircraft type

Table 4.1: Parameters in Sabre Airport Data

Parameters in the *Sabre Airport Intelligence Data* database as shown in Table. 4.1. The dataset consisted of **8,738,841** flights between **49,416** unique OD pairs data as shown in the Figure 4.1. For **8,484,674** there were matching flights as shown in Figure 4.1(top). After translating IATA to ICAO Codes a significant reduction in the data was observed regarding unique OD pairs, which was reduced to **46,965** unique data as shown in Figure 4.1(bottom).

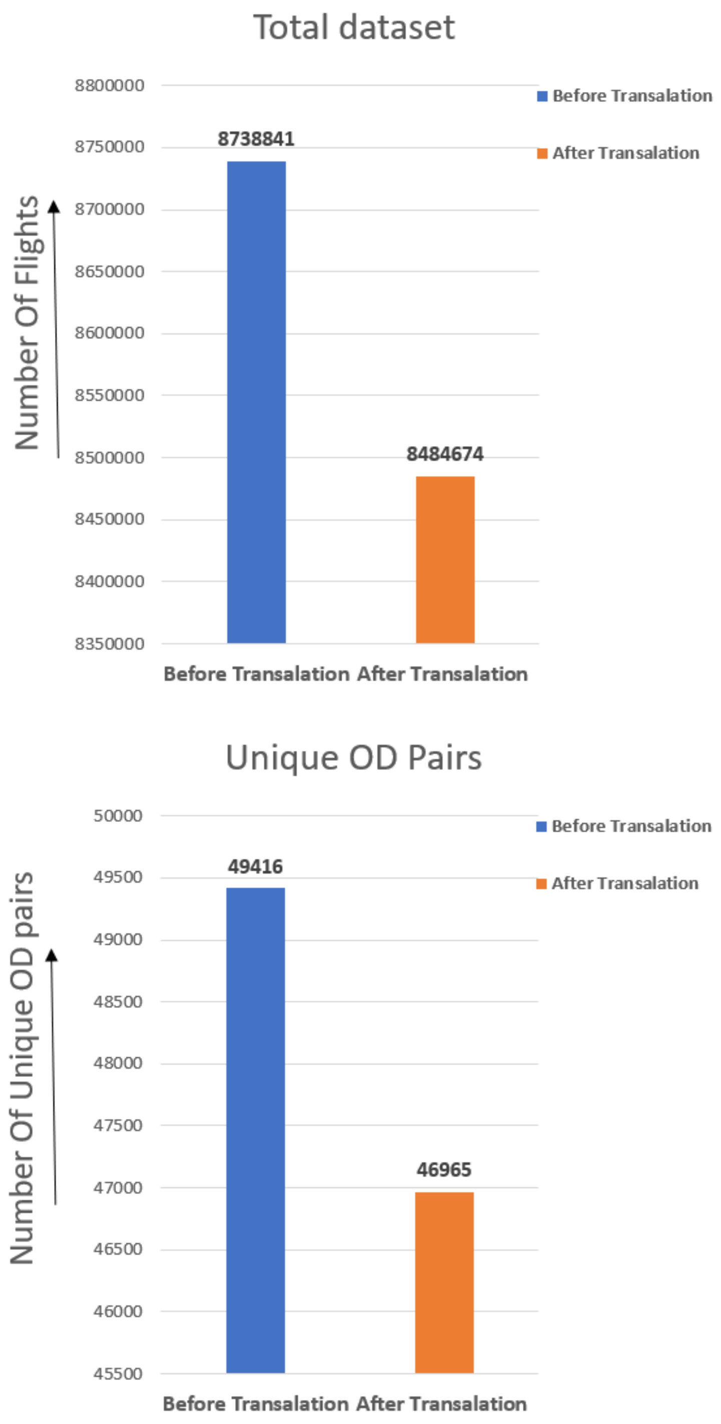


Fig. 4.1: Data loss during the IATA to ICAO translation process for the Sabre Airport Intelligence Data regarding the total number of flights (top) and the number of unique OD pairs (bottom)

As the Sabre Airport Intelligence Data contains flights information from all over the world for the major airlines, while translating the dataset to ICAO codes, there was no sufficient data translating airlines and aircraft IATA codes to ICAO codes. Specifically, data loss happened for the domestic airlines in the European region. Hence there was a significant reduction in corresponding OD pairs data present in European airspace. Therefore, the loss of airlines, aircraft, and OD pairs data for the European region potentially affect the dataset during the merging process of the databases.

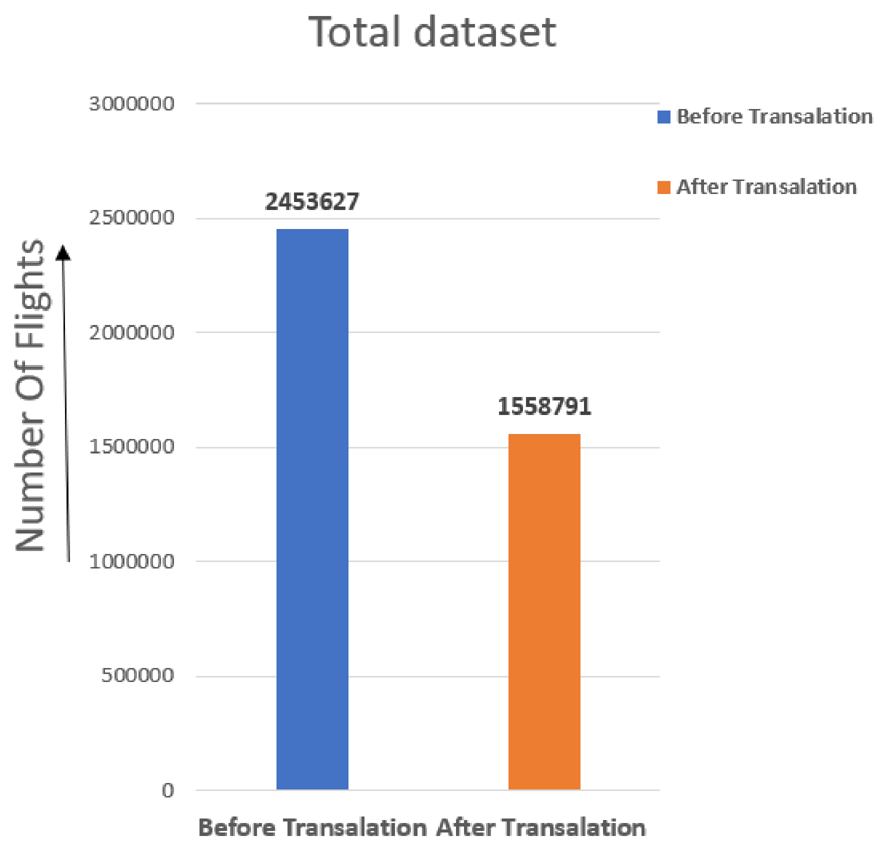
4.1.2 Merging Sabre Intelligence Data With EUROCONTROL RnD Data

EUROCONTROL RnD Data for three months June, August and November of 2016 extracted from *EUROCONTROL* database consisted of **2,453,627** flights between **58,833** unique OD pairs as shown in Figure 4.2. After merging with Sabre Airport Intelligence Data, the data was reduced to **1,558,791** flights and the unique OD pairs were reduced to **16,304** as shown in Figure 4.2. Since the entire dataset for the Machine Learning model should be represented as *EUROCONTROL RnD Data* which has the flight data for European airspace. During the merging process, it can be observed there was huge data loss, the major reason being *Sabre Airport Intelligence Data* did not have sufficient airlines and aircraft data for European airspace. That resulted in the loss of many OD pairs connected to lost airlines and aircraft data.

On concluding the result of data preparation, it can be assumed that the high number of missing flights in the matched dataset affected the results for the Machine Learning prediction as it does not represent the real distribution of the data. Data loss could be minimized by collecting the proper translation codes from trusted sources. The data sources for the translation codes used in this master thesis where inconsistent.

Nr.	**Flight Data**	**Meaning**
1	ECTRL ID	Unique numeric identifier of each flight
2	ADEP	ICAO code for departure airport
3	ADEP Latitude	Departure airport latitude
4	ADEP Longitude	Departure airport longitude
5	ADES	ICAO code for arrival airport
6	ADES Latitude	Arrival airport latitude
7	ADES Longitude	Arrival airport longitude
8	FILED OFF BLOCK TIME	Planned departure time
9	FILED ARRIVAL TIME	Planned arrival time
10	AC Type	ICAO code for Aircraft type
11	AC Operator	ICAO code for Aircraft operator
12	AC Registration	Aircraft registration
13	ICAO Flight Type	Type of flight

Table 4.2: EUROCONTROL Flight Data



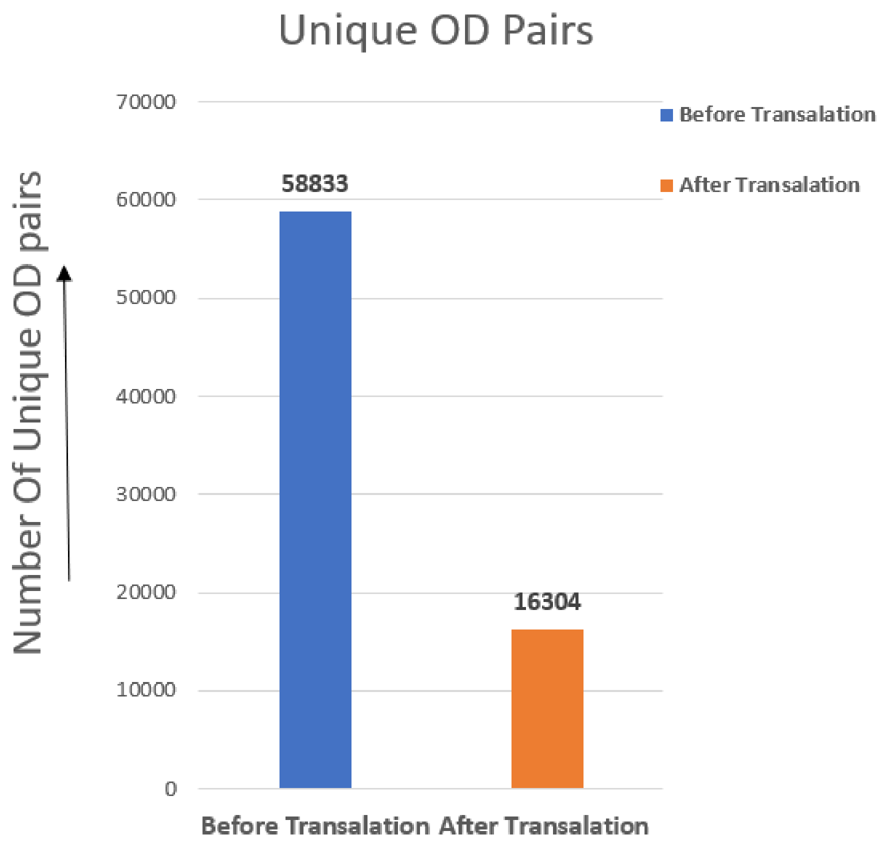


Fig. 4.2: Data loss during merging of Sabre Airport Intelligence Data with EUROCONTROL RnD data, the total number of flights (top) and the number of unique OD pairs (bottom)

4.2 Evaluation of the Clustering process

In this section, results from the *Clustering* algorithm will be presented and will carry out a discussion on evaluating the quality of the clustering of route trajectories.

4.2.1 DBSCAN Clustering

Clustering was carried on the dataset by grouping the dataset by OD pair, where each OD pairs had planned trajectory points as mentioned in the section3.3. A total of **21,199** clusters were created for **16,304** unique OD pairs.The quality of the clusters formed is checked. First, X representative OD pairs (see Table 4.3) are plotted on the map for visualization. Lastly, a scatter plot is created to assess the distance matrix values for each clusters with it's corresponding OD pairs.

ICAO-Codes	**O-D Pairs**
EDDF - EGLL	Frankfurt am Main - London Heathrow
EDDF - OMDB	Frankfurt am Main - Dubai Intl
EGLL - OMDB	London Heathrow - Dubai Intl

Table 4.3: ICAO-Codes for OD Pairs

4.2.2 Visual Evaluation of the Clustering process

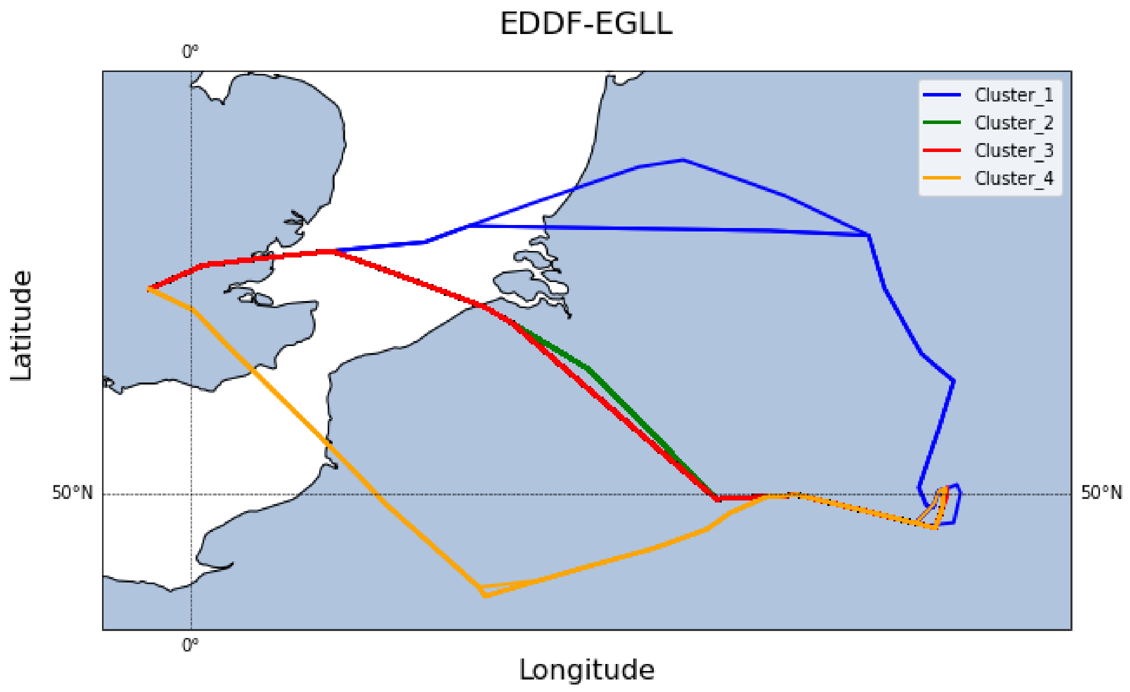
EDDF-EGLL

Figure. 4.3 shows the trajectories clustered for the route between Frankfurt am Main - London Heathrow (top) and London Heathrow - Frankfurt am Main (bottom).

For the route, Frankfurt - London, the flight planned trajectories are clustered into four clusters (4.3). From the plot, it is evident that there are three clusters present, but the algorithm forms four clusters. As it is a longer-haul route (Cluster_1 in Figure 4.3), the area between the two routes is large. Hence, the algorithm considered this as one cluster rather than two.

For the route, London - Frankfurt, the flight planned trajectories are clustered into six clusters (4.3. Here the area between each route is small. That's why the it can be observed that separate clusters are formed.

Whether the results from the *DBSCAN* algorithm is similar among other OD pairs shall be investigated further by looking at the results of two other OD pairs to understand the reasons behind it.



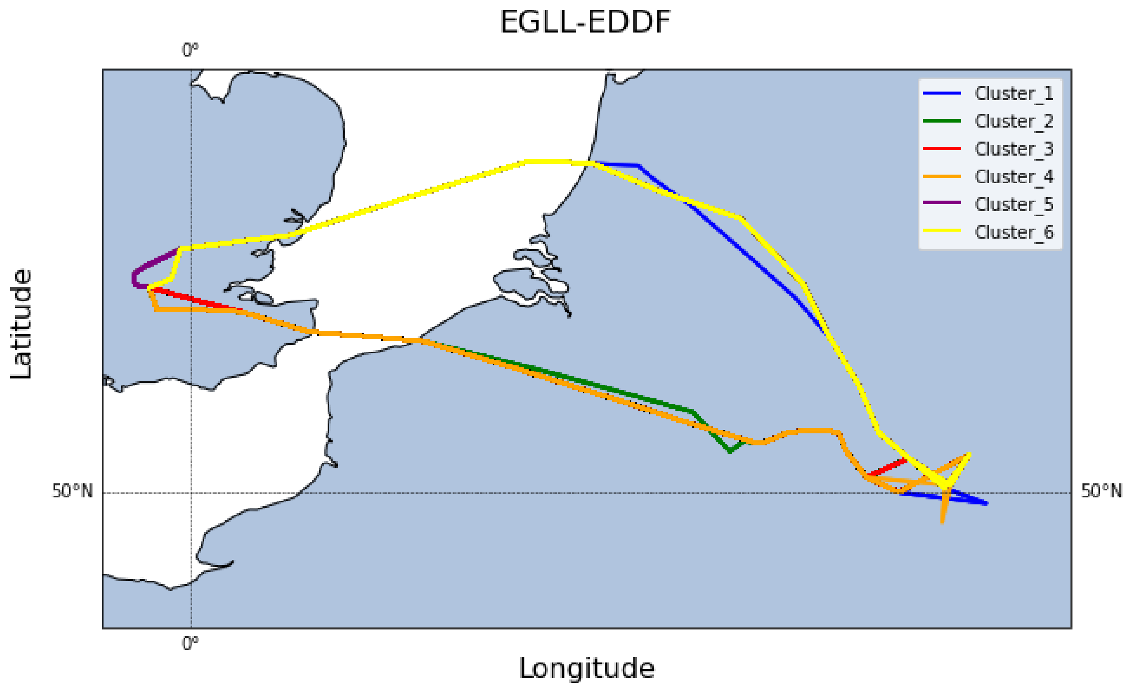
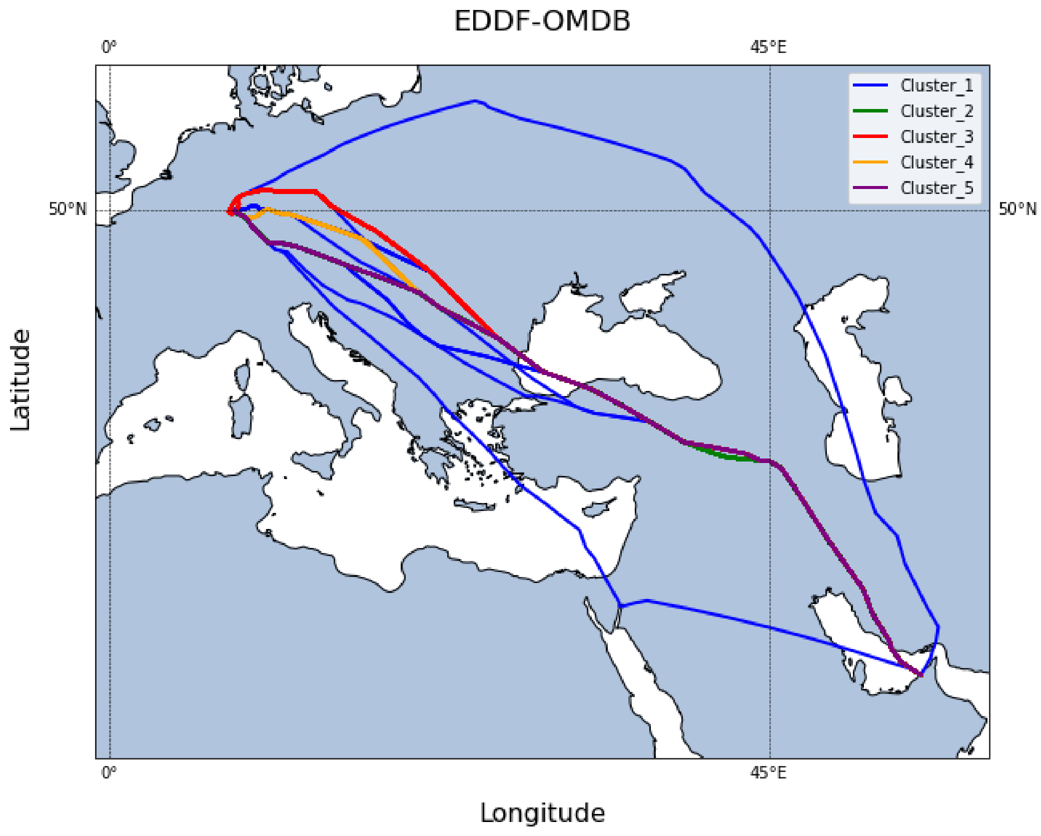


Fig. 4.3: Resulting clusters of flight trajectories from DBSCAN for Frankfurt am Main - London Heathrow (top) and London Heathrow - Frankfurt am Main (bottom).

EDDF-OMDB

Figure. 4.4 shows the trajectories clustered for the route between Frankfurt am Main - Dubai Intl (top) and Dubai Intl - Frankfurt am Main (bottom)

Five clusters were formed for the route Frankfurt am Main - Dubai and six clusters were formed for the route Dubai - Frankfurt am Main, as shown in Figure. 4.4. Here, it could be seen that the algorithm has created as a single cluster for the long-haul flight route, even though separate branches for the routes are present (cluster_1). The reason is similar as mentioned in section 4.2.2



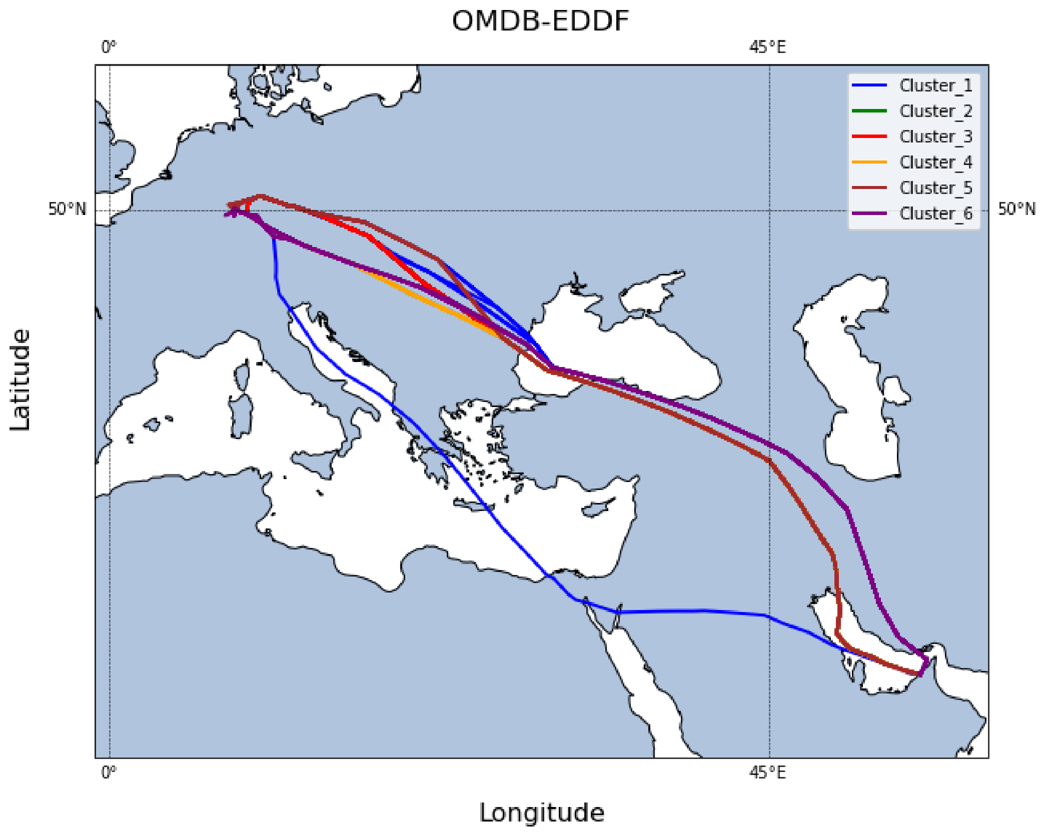


Fig. 4.4: Resulting clusters of flight trajectories from DBSCAN for Frankfurt am Main - Dubai Intl (top) and Dubai Intl - Frankfurt am Main (bottom).

EGLL-OMDB

Figure. 4.5 shows the trajectories clustered for the route between London Heathrow - Dubai Intl (top) and Dubai Intl - London Heathrow (bottom)

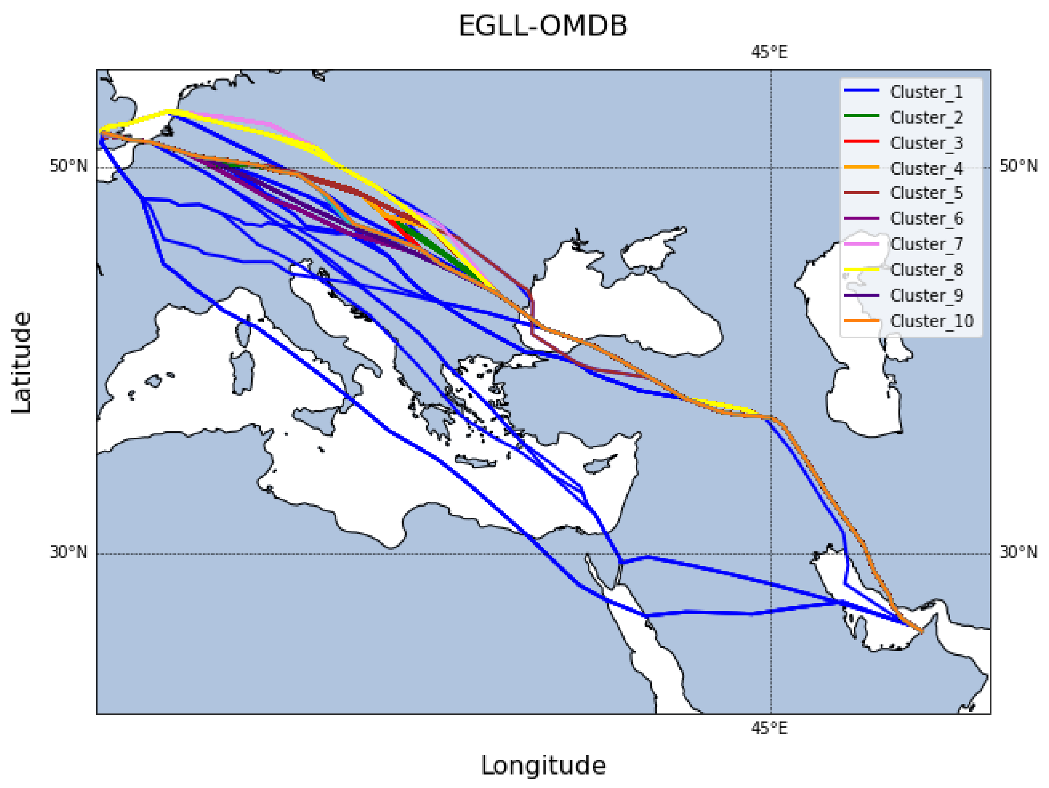
For each route mentioned above, ten clusters are formed by the algorithm, as shown in Figure. 4.5. Similar behavior is observed where the area is large, i.e., for the long-haul flight route, the algorithm has considered it as a single cluster.

Discussion

The reason for this behavior could be the choice of hyperparameter **eps**(ϵ) used in DBSCAN. **eps**(ϵ) in DBCAN is a distance measure that will be used to locate points that are in close proximity to any given point. The Mean value of distance metric for each OD pairs is used as the (ϵ). Because in order to create the cluster DBSCAN makes use of metric in this thesis work it is nothing but a distance matrix which is a square matrix that has the values of the area between two routes for each trajectory. Therefore, (ϵ) was assigned with an average value of the distance metric.

Reachability is a concept in DBSCAN, where a point is considered reachable from another if it is located within a specified distance (ϵ) to the first. For longer-haul flights, the distance from the (ϵ) to all the trajectory points corresponding to long haul flights are considered more or less similar. Thus, this could be another reason that single cluster was formed for long-haul flights.

On concluding, the creation of clusters could be improved by hyperparameter tuning. Also, in general there is lack research on the hyperparameter tuning of (ϵ) and hyper parameter tuning was not part of this thesis work. Hence hyperparameter tuning has not been done in this thesis work.



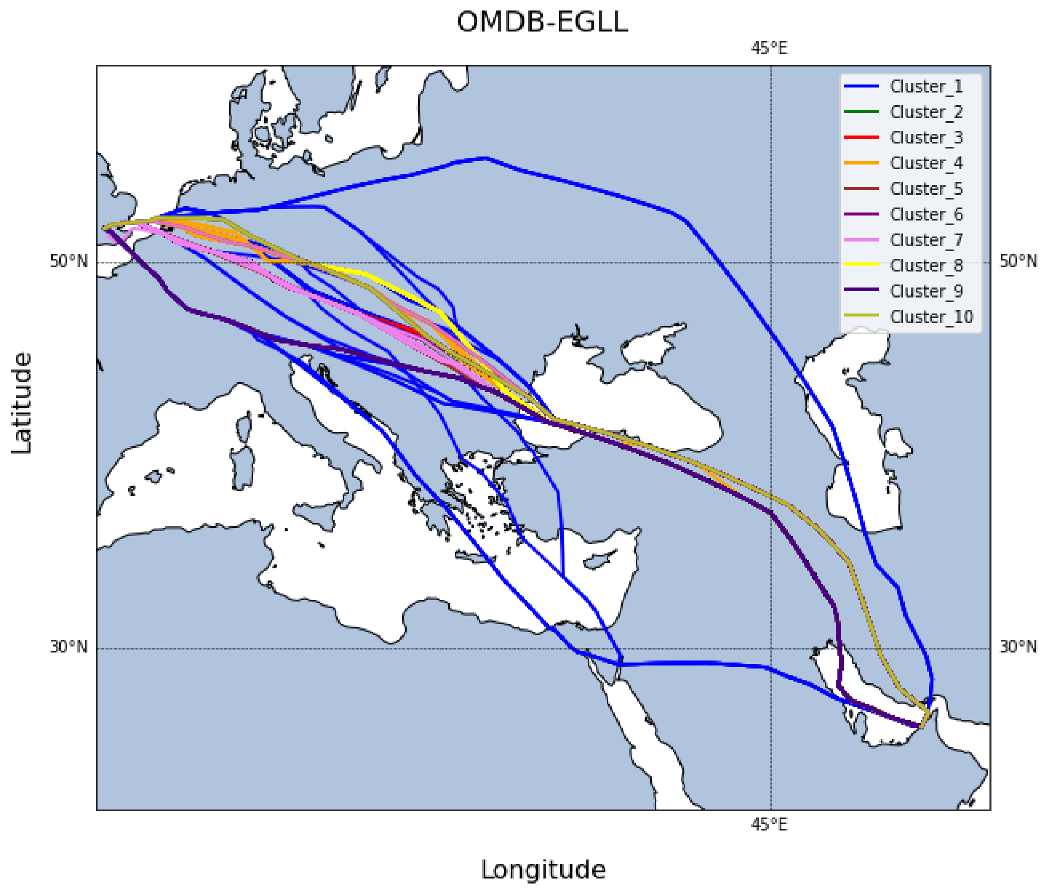


Fig. 4.5: Resulting clusters of flight trajectories from DBSCAN for London Heathrow - Dubai Intl (top) and Dubai Intl - London Heathrow (bottom).

4.2.3 Evaluation of the Clustering process by Distance Matrices

Figure 4.6 shows the scatter plot, where the values of the average distance metric for an OD pair represent the x-axis for each OD pair and the average distance metric for each cluster represents the y-axis for each trajectory. In addition, curve fitting is performed to check the quality of the cluster. Ideally, it is expected that all the points should be above the curve.

From graph 4.6, the discussion can be made on two areas. Firstly, on the lower left area where a majority of the points on the red line and a few points which are below the line, indicating the points for short-haul flights and only few outiliers were present. Next to upper right corner of the graph where a majority of the points are over the red line and some points are below the red line. These points represents the long-haul flight, which were clustered into single cluster as discussed in the previous section.

In the cases where the area between two trajectories was comparatively high compared to other trajectories for the same OD pair. After clustering, the investigation showed that the algorithm labeled these points as outliers. Around **800** clusters were found as outliers from the study.

Before looking into the results of the Machine Learning models, a few assumptions were made.At first, only OD pairs with two or more clusters are considered for the training and validation process of the Machine Learning models. Secondly, outliers are neglected which is a common practice in Machine Learning.

After clustering there were 3407 OD pairs identified containing only one cluster, as the routes trajectories inside these OD pair were identical. So in the Machine Learning model ,a total of 12,097 OD pairs were considered.

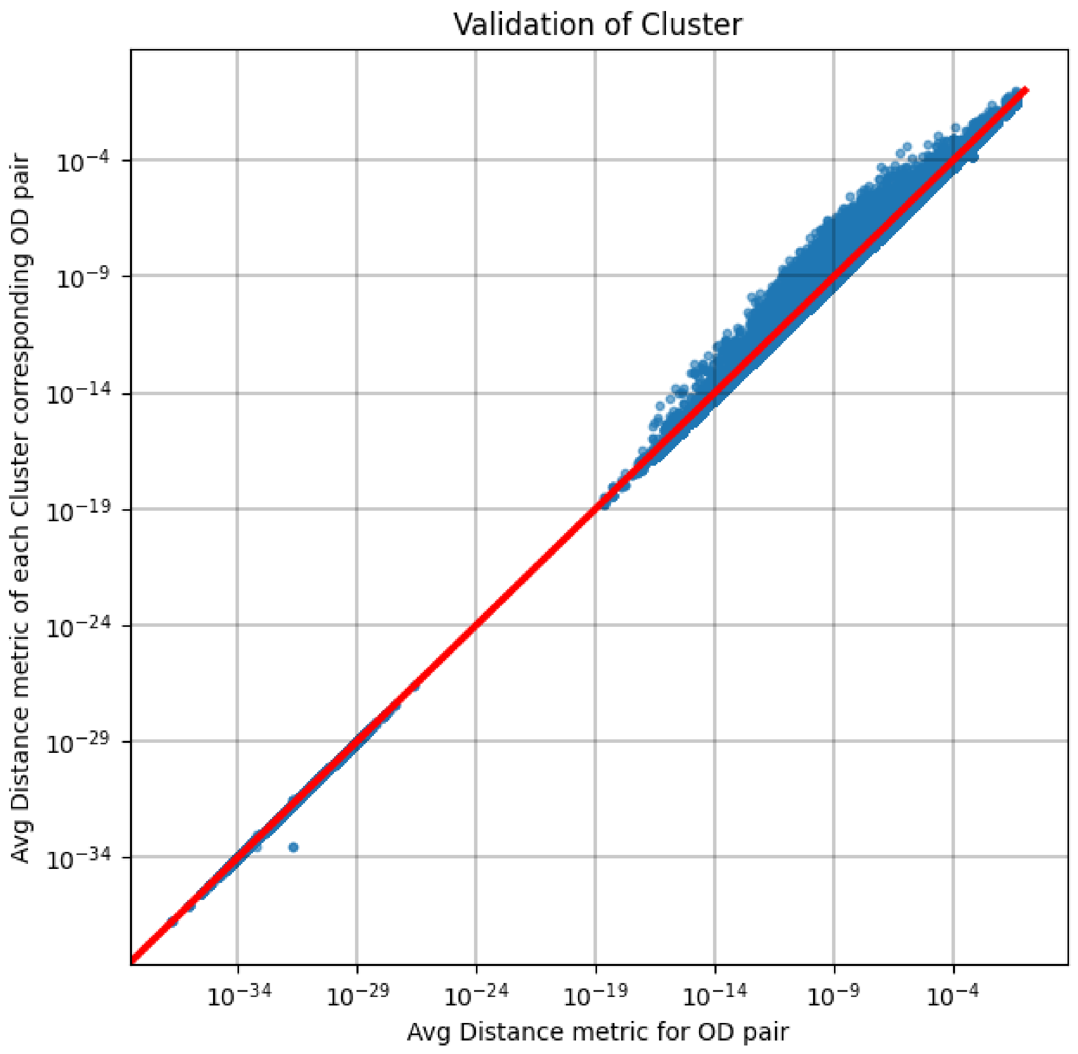


Fig. 4.6: Average distance metric of each cluster compared to the average distance metric of the corresponding OD pair

4.3 Evaluation of the used Machine Learning Models

Initially to observe the performance on the prediction three Machine Learning models were selected *Random Forest*,*XG Boost*, and *Light GBM*. Due to computational constraints only *Random Forest* was trained. The results from used Machine Learning algorithm *Random Forest* model will be discussed in the following section.

4.3.1 Validation of the Random Forest Algorithm

Evaluation of the *Random Forest* algorithm mentioned in section3.4.1 followed the particular procedure that will be helpful in the validation of predicting trajectory labels for each OD pairs considered in the Machine Learning model training.

Performance validation of model prediction on the validation dataset is undertaken using the *f1-score* as primary metric , because as mentioned section 2.2.8 using accuracy as evaluation metric is not suited for where there is imbalance distribution of labels. If the dataset which comprises of multiple labels with imbalanced distribution to be predicted *f1-score* is best metric for the evaluation. Following assumptions are made before the evaluating the models:

- The true and predicted discrete cluster labels were used for the validation.
- The *f1-score* is a value between 0 and 1, whereas 1 equals a perfect prediction and 0 equals no correct prediction at all.
- For each cluster label, *f1-score* is calculated as ratio of harmonic mean between precision and recall as mentioned in chap .2.2.8
- The number of flights for each OD pair is considered as weight for the graph depicting the *f1-score*.

Figure.4.7 depicts the results of *f1-score* for each OD pair, evaluating predictions based on the PREDICT Logic (X-axis) and based on the Machine Learning model (y-axis).

In general the Machine Learning model scores a higher f1-score than the PREDICT logic. The Overall *f1-score* of the Machine Learning model predicted **82**% of cluster labels were predicted correctly. This is **5**% more in *f1-score* than what the PREDICT Logic has predicted. Furthermore, on OD pair level evaluation the following results can be observed:

- For **18.3**% of the OD pairs, Machine Learning model outperformed PREDICT Logic on *f1-score* by **25**% on an average
- For **17.3**% of the OD pairs, PREDICT Logic outperformed the Machine Learning model by *f1-score* by **9**% on an average

- For **63.7**% of the OD pairs predicted by the Machine Learning model is similar to the predictions made by the PREDICT Logic

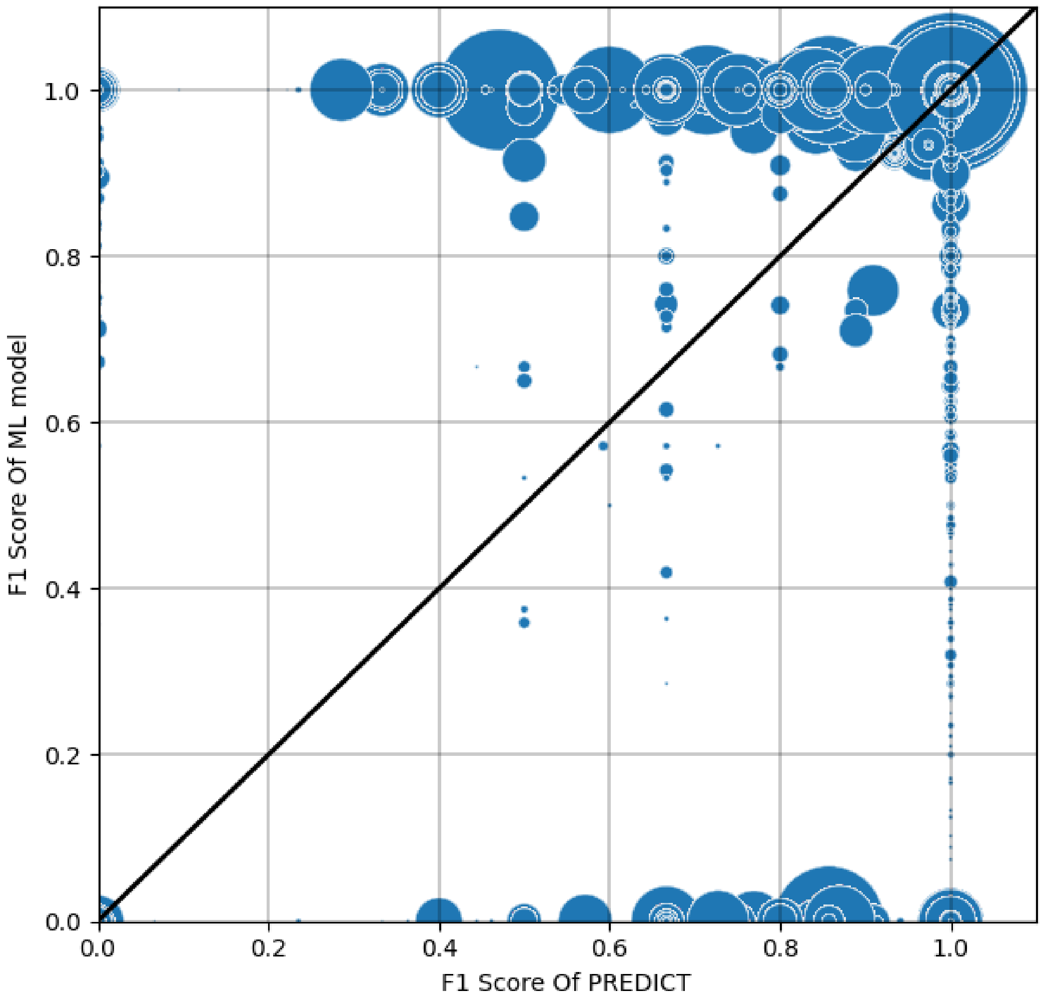


Fig. 4.7: Average f1-scores for predictions made by the Machine Learning model (y-axis) vs the PREDICT logic (x-axis) for each OD pair. Circle size represents the number of flights in that OD pair

f1-scores is by given harmonic mean of precision and recall as mentioned in section 2.2.8, precision for this model is around **83**% and recall is around **81**%. Hence, the f1-score is **82**%. This is the reason the why majority of the values in the scatter plot is at **1** on y axis. This means there is a significant miss in the classified instances. Around **22**% of the cluster label did not appear in the Machine Learning prediction. Hyper parameter tuning was performed for the optimized results, but it was observed that more computation of hyper parameter was necessary which required more time. The model proposed does seem to have a less recall when compared to precision. Hence, the prediction is biased towards precision.

On concluding a better f1 score can be achieved by finding better trade-off between precision and recall. Which can be achieved with extensive hyper parameter tuning. This will result in better prediction of the trajectories.

5. Summary

In the context of this master thesis,a structured analysis of a comprehensive set of flight plan data is carried out with the help of advanced analytical methods from the fields of Machine Learning and Deep Learning in order to draw conclusions about ATFM-decision processes in the pre-tactical phase regarding operational influence parameter. For this purpose, operational flight plan data for three months, June, August, and November for year 2016 was taken from *EUROCONTROL RnD* and *Sabre Airport Intelligence Data* database, and was processed and parameterized and fed into ML/DL model to predict the flight planned trajectories.

The entire dataset used in thesis contained flight specific datapoints for **1,558,791** number of flights . The datasets where merged in a pre-processing step in order to use them as an input for the Machine Learning models. Before the merging process, the Sabre Airport Intelligence data first had to be translated from IATA to ICAO Codes After, translation the data was merged with *EUROCONTROL RnD*. During the merge, data loss was observed since data from *Sabre Airport Intelligence Data* did not contain the information on domestic flights in European region.

Semi-Supervised Learning approach was taken, which is the combination of both *Unsupervised Learning* and *Supervised Learning*. The output labels used in the training and validation process are determined by the Unsupervised Learning technique called clustering. Hence, *Unuspervised Learning* was employed to generate the output labels by using a technique called *clustering*.In order to handle the large amount of data, an automated clustering technique called DBSCAN was selected. Before feeding into *DBSCAN* dataset was split randomly into training set which had 80% and test set which 20% of the dataset respectively. Clustering was performed on the training dataset, and the test dataset was transformed to the training set correspondingly.

The *DBSCAN* algorithm determined the clusters labels for each *O-D pairs* by taking the lateral area between two trajectories and normalizing over the haversine distance between the corresponding OD pair as distance matrix value. A total of **21,199** clusters were formed for **16,304** unique OD pairs.

In order to validate the performance and quality of the clusters created, clusters obtained for a representative OD pairs was plotted on the map. It was observed, in general the clustering algorithm produced decent results. While determining the clusters for log haul flights, in spite of having many branches it was clustered as single cluster. This was because of the lateral area for the long flights was bigger. On further investigation, plotting scatter plot by taking an average distance matrix for each OD pair against the average distance matrix to its corresponding cluster.

The plot confirmed again that long haul flight was below the curve fit over the data points. The reasons for the data points below the curve was that the average lateral area between those trajectories were more than the average distance matrix.

Once the cluster labels were created for each flights, the dataset was fed into the Machine Learning algorithm. Using *Supervised Learning* techniques prediction of trajectory was carried out. The presented dataset was imbalanced data with *multi-class* and *multi-label* classification problem. Ensemble Learning was selected as a *Supervised Learning* in order to predict planned flight routes was selected over the popular *Neural Network. Ensembling Learning* combines several individual models to obtain better generalization performance. Also, *multi-class* and *multi-label* problems don't need any sampling of the data.

Ensembling Learning like *Random Forest*, *XG Boost* and *Light GBM* algorithms were implemented. Then compare the performance among each other and also with the PREDICT Logic. However, due to the computational constraint, only *Random Forest* was trained. The predictions created by the Random Forest algorithm showed an average f1-score of 82%

Similarly *f1-score* for PREDICT Logic with the original cluster label, which performed **5**% was less than the Machine Learning model. In general, the Ml model performed well predicting the labels, the model is biased towards precision of the model with slightly less recall.

In conclusion, even though the prediction from the Machine Learning model performed well, the model's recall is low. This could be improved by doing hyperparameter tuning, like performing *Randomized Search Cross-Validation* or *Grid Search Cross-Validation.*

6. Future Work

Successful use of the model resulting in more accurate prediction of trajectories in the pre-tactical phase would help to predict the airspace demand earlier. It also leads to more efficient planning processes for the air-traffic controller.

There are several obstacles to overcome in order to successfully transfer the developed models into the ATFM's applied area. As the aviation industry is a safety-critical and strictly regulated industry, it must overcome bureaucratic obstacles.The mode of action of complex Machine Learning models is not entirely transparent, additional security precautions would be required. Finally, the models would have to be retrained using the new data after significant changes in traffic patterns.

In order to improve the performance of the proposed model in this thesis, a more consistent database could be used including all airports codes in Europe. A larger and more consistent database could increase the performance of the model, because this would eventually increase the performance of the model as more data is available to train the model.

Hyperparameter tuning of DBSCAN could be carried out, which could increase cluster accuracy. In this work, only the geometrical aspect of the OD pair's route trajectory has been considered in the clustering process. The use of airspace and sector information would result in a more accurate clustering of the trajectories.

In theory, boosting models in Ensemble Learning perform better than the model proposed in this work. the investigated boosting methods like, XG Boost and Light BGM, could be evaluated for better performance with additional computational resources. Furthermore, even deep learning approaches like LSTM with attention mechanism could be one of the approaches that can be looked into.

The model used in this work is using the day of the week, the time, and the airlines and aircraft type as inputs. Adding weather information, especially wind, congestion, and special events, would help predict the planned trajectories and helps in ATFM decisions and in reducing resources involving ATFM measures as mentioned in [55] [56].

Literature index

[1] Statista, “Number of flights performed by the global airline industry from 2004 to 2022.” [Online]. Available: https://www.statista.com/statistics/564769/airline-industry-number-of-flights/weltweiten-fluege/ [Accessed on 10, 2021]

[2] EUROCONTRO2021, “European flight movements and service units-three scenarios for recovery from covid-19.” [Online]. Available: https://www.eurocontrol.int/sites/default/files/2021-10/eurocontrol-7-year-forecast-2021-2027.pdf [Accessed on 12 2021]

[3] “Karlsruhe in 2018: Nm user forum,” in *Main Operational Main Operational Issues at UAC*, H. Schneider, Ed.

[4] M. R. Mateos, I. Martín, P. García, Ricardo, Herránz, and O. G. Cantu-Ros, “Full-scale pre-tactical route prediction machine learning to increase pre-tactical demand forecast accuracy,” 2020.

[5] “Air traffic flow management (atfm).” [Online]. Available: Available:https://skybrary.aero/articles/air-traffic-flow-management-atfm [Accessed on 10 2021]

[6] Anestis Doutsis, “Predicting the amount of air traffic demand regulations using machine learning,” 2020. [Online]. Available: http://resolver.tudelft.nl/uuid:72cb92d5-a904-4038-b075-a05cae1376ca [Accessed On 11 2021]

[7] Flynn, Brian, “Atfm in europe,” uRL. [Online]. Available: https://www.icao.int/APAC/Meetings/2013

[8] Cech,S.Niarchakou, *ATFCM Operations Manual.* Brussels, Belgium: EUROCONTROL. [Online]. Available: https://www.eurocontrol.int/publication/atfcm-operations-manual [Accessed on 06 2021]

[9] Kuenz, Alexander and Schwoch, Gunnar, Korn, Bernd, Gräupl, Thomas, Rippl, Markus, Forster, Caroline, Gerz, Thomas, Volker, Grewe, Matthes, Sigrun, Linke, Florian, and Radde, Marius, “Optimization without limits-the world wide air traffic management project,” 09 2017.

[10] Eurocontrol, “Supporting supporting european aviation.” [Online]. Available: https://www.eurocontrol.int/ [Accessed on 07 2021]

[11] Eurocontrol Network Operator, "Network operations report 2020." [Online]. Available: https://www.eurocontrol.int/sites/default/files/2021-04/eurocontrol-annual-nor-2020-main-report.pdf [Accessed on 07 2021]

[12] McCarthy, John and Minsky, Marvin L and Rochester, Nathaniel and Shannon, Claude E, "A proposal for the dartmouth summer research project on artificial intelligence, august 31, 1955," *AI magazine*, vol. 27, no. 4, pp. 12–12, 2006.

[13] Samuel, A. L., "Some studies in machine learning using the game of checkers," *IBM Journal of Research and Development*, vol. 3, no. 3, pp. 210–229, 1959.

[14] Joshua Mills, "Introduction to ai part 1," *Edzion*. [Online]. Available: https://edzion.com/2020/12/09/introduction-to-ai-part-1/ [Accessed on 12 2021]

[15] Özbek, A.F., "How to train a model with mnist dataset." [Online]. Available: https://medium.com/@afozbek/how-to-train-a-model-with-mnistdataset- [Accessed on 9 2021]

[16] Deng, Li, "The mnist database of handwritten digit images for machine learning research," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012.

[17] Arthur V. Ratz, "Classifying data using artificial intelligence k-means clustering algorithm," 2020. [Online]. Available: https://www.codeproject.com/Articles/5256294/Classifying-Data-Using-Artificial-Intelligence-K-M [Accessed on 12 2021]

[18] Alexander Thamm, "Einfach erklärt: So funktioniert reinforcement learning." [Online]. Available: https://www.alexanderthamm.com/de/artikel/einfach-erklaert-so-funktioniert-reinforcement-learning/ [Accessed on 10 2021]

[19] Ho, Yu-Chi and Pepyne, David L, "Simple explanation of the no-free-lunch theorem and its implications," *Journal of optimization theory and applications*, vol. 115, no. 3, pp. 549–570, 2002.

[20] Aurélien Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*, 2nd ed. O'Reilly, 2019.

[21] "What to know about retrieving video evidence in intersection accident cases." [Online]. Available: https://www.fosterwallace.com/blog/retrieving-video-evidence-in-intersection-accident-cases.cfm [Accessed on 12 2021]

[22] GeeksforGeeks, "Clustering in machine learning," 2021. [Online]. Available: https://www.geeksforgeeks.org/clustering-in-machine-learning/ [Accessed on 12 2021]

[23] Nagesh Singh Chauhan, "Dbscan clustering algorithm in machine learning," 2020. [Online]. Available: https://www.kdnuggets.com/2020/04/dbscan-clustering-algorithm-machine-learning.html [Accessed on 12 2021]

[24] javatpoint, "Regression vs. classification in machine learning," 2020. [Online]. Available: https://www.javatpoint.com/regression-vs-classification-in-machine-learning[Accessed on 12 2021]

[25] J. Brownlee, “4 types of classification tasks in machine learning,” 2020. [Online]. Available: https://machinelearningmastery.com/types-of-classification-in-machine-learning/ [Accessed on 12 2021]

[26] Yufeng G, “The 7 steps of machine learning,” 2017. [Online]. Available: https://towardsdatascience.com/the-7-steps-of-machine-learning-2877d7e5548e [Accessed on 12 2021]

[27] Scikit-learn, “Underfitting vs. overfitting.” [Online]. Available: https://scikitlearn.org/0.15/auto_examples/plot_underfitting_overfitting.html [Accessed on 12 2021]

[28] Cheatsheet, M.L., “Loss functions.” [Online]. Available: https://mlcheatsheet.readthedocs.io/en/latest/loss_functions.html [Accessed on 12 2021]

[29] Ertel, Wolfgang and Black, Nathanael T, *Grundkurs Künstliche Intelligenz.* Springer, 2016.

[30] Müller, A.C. and Guido, S., *Introduction to machine learning with Python: guide for data scientists.* Beijing: O’Reilly.

[31] Will Koehrsen, “Random forest simple explanation.” [Online]. Available: https://williamkoehrsen.medium.com/random-forest-simple-explanation-377895a60d2d [Accessed on 12 2021]

[32] Jake Hoare, “Gradient boosting explained – the coolest kid on the machine learning block.” [Online]. Available: https://www.displayr.com/gradient-boosting-the-coolest-kid-on-the-machine-learning-block/[Accessed on 12 2021]

[33] UC Business Analytics R Programming Guide, “Gradient boosting machines.” [Online]. Available: http://uc-r.github.io/gbm_regression [Accessed on 12 2021]

[34] W. Wang, G. Chakraborty, and B. Chakraborty, “Predicting the risk of chronic kidney disease (ckd) using machine learning algorithm,” *Applied Sciences*, vol. 11, p. 202, 12 2020.

[35] Derrick Mwiti, “Lightgbm: A highly-efficient gradient boosting decision tree,” 2018. [Online]. Available: https://www.kdnuggets.com/2020/06/lightgbm-gradient-boosting-decision-tree.html [Accessed on 12 2021]

[36] W. Hulleman and R. Vos, “Modeling abiotic niches of crops and wild ancestors using deep learning: A generalized approach,” 10 2019.

[37] Clare Liu, “More performance evaluation metrics for classification problems you should know,” 2020. [Online]. Available: https://www.kdnuggets.com/2020/04/performance-evaluation-metrics-classification.html [Accessed on 12 2021]

[38] Jaap Bouwer, Steve Saxon and Nina Wittkamp, “Back to the future? airline sector poised for change post-covid-19.” [Online]. Available:

https://www.mckinsey.com/industries/travel-logistics-and-infrastructure/our-insights/back-to-the-future-airline-sector-poised-for-change-post-covid-19 [Accessed on 12 2021]

[39] European Comission, “European partnership under horizon europe integrated air traffic management,” 2021. [Online]. Available: https://ec.europa.eu/info/sites/default/files/research-and-innovation/funding /documents/ec-rtd-he-partnerships-integrated-atm.pdf [Accessed on 12 2021]

[40] James Nurton, “World intellectual property organization, “the ip behind the ai boom.” [Online]. Available: https://www.wipo.int/wipo_magazine/en/2019/01/article_0001.html: :text=Artificial[Accessed on 12 2021]

[41] axelsoft, “10,ways airlines use artificial intelligence and data science to improve operation.” [Online]. Available: https://www.altexsoft.com/blog/engineering/ai-airlines/[Accessed on 12 2021]

[42] S. Khanna, “Post covid-19 norms speed up tech adoption in the aviation sector.” [Online]. Available: https://www.linkedin.com/pulse/post-covid-19-norms-speed-up-tech-adoption-aviation-sector-khanna/[Accessed on 12 2021]

[43] Hamed, M.G. and Gianazza, D. and Serrurier, M. and Durand, N., “Statistical prediction of aircraft trajectory: regression methods vs point-mass model.”

[44] H. Naessens, T. Philip, M. Piatek, K. Schippers, and R. Parys, “Predicting flight routes with a deep neural network in the operational air traffic flow and capacity management system. predicting flight routes february 2018, eurocontrol.”

[45] Z. Wang, M. Liang, and D. Delahaye, “Short-term 4d trajectory prediction using machine learning methods,” in *In proceedings of the 7th SESAR Innovation Days.*

[46] R. Sanaei, B. A. Pinto, and V. Gollnick, “Toward atm resiliency: A deep cnn to predict number of delayed flights and atfm delay,” *Aerospace*, vol. 8, no. 2, 2021. [Online]. Available: https://www.mdpi.com/2226-4310/8/2/28

[47] “Atfcm operating procedures for flow management position.” [Online]. Available: https://www.eurocontrol.int/publication/atfcm-operations-manual [Accessed on 12 2021]

[48] “User guide sabre airvision market intelligence.” [Online]. Available: https://www.eurocontrol.int/dashboard/rnd-data-archive [Accessed on 06 2021]

[49] “User guide sabre airvision market intelligence.” [Online]. Available: https://www.sabre.com/products/market-intelligence/ [Accessed on 06 2021]

[50] Jason Brownlee, “Ensemble learning methods for deep learning neural networks,” 2019. [Online]. Available: https://machinelearningmastery.com/ensemble-methods-for-deep-learning-neural-networks/ [Accessed on 12 2021]

[51] TUHH, "Linuxcluster Bombus," 2021. [Online]. Available: https://www.tuhh.de/rzt/services/high-performance-computing/hardware.html [Accessed on 12 2021]

[52] Scikit-Learn, "scikit-learn machine learning in python." [Online]. Available: https://scikit-learn.org/stable/ [Accessed on 12 2021]

[53] XGBoost, "Xgboost documentation," 2021. [Online]. Available: https://xgboost.readthedocs.io/en/stable/ [Accessed on 12 2021]

[54] LightGBM, "Welcome to lightgbm's documentation," 2021. [Online]. Available: https://lightgbm.readthedocs.io/en/latest/ [Accessed on 12 2021]

[55] A. Jardines, M. Soler, A. Cervantes, J. García-Heras, and J. Simarro, "Convection indicator for pre-tactical air traffic flow management using neural networks," *Machine Learning with Applications*, vol. 5, p. 100053, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2666827021000256

[56] M. Mateos Villar, I. Martín, R. Alcolea, R. Herranz, O. García Cantú-Ros, and X. Prats Menéndez, "Unveiling airline preferences for pre-tactical route forecast through machine learning. an innovative system for atfcm pre-tactical planning support," in *Proceedings of the 11th SESAR Innovation Days*, 2021.

List of Figures

List of Tables

A. Appendix

A.0.1 Sabre Airport Data Intelligence

Nr.	Sabre Airport data	Meaning
1	flightname	Unique numeric identifier of each flight
2	Origin	IATA code for departure airport
3	Destination	IATA code for arrival airport
4	Airline	IATA code for Aircraft operator
5	DayOfDeparture	Planned Day of flight departure
6	DayOfArrival	Planned Day of flight arrival
7	TimeOfDeparture	Planned Time of flight departure
8	TimeOfArrival	Planned Time of flight departure
9	Equipment	IATA code for Aircraft type
10	Duration	Duration of flight flown
11	Distance	Distance of flight flown
12	Equipment Capacity	Aircraft capacity

Table A.1: Sabre Airport Data Intelligence [49]

A.0.2 EUROCONTROL RnD Data

Nr.	Flight Data	Meaning
1	ECTRL ID	Unique numeric identifier of each flight
2	ADEP	ICAO code for departure airport
3	ADEP Latitude	Departure airport latitude
4	ADEP Longitude	Departure airport longitude
5	ADES	ICAO code for arrival airport
6	ADES Latitude	Arrival airport latitude
7	ADES Longitude	Arrival airport longitude
8	FILED OFF BLOCK TIME	Planned departure time
9	FILED ARRIVAL TIME	Planned arrival time
10	AC Type	ICAO code for Aircraft type
11	AC Operator	ICAO code for Aircraft operator
12	AC Registration	Aircraft registration
13	ICAO Flight Type	Type of flight
14	Actual Off-Block Time	Actual departure time
15	Actual Arrival Time	Actual arrival time

16	ICAO Flight Type	ICAO Flight Type: S – Scheduled, N - Non-scheduled commercial operation
17	STATFOR Market Segment	Market Segement definitions
18	Requested FL	Requested Cruising flight level
19	Actual Distance Flown (nm)	Distance flown in nautical miles

Table A.2: EUROCONTROL Rnd Data [48]