Report

Assignment 2: Spatio-temporal Data

Flight Data Analyst

CSC 501

University of Victoria

Submitted to:

Prof. Sean Chester

Submitted by:

Anushka Halder (V00961967)

Sanjana Arora (V00966221)

Tavanpreet S. Oberoi (V00963163)

1. Introduction

A geographic information system (GIS) is a computer system that collects, saves, verifies, and displays data about locations on the Earth's surface. By connecting seemingly unconnected data, GIS may help individuals and organizations better understand spatial patterns and relationships.

In this assignment, we are focusing on the dataset given by general aviation data which contains two types of data: Weather data (weather_data.zip), containing various weather measurements at Pittsburgh-Butler Regional Airport station, and flight data (111_days.zip), containing various flight data over 111 days. In this we only considered the raw data and discarded the pre-processed data. The objective of the analysis is to identify the nearest neighbor flight (spatial co-ordinates) operating at the same instance (time) based upon different weather attributes (temperature, wind speed, weather type). It helps in controlling and planning the air traffic, simultaneously minimizing the probability of the collision.

There are total of 111 files for 111 days which are merged into single ***'Out.csv'*** file which has been used in our data analysis. The file 'Out.csv' is then joined with file 'weather.csv' on field 'metar' to perform the exploratory data analysis (EDA). As observed from the given dataset, the data size is humongous and contains spatial-temporal data. Hence, it is important to design a model that supports the spatial-temporal aspect of the data keeping scalability and time complexity of the model in mind.

In the below report, there is a discussion on conceptual and logical model keeping the scalability factor in the process. There is thorough discussion why vector data structure is preferred over raster data structure. Under the logical modelling section, there is discussion on selecting the right database among R-tree, KD-Tree and Quad-Tree. At the end, scalability of the model is compared with respect to RDBMS (Relational Database Management System) – SQLite 3 performance and time execution to do EDAs.

2. Data Modelling

2.1 Conceptual Data Model

It can be seen from the given dataset that it has both spatial and temporal attributes, hence it is required to model data that supports both spatial-temporal modelling.
Apart from Spatial-temporal attributes (Dataset File 1), the dataset also contains some weather and temperature related attributes (Dataset File 2), both are connected using 'Metar' attribute.

Problem Statement: It is a situation of finding the nearest neighbor. It is important to identify the spatial location of multiple aircrafts based on different weather conditions to minimize the collision percentage and for the smoother air traffic movement. It is also important to analyze at what time of the day, more aircrafts can be allowed to be flown from the airport.

The Spatial attributes includes 'Latitude', 'Longitude', 'Altitude', 'Angle', 'Bearing', 'Heading' etc. while temporal attributes include 'Date' and 'Time'.

There are possibly two types of data structures which can be implemented on Spatial dataset (GIS). The foremost is Vector and the other is Raster. The vector representation of the model is shown in Figure 2.1. In our scenario vectors are preferred over raster for following reasons:

- Vectors are composed of points, lines and polygons and the given dataset can be model based on these three attributes. We considered 'Polygon' as Airport Area, 'Lines' as Airplane Trajectory and 'Points' are the co-ordinates (Spatial attributes) of the Airplane at one instance of time.

- Vector data can easily render geographic features with great decision, but it also cost high processing time and complex data structures.
- Each node, label, vertex, line is stored with its own attributes, hence makes model complex, but best suitable when dealing with geospatial data.
- Raster datasets are composed of rectangular arrays of regularly spaced square grid cells. It is usually preferred when dealing with 'Continuous' or 'Image' dataset. Since our dataset is neither of them, so is not suitable to use Raster Modelling.
- In Raster data structures, grids are formed which aren't required in our data modelling.
- The goal of our modelling is to find the nearest neighbor point (another airplane) to reduce the chance of collision while allowing maximum traffic to flow, hence raster data structure won't yield better results as those works well with continuous data.
- In Raster datasets, cell size remains constant, hence it does not have absolute co-ordinate's locations, requiring more computation and storage to calculate any decisions (nearest neighbor in this case).

In the conceptual model for airport dataset – Figure 2.2, it can be seen that trajectory of an aircraft can be found using *'Lines'*, spatial attributes like longitude, latitude, angle, heading etc. are represented by *'Points'* which helps in identifying the position of an aircraft at particular instance of a day, and the runway region can be denoted by *'Polygon'*.

The model has been designed such that it can easily be expanded if more spatial attributes be added to the given dataset. The modelling is not done using raster data structure, hence worrying about size of the grid when new data is added is not the matter of concern.

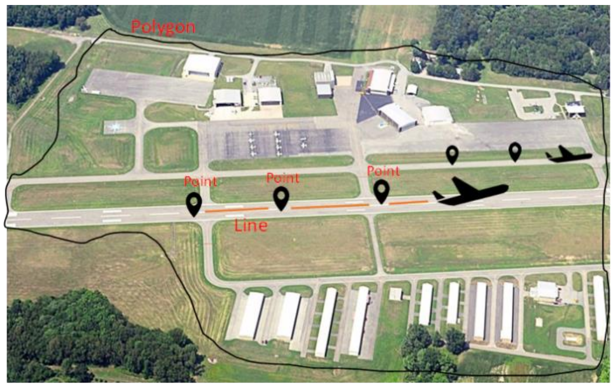


Fig. 2.1. Vector Representation of the model Dataset

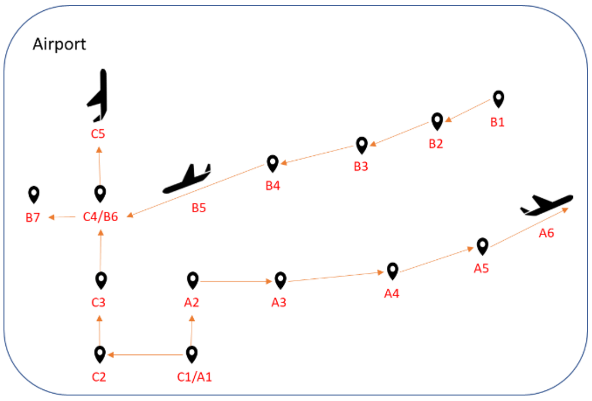


Fig. 2.2. Conceptual model for Airport

2.2 Logical Data Model

Logical Data Model add details about data structures being used when implementing the model. The three common data structures used with spatial data, each having its own advantages over other are:

- R-tree
- Quad-Tree
- KD-Tree

KD-trees are space portioning data structures for representing points in K-Dimensions. They are a type of data structure used to efficiently describe our data. KD-trees aid in the organization and partitioning of data points based on specified parameters. For our dataset, we have represented the data using KD-Tress and used it for query retrieval. It is preferred over R-Tree and Quad Tree for following reasons:

- R-Tree partition the data into rectangles (Area), while KD-Tree do binary split (Points). Binary Split points are also disjoint while R-Tree could have overlapped area.
- R-trees can hold rectangles and polygons, whereas k-d-trees can only hold point vectors (as overlap is needed for polygons) which is required in our case.
- Quad-tree always split data in all the dimensions, which increases the computation and empty cells in the data structure. Since our spatial dataset has 6 dimensions, so it will take lot of time to form a tree with many empty child nodes.
- The problem is to find nearest neighboring co-ordinate to specific co-ordinates, which is most efficient in KD-Trees as it doesn't have any empty cells as in quad tree.
- Searching in KD-Tree is easier as compared to quad tree as the closest object might be placed right on the other side of a division between nodes. KD-trees on the other hand, allows implementation of very efficient nearest-neighbor search.

While implementing KD-Tree, temporal dimension has been converted to numerical dimension, so that it can be considered as attribute. For converting the temporal data, differential time in ns (nanoseconds) are considered as numerical dimension. It also helps in saving the space in the model.

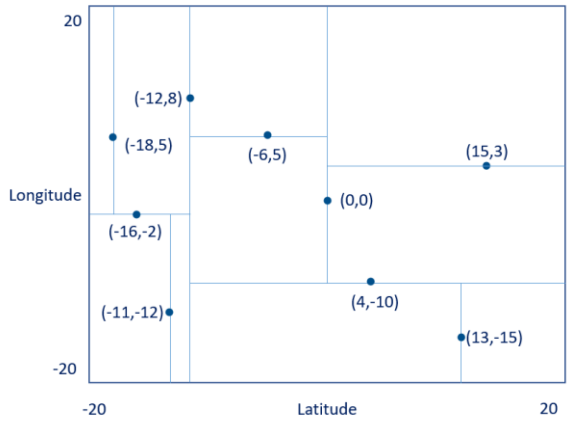


Fig. 2.3. Logical Data Model (considering 2-D) for Airport Dataset (KD-Trees)

The above diagram in Fig. 2.4. gives an overview of Logical Data Model for the given dataset. It has been displayed in 2-D, but when implementing 6 spatial dimensions are considered. In KD-Tree, alternatively tree split on each dimension as it can be seen in Fig. 5 below. Using binary search in 2-D case and exhaustive search in N-D case, KD-trees are useful to find the nearest neighbor, also allowing to limit the tree based on ranges. Therefore, below logical model has been implemented in our analysis.

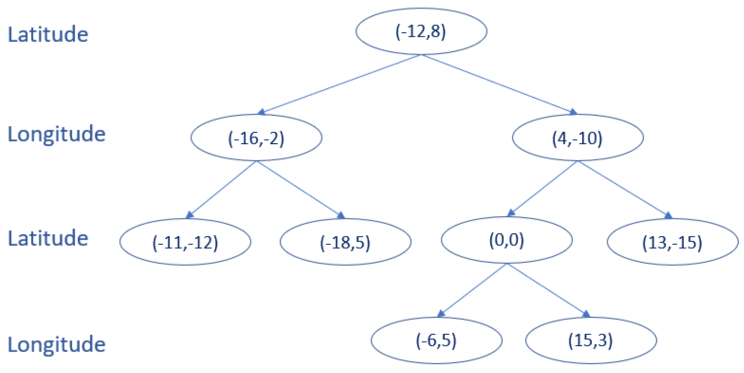


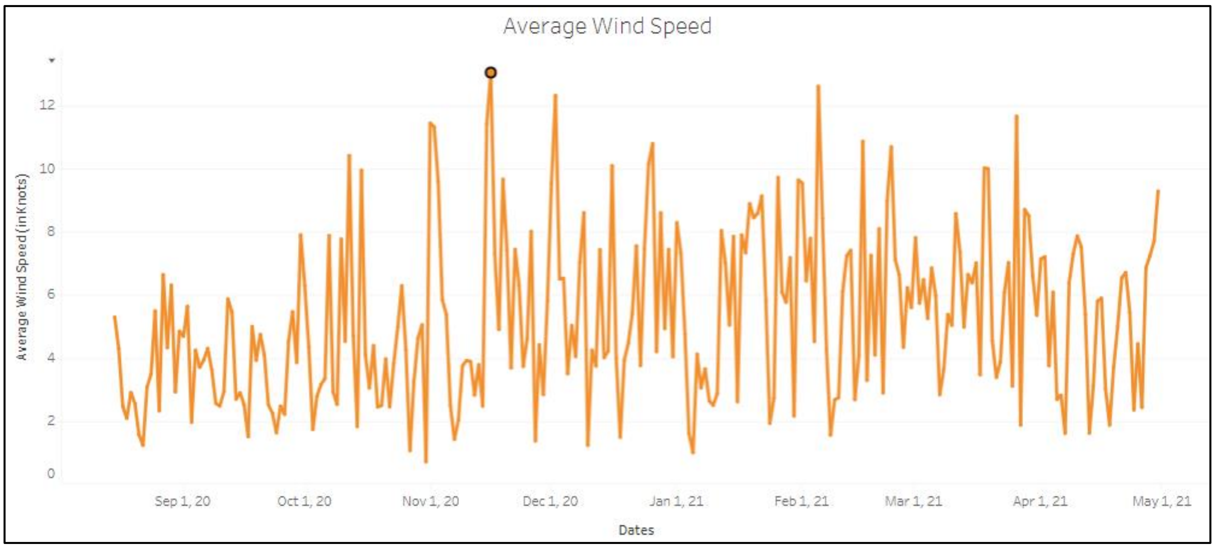
Fig. 2.4. KD-Tree Representation for 2-D (Latitude and Longitude)

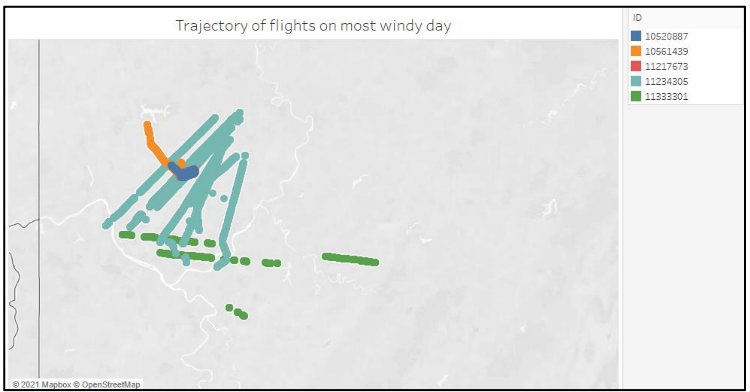
3. Exploratory Data Analysis

PLOT-1, 2& 3: <u>Most windy day and number of flights on that day</u>

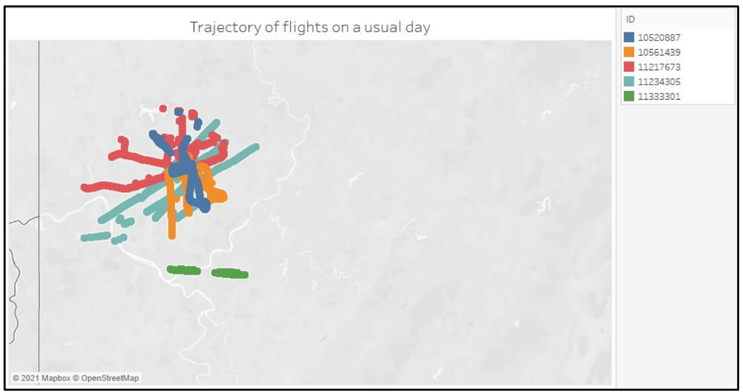
FILES OR TABLES USED – weather.csv and out.csv
We have used the resample function of pandas to find average wind speed over the provided weather dataset. As shown in graphs below, we identified November 16, 2020 to be the windiest day and a normal day like March 20, 2021. We then query the flight dataset using these dates and try to visualize the effects of wind speeds on the flight trajectories.

INSIGHTS – The comparison of the graphs show that numbers of flights on a normal day(Fig 3.2) were higher than those on the windiest day(Fig 3.1). This fact aligns well with the fact that it's relatively safer to fly on a normal day than on a day which is windy more than average. Further, comparison of graph also shows change in trajectories of the same flights flying on the two dates. There seems to be a lot of changes in the flight trajectory due to the high wind speed on November 16, 2020 and probably only the most important flights were flown that day.





Trajectory of flights on most windy day
ID
10520887
10561439
11217673
11234305
11333301
© 2021 Mapbox © OpenStreetMap



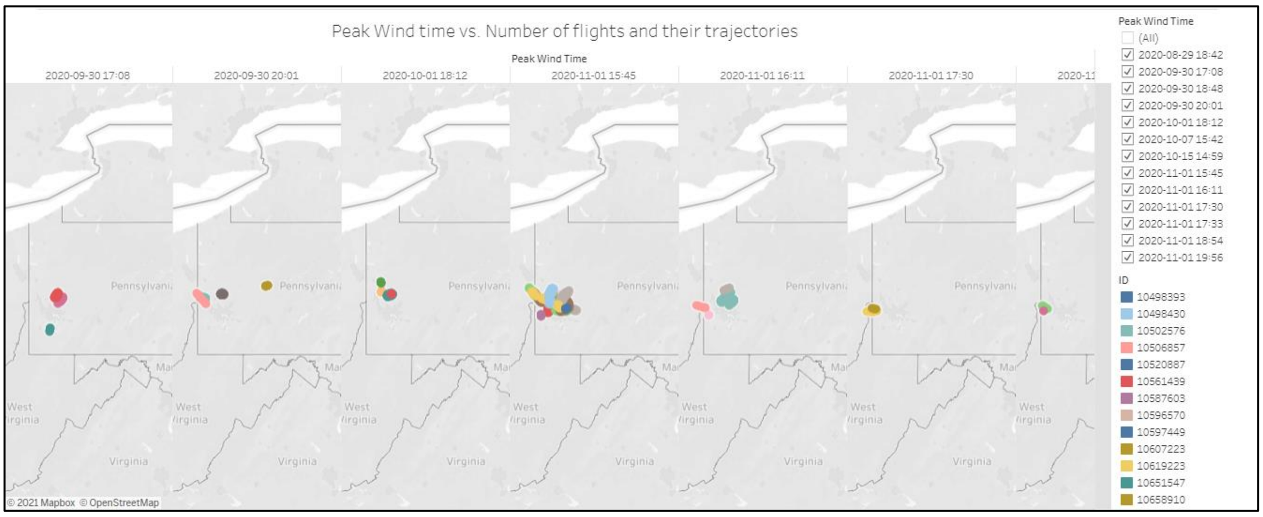
Trajectory of flights on a usual day
ID
10520887
10561439
11217673
11234305
11333301
© 2021 Mapbox © OpenStreetMap

Fig 3.1 Fig 3.2

PLOT-4: <u>Flights at the peak wind times of different days</u>

FILES OR TABLES USED – weather.csv and out.csv
We have used Tableau relationships on weather and out tables on Metar column to visualize the relationship between the two datasets and plot the number of flights and their trajectories at the peak wind times of the day.

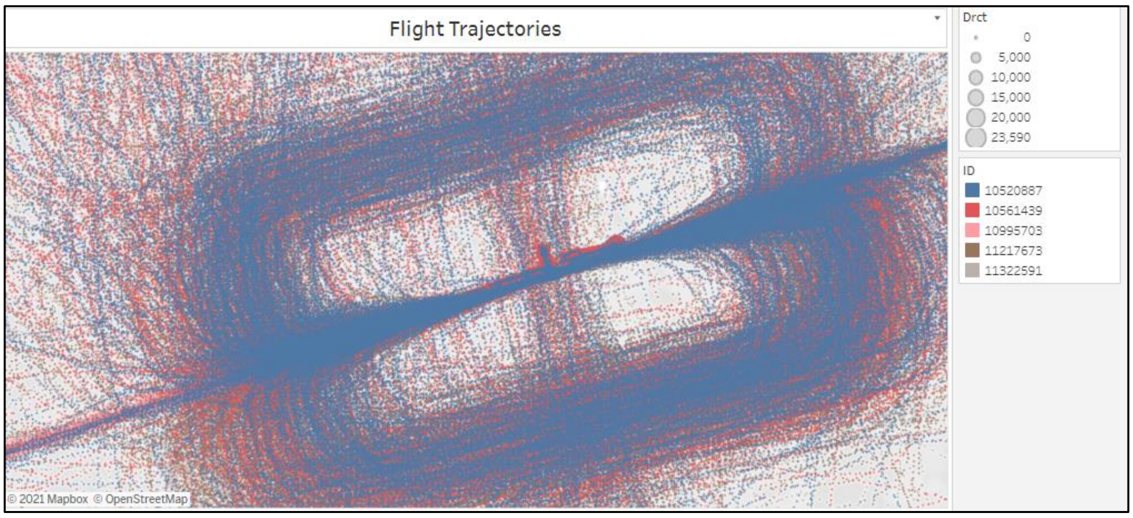
INSIGHTS – From the visualization we see that the maximum number of flights fly on November 1, 2020 even during peak wind time. It was observed that all dates other than November 1, 2020 were weekdays and since, relatively more number of flights fly on weekends, the graph quite rightly reflects the same. November 1, 2020 was a Sunday and most flights flew that day even at peak wind times. It could be concluded that either the flights got cancelled at the peak wind times on other dates or coincidently less number of flights fly on weekdays.



PLOT-5: <u>Flight Trajectories</u>

FILES OR TABLES USED – out.csv
We have used the longitude, latitude, altitude, direction and heading attributes of the flight data to prepare the flight trajectories.

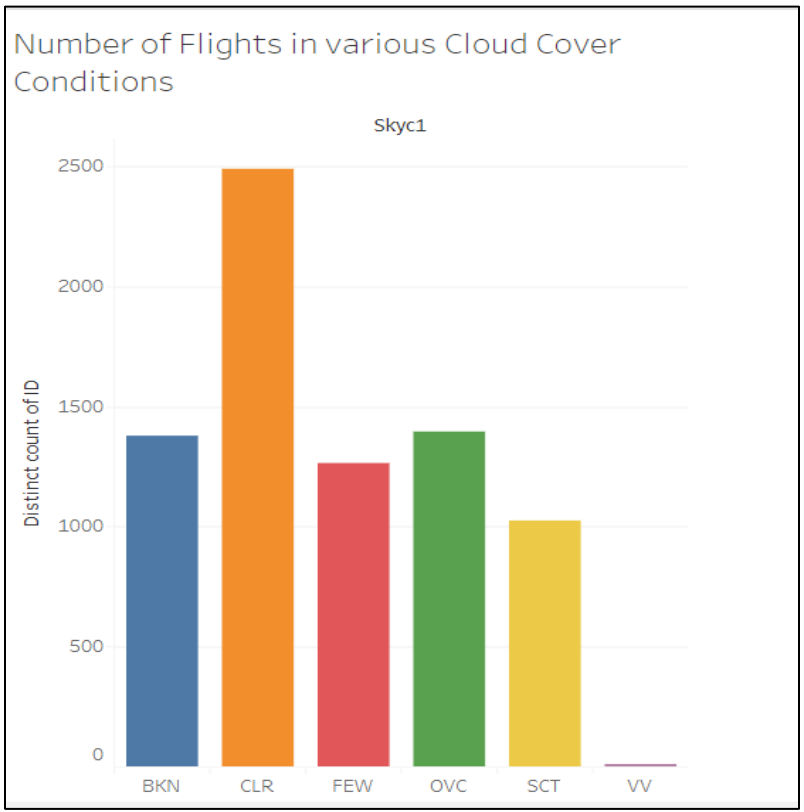
INSIGHTS – The visualization below quite closely represents the trajectories followed by various aircrafts while landing and taking off. This graph quite interesting shows how each of the aircraft follow mostly the same trajectory for each flight.



PLOT-5: **Number of flights in different Cloud Cover Conditions**

FILES OR TABLES USED – out..csv and weather.csv
We have used relationship between flight data and weather data based on METAR column. We have used the ID column of the flight data and the skyc1 of weather data.

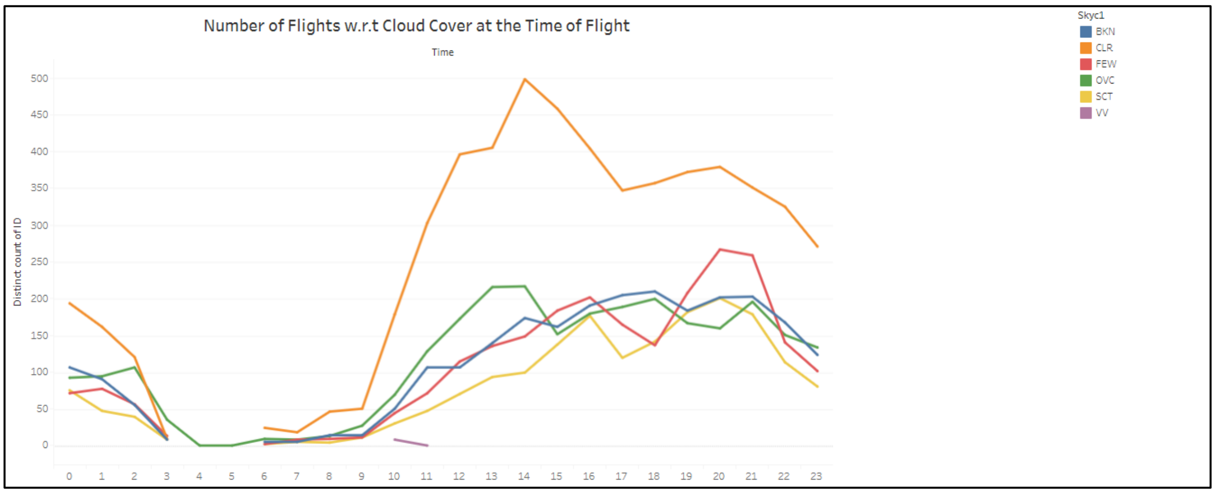
INSIGHTS – The cloud cover is described by a unit called okta that varies from clear to overcast. We see that in clear cloud conditions, the maximum number of flights operate. In other unclear cloud cover conditions, the count reduces by almost half. In cases of obscure visibility, airports might only want to run flights with extremely qualified pilots who can drive safely in unclear conditions. Also, there is a possibility of landing the flight at other nearby destination in case of blurred visibility.



PLOT-6: **<u>Number of flights according to time of flight and amount of cloud cover</u>**

FILES OR TABLES USED – out.csv and weather.csv
We have used relationship between flight data and weather data based on METAR column to identify the times when airport is busy the most at different cloud covers.

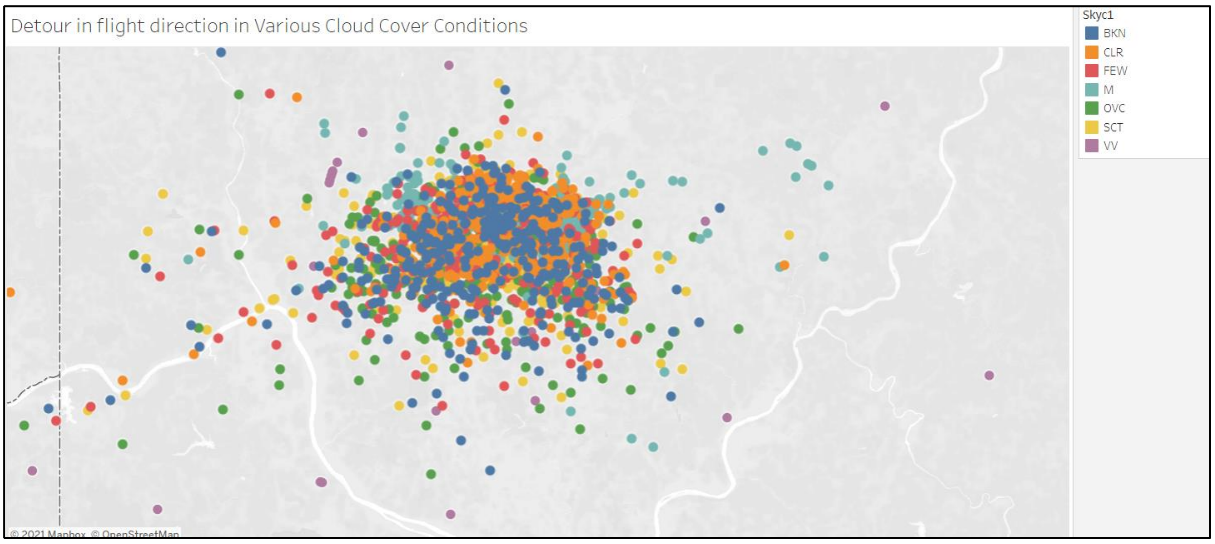
INSIGHTS –From the visualization it can be easily concluded that highest number of flights fly on clear weather days between the time of 2:00pm to 5:00pm. Very few flights fly from 2:00am to 6:00am in the morning. The graph also indicates that least number of flights fly in scattered cloud conditions.



PLOT-7: **<u>Detour in direction of flights in different Cloud Cover Conditions</u>**

FILES OR TABLES USED – out.csv and weather.csv
We have used relationship between flight data and weather data based on METAR column . We have used the heading column of the flight data to determine the change in angle of the flight and the skyc1 of weather data for the cloud cover conditions.

INSIGHTS – This visualization supports the previous bar chart showing the direction of flights in all the cloud cover conditions.


Detour in flight direction in Various Cloud Cover Conditions
Skyc1
BKN
CLR
FEW
M
OVC
SCT
VV
© 2021 Mapbox © OpenStreetMap

We see that in clear conditions(Fig3.3), flights follow the usual path and seldom deviate from their direction. However, in unclear conditions like overcast(Fig 3.4), the number of flights are less in number and also many flights detour and possibly land at nearby airports.

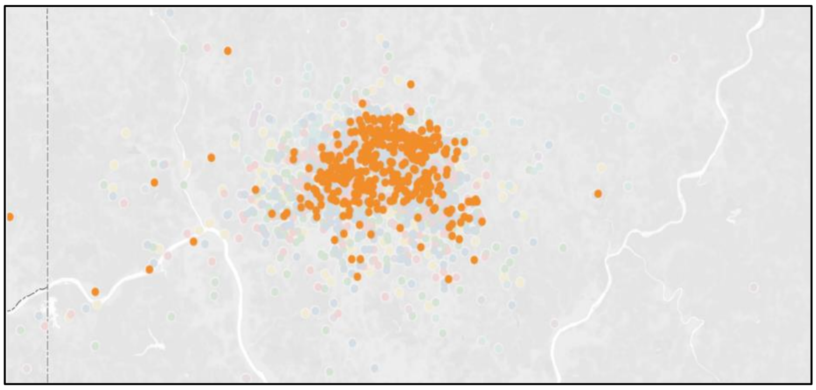


Fig 3.3

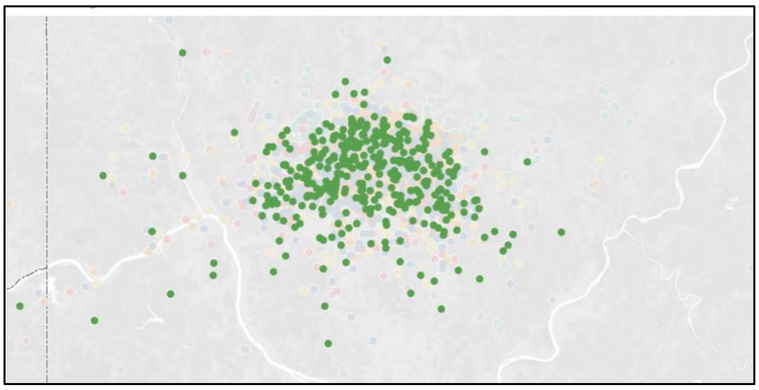
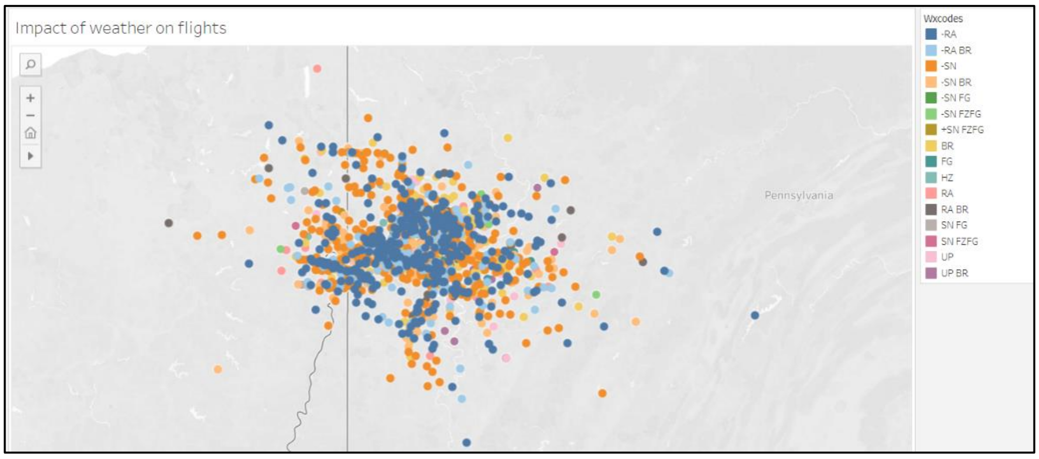


Fig 3.4

PLOT-8: Impact of weather on the flights

FILES OR TABLES USED – out.csv and weather.csv

We have used relationship between flight data and weather data based on METAR column .We have used the ID column of the flight data to determine the different flights and wxcodes of weather data for the weather conditions.



INSIGHTS – This visualization show the number of operating flights in each weather condition.
We can see that in rainy(Fig 3.5) or snowy(Fig 3.6) weather there are many flights operating as the visibility in these conditions are clear for majority of the flights to operate.

Fig 3.5

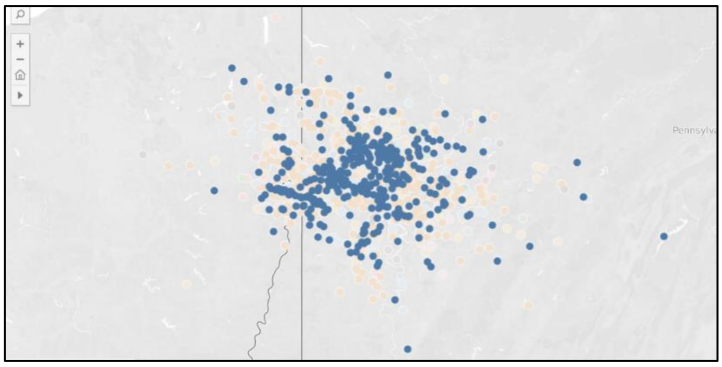
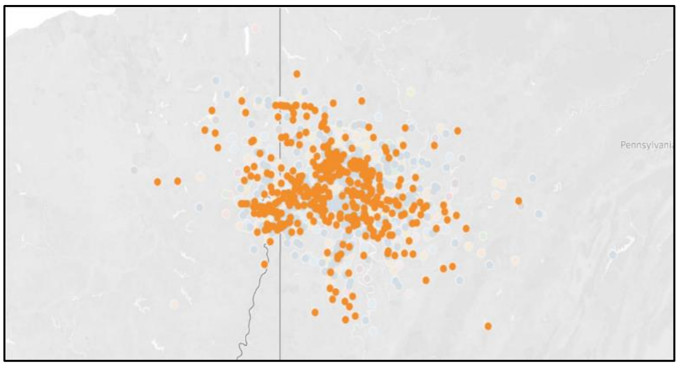


Fig 3.6



However, whenever there is fog(Fig 3.7) or haze(Fig 3.8) the visual clarity reduces resulting in a decrease in the number of operating flights. In such conditions, aviation department allows only well experienced crew members to fly the aircrafts. Also, durable aircrafts like Boeing and Airbus would be preferred in these weather conditions as compared to smaller aircrafts.

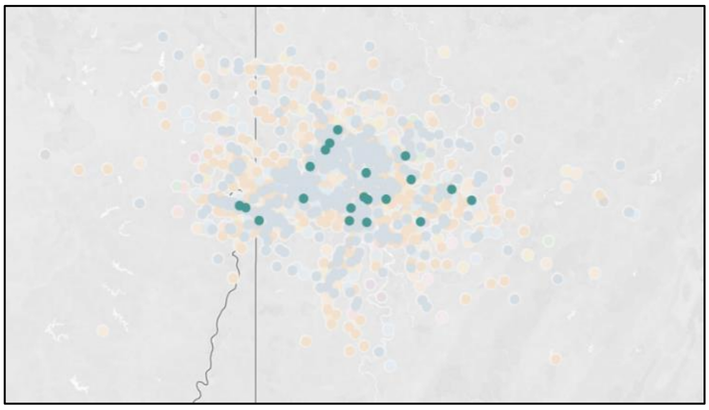


Fig 3.7

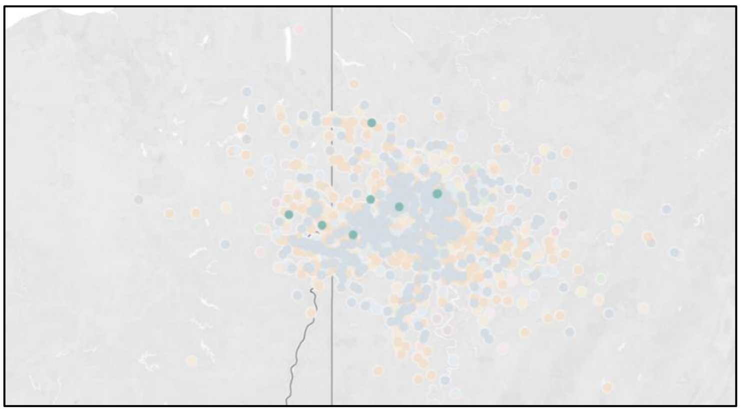


Fig 3.8

4. Scalability

Processor – Intel(R) Core(TM) i7-4510U CPU @ 2.00GHz   2.60 GHz

RAM – 8GB

Tool/software – Jupyter Notebook and Google Colab

Language – Python 3.7 and SQLite3

<u>Resource Consumption</u>

Google colab provides each user with 61.54 GB memory. For comparing the resource consumption by the original data files and our data model created on the SQL database, we downloaded the tables into .csv format. The Figures 4 and 6 below show the RAM & Disk consumption with the original five files were being used in Google colab. While, Fig.5 and Fig.7 show the amount of RAM and disk consumption when the 4 files of the tables that we created. As can be seen from comparison of these images, the original data files are consuming much more RAM than the data model we have created. Further, as seen from Fig.5 and Fig.7, the amount of memory available after the original .csv files is 60.16 GB, while it is 61.48 GB with the .csv files of the tables modeled. Therefore, the fetching from tables modeled is much more efficient in terms of memory and resource consumption as compared to the original dataset.

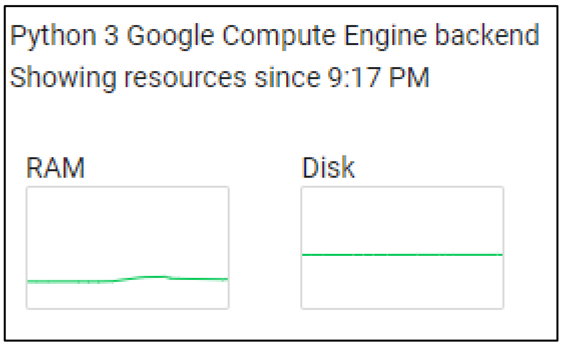

Python 3 Google Compute Engine backend
Showing resources since 9:17 PM

Fig.4. RAM and Disk for original data

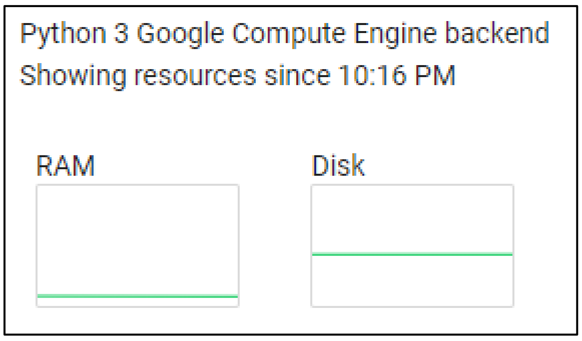

Python 3 Google Compute Engine backend
Showing resources since 10:16 PM

Fig.5. RAM and Disk for our tables

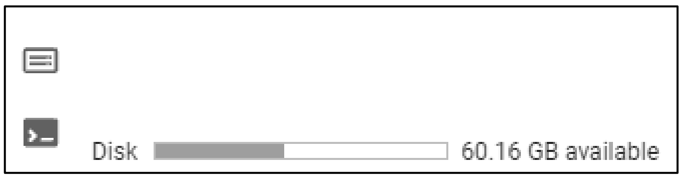

Disk 60.16 GB available

Fig.6. Memory available after original data

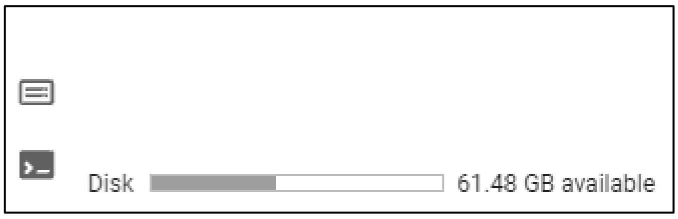

Disk 61.48 GB available

Fig.7. Memory available after our tables

Removing Data redundancy

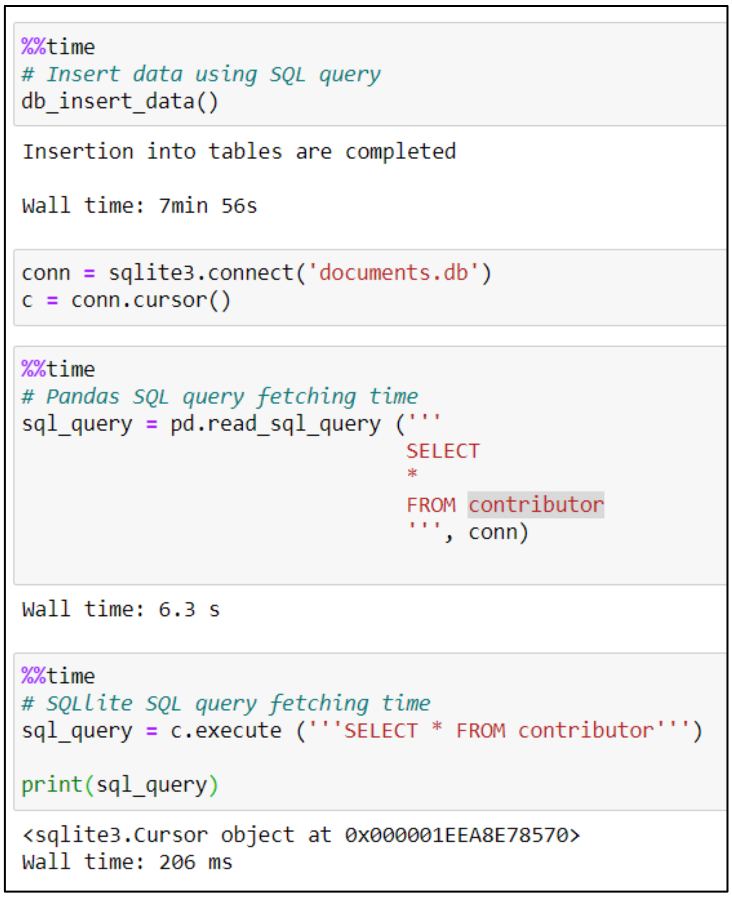
Further, if we would have stored the area related details of each data file in separate tables, the number of rows being used only by the area related details is **228141.** Our data model allows storing the area information of all the five data files in one table. The number of rows fetched from the area table is **195251** and hence, the data model has reduced data redundancy by storing only the distinct area details because of the presence of postal code attribute.

Data Scalability wrt. Data Analysis

As also captured in the Exploratory Data Analysis section of this report, we wanted to capture the analysis of contributions made for all the available data from 1993 to present. Hence, the data model is built such that it does not require separate tables to be created for a specific year range. Whenever, some new data needs to be added to this data model, we could easily do that by adding the data to these tables with respect to the attributes. Further, incase new data comes with some new attributes; those attributes could be easily included by creating more tables and relating those with the current tables.

Performance

As shown in the Fig.8, the amount of time taken to access the tables using pandas is 6.3 secs, while it is 206ms by using the SQL query. Hence, we preferred using the SQL queries for further calculations of our data analysis.



```
%%time
# Insert data using SQL query
db_insert_data()
```

```
Insertion into tables are completed

Wall time: 7min 56s
```

```
conn = sqlite3.connect('documents.db')
c = conn.cursor()
```

```
%%time
# Pandas SQL query fetching time
sql_query = pd.read_sql_query ('''
                                SELECT
                                *
                                FROM contributor
                                ''', conn)
```

```
Wall time: 6.3 s
```

```
%%time
# SQLlite SQL query fetching time
sql_query = c.execute ('''SELECT * FROM contributor''')

print(sql_query)
```

```
<sqlite3.Cursor object at 0x000001EEA8E78570>
Wall time: 206 ms
```

Fig.8. Python pandas vs. Python SQL query