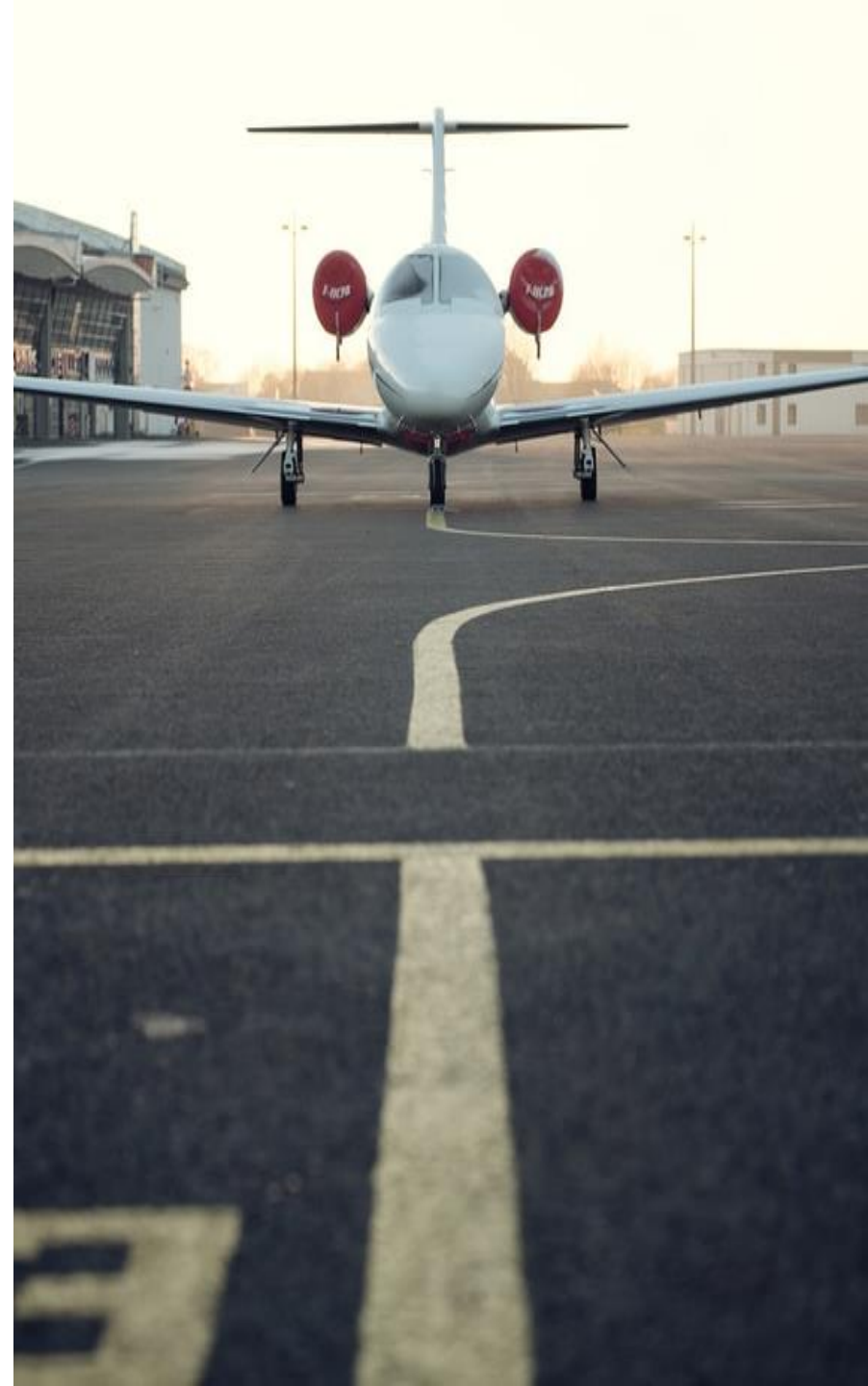


Supercase 2

Taxi-time prediction

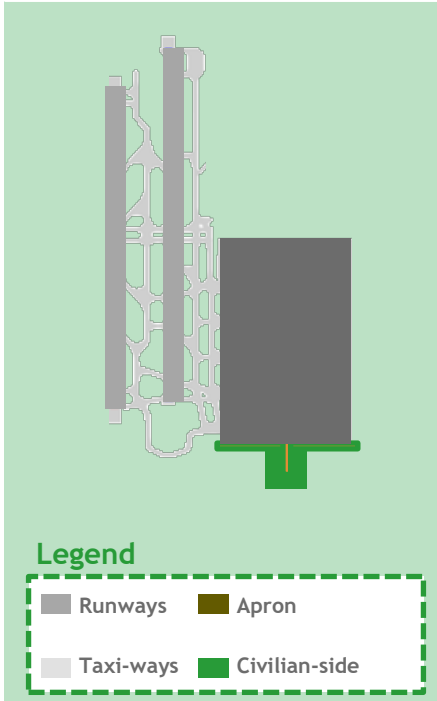
To the attention of Data Science for Business' master students

September 29th, 2020

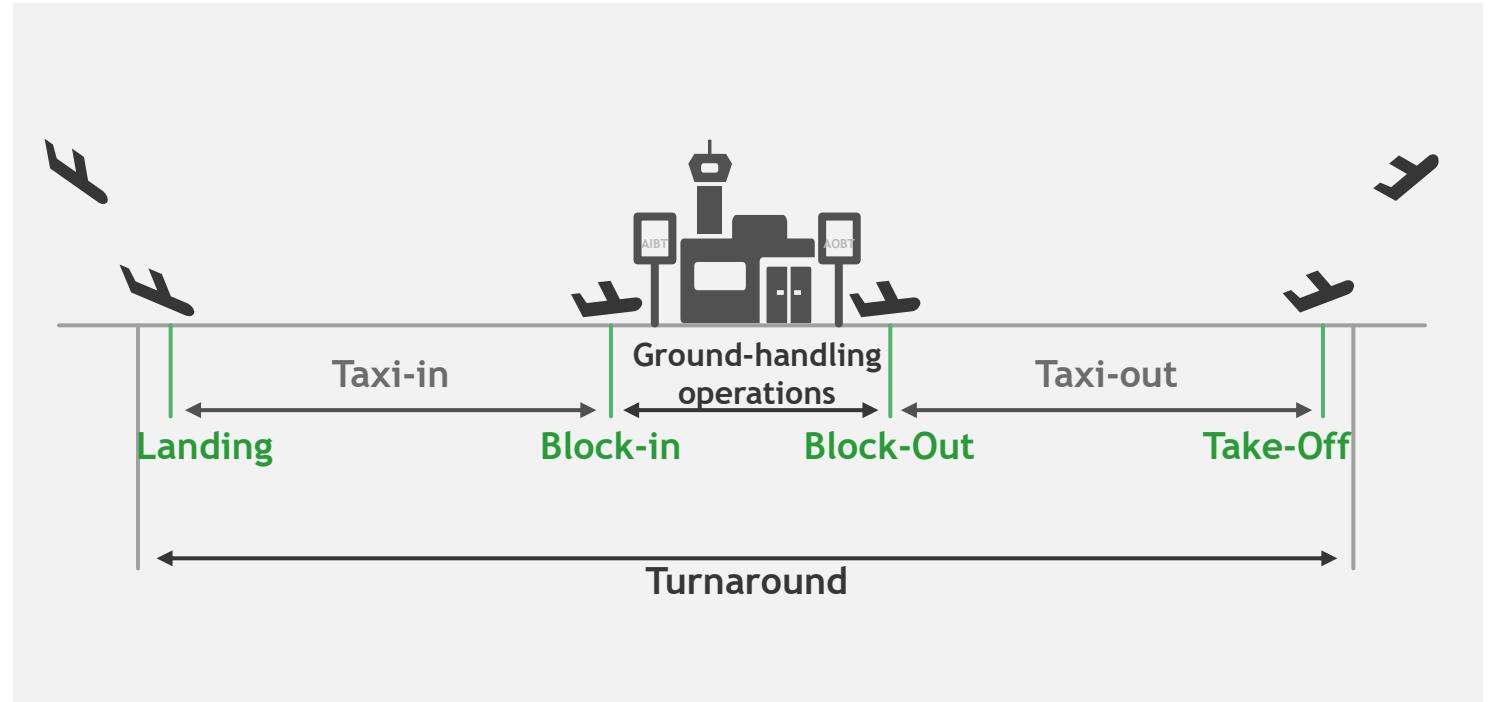


Take-off time (TOT) prediction - Airport overview

Airport overview



Turnaround process breakdown



TAXI-TIME PREDICTION CAN IMPROVE AIRPORT AND AIRLINES OPERATIONS AND REVENUES AS WELL AS REDUCE OVERALL GHG EMISSIONS

Take-off time (TOT) prediction - Use case description

TOT PREDICTION



Taxi-time definition

The taxi-time is the time an airplane spends “driving” on the ground:

- **Taxi-in** is the time window between the moment the airplane’s wheels touch the ground i.e. the **Actual Landing Time (ALDT)** and the moment it arrives at its assigned dock i.e. **Actual In-Block Time (AIBT)**
- **Taxi-out** is the time window between the moment the airplane starts moving from its dock i.e. **Actual Off-Block Time (AOBT)** to the moment its wheels leave the ground i.e. **Actual Take-Off Time (ATOT)**



Use case description

- Provide an **accurate Take-Off Time (ATOT) prediction** based on an actual off-block time (AOBT) and an **algorithm-based taxi-out time prediction** considering factors such as **airport configuration, AC type, weather...**



Status quo

- Currently almost every airport around the world is using a **moving average approach** to predict TOT: the airport assumes that the **taxi-out time for a given day will be equal to the average of taxi-outs during the past two months**



Customer type & examples



Airlines



Ground handlers



Airports



User



Air Traffic Controllers



Operation center



Key Expected Benefits



Know more accurately **when an aircraft will be airborne**



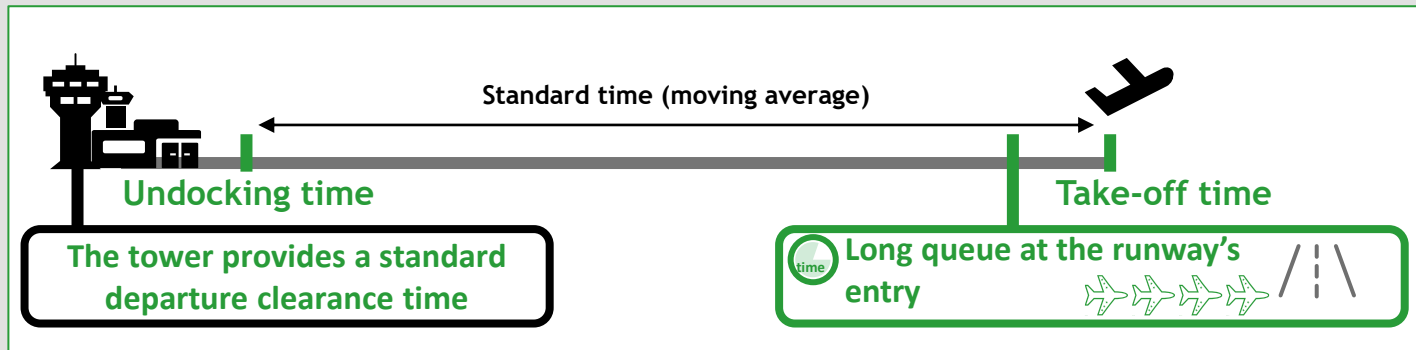
Reduce GHG emissions resulting from airplanes’ idle time at the runway entrance



Optimize ground movement and airport flow

Take-off time (TOT) prediction - Use case description

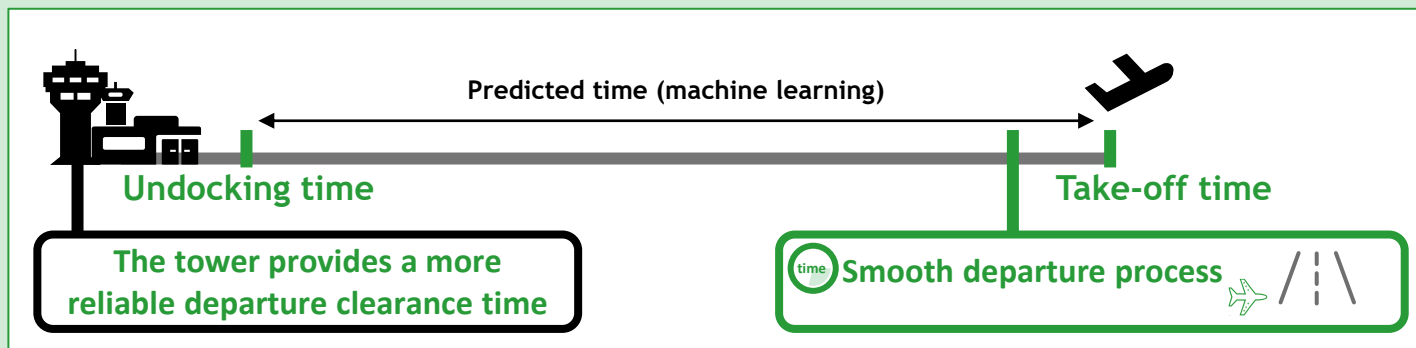
Status quo



Awareness

Efficiency

A.I. solution



Safety

Green

Expected output

- A **PowerPoint presentation** that should include at least the following:
 - ✓ A presentation of your **models' results** and how they **compare** to the **status quo** (the moving average)¹
 - ✓ An explanation of the **expected impact** of your best model **on ground operations** at the airport
 - ✓ A **final assessment** of your models by using **interpretability** methods
- Your **code** which should include:
 - ✓ Your **feature engineering code** specifying how you modified your data and why (make sure to **clearly comment your code** to explain why you processed the data the way you chose to)
 - ✓ Your models' parametrization, training code and testing code

PRESENTATION



CODE

```
# Get parameter of vehicle to consider for the selected context
engine = create_engine('mysql+pymysql://root:root@localhost:3306/eleven')
query = "SELECT v.id, v.equipment_plant_number FROM equipment_contexts_voc_vehicles v as v, ref_vehicle v as v WHERE v.vehicle_to_consider = v.id"
vehicle_to_consider = pd.read_sql_query(query, engine) # DataFrame with 'id' and 'equipment_plant_number'

vehicle_to_consider_id = str(tuple(list(vehicle_to_consider['id'])))
vehicle_to_consider_name = str(tuple(list(vehicle_to_consider['equipment_plant_number'])))

print("Vehicle to consider for the context " + str(vehicle_to_consider_id) + " in the batch " + str(vehicle_to_consider_name))
print(str(vehicle_to_consider_id) + " " + str(vehicle_to_consider_name))

# Import DRIVERS LOGS from RDD logs with INPUT = (vehicle_to_consider, startDate, endDate)
if params["id_log"] == "Apprenti":
    engine = create_engine('mysql+pymysql://root:root@localhost:3306/eleven')
    query = "SELECT log_id, log_driver_id, d.name, d.equipment_plant_number, log.action, log.date FROM app_log_drivers_data d, log_driver d WHERE d.id = log_driver_id AND log_driver_id = log_driver_id AND log_driver_id = log_driver_id"
    drivers_data = pd.read_sql_query(query, engine) # DataFrame with 'id', 'equipment_plant_number', 'action', 'date'
    drivers_data['action'] = drivers_data['action'].apply(lambda x: int(x))
    drivers_data = drivers_data[drivers_data['equipment_plant_number'] == vehicle_to_consider_name]
    drivers_data = drivers_data[drivers_data['date'] > startDate & (drivers_data['date'] <= endDate)]
    drivers_data['driver_name'] = drivers_data['equipment_plant_number'] + " " + drivers_data['name']

    id_to_consider = str(tuple(list(set(drivers_data['driver_id'])))
    query = "SELECT name, equipment_plant_number FROM ref_drivers WHERE id IN " + str(id_to_consider)
    drivers_data = pd.read_sql_query(query, engine) # DataFrame with 'id', 'equipment_plant_number', 'action', 'date'
    drivers_data['driver_name'] = drivers_data['equipment_plant_number'] + " " + drivers_data['name']

    drivers_data = drivers_data[drivers_data['action'] == 'id']
    print("Drivers data DataFrame with driver log data")
    print("----- Number of rows : " + str(drivers_data.shape[0]))
    print(drivers_data.head())
    print(drivers_data.shape)
    print(set(drivers_data['equipment_plant_number']))
    print(set(drivers_data['equipment_plant_number']))
    print(set(drivers_data['driver_name']))
    print(set(drivers_data['driver_name']))

if params["id_log"] == "Pilotage":
    # ... (code continues) ...
```

Provided input

- **An airport terms glossary:** Glossary > Glossary.xlsx
- **Historical airport and weather data :**
 - *Airport data:*
 - ❑ Data > Airport data > training_set_airport_data.csv
 - ❑ Data > Airport data geographic_data.csv
 - *Weather data:*
 - ❑ Data > Weather data > training_set_weather_data.csv
- **Academic papers on the taxi-time prediction subject:** Taxi time academic papers > Paper 1.pdf...Paper 6.pdf
- **Aircraft (A/C) types' characteristics:** AC characteristics > ACchar.xlsx
- **A test set:** Test set - this folder contains weather data, airport data and geographical data for your model testing

AIRPORT & WEATHER DATA

[illegible]

GLOSSARY

[illegible]

TEST SET

[illegible]

RESEARCH PAPERS

A/C TYPE CHARACTERISTICS

The screenshot displays a complex financial model spreadsheet. The main table has columns for various financial metrics, including Revenue, Costs, and Profit. The rows are organized into sections, with some cells highlighted in orange and others in green. The spreadsheet is titled 'Financial Model' and includes a 'Summary' tab. The data is organized into multiple sections, with some cells highlighted in orange and others in green. The spreadsheet is titled 'Financial Model' and includes a 'Summary' tab.

APPENDIX



Example of model performance comparison sheet

Chosen models' description and performance overview

		IBT Prediction			
		Rolling average	Model 1	Model 2	...
Key elements		<ul style="list-style-type: none">Average of the last 2 months of taxi-times	<ul style="list-style-type: none">Model details:Used features:<ul style="list-style-type: none">○ ...○ ...	<ul style="list-style-type: none">Model details:Used features:<ul style="list-style-type: none">○ ...○ ...	<ul style="list-style-type: none">Model details:Used features:<ul style="list-style-type: none">○ ...○ ...
Results	Average error vs. real data	2.8 min	xx min	xx min	xx min
	First quartile max error	1.7 min	xx min	xx min	xx min
	Third quartile max error	3.9 min	xx min	xx min	xx min