Dataset link:

https://www.kaggle.com/datasets/usdot/flight-delays?select=flights.csv

This dataset contains information about all flights that

departed from NYC(e.g. EWR, JFKand LGA)in 2013: 336,776 flights in total.

Assumptions:

Values for the time columns are represented in minutes and distance column is represented in miles.

Pre-Conditions:

\* Importing required packages - Numpy, Pandas, Matplotlib, Seaborn, PandasProfiling(if required)

\* Read all the contents of the file.

Analysis:

\* Flight airtime will depends on the distance between the source and destination airports.

\* Field ““sched\_dep\_time“ & “dep\_time” will be completey depends on the “hour” field.

\* Hour and dep\_time, scheduled\_dep\_time are highly correlated.

\* Distance and air\_time are highly correlated

\* Arr\_delay and dep\_delay are also highly correlated

\* dep\_delay is highly correlated with arr\_delay

\* distance is highly correlated with air\_time

\* hour is highly correlated with dep\_time

\* sched\_dep\_time is highly correlated with hour.

Smalll Hint:

1. Find-out and fill the the missing values .

Drop 'tailnumber' and 'year' fields as there will be not used for the Explotary data analysis.

Day-wise planning:

1. Required Software installation:

Python,pysaprk,Google-colab,sql,Django.

2. Data set underestanding:

3. Find solutions using python & pyspark both.

3. dataset loading into the database

4. load data from database to pyspark ,python dataframe

5. Perform technical analysis - finding null and missing values

6. Perform data visualition and data summary

7. check data-type of each column.

8. Perform explatory data analysis(EDA)

Business questions:

Pritam Das:

1. Destination with maximum departure delays?

2. Destination with minimum departure delays ?

Hint for 1 & 2:

Calculate number of origin & destination point.

calculate mean and compare flight delay with some thresold value like 200

Display graph and get the maximam & minimum delays or calculate the maximam and minimum flight deplays using API

3. Delay calculations at the origin locations ?

Hint: perform group by and aggregate with mean values

4. Month during which Maximum Delays observed ?

5. Month during which Minimum Delays observed ?

Hint for 4 & 5:

get the dataset by performing group by on 'airline' column and aggregate with mean value and plot graph to get month

Rishav Viabhav:

6. Airline with highest and lowest mean speed ?

Ans :

To find the airline with the highest and lowest mean speed using Python, PySpark, and Google Colab, you can use the following steps:

1. First, you will need to gather data on the flights and their speeds. You can use a dataset from a database, such as a SQL database, or you can scrape data from a website using a tool like Django.
2. Once you have the data, you can use PySpark to perform distributed data processing and analysis.
3. To find the mean speed for each airline, you can use PySpark's **groupBy** and **avg** functions to group the flights by airline and calculate the average speed for each group.
4. You can then use PySpark's **sort** function to sort the airlines by their mean speed.
5. Finally, you can use Python's **print** function to print the airline with the highest and lowest mean speed.

Here is some example code that demonstrates these steps:

Copy code

# Import PySpark and create a SparkSession

from pyspark.sql import SparkSession

spark = SparkSession.builder.appName("Airline Speed").getOrCreate()

# Load the data into a PySpark DataFrame

df = spark.read.csv("/path/to/data.csv", header=True)

# Group the data by airline and calculate the mean speed for each group

mean\_speeds = df.groupBy("airline").avg("speed")

# Sort the airlines by mean speed

sorted\_speeds = mean\_speeds.sort("avg(speed)", ascending=False)

# Print the airline with the highest mean speed

print("Airline with highest mean speed:")

print(sorted\_speeds.first())

# Print the airline with the lowest mean speed

print("Airline with lowest mean speed:")

print(sorted\_speeds.last())

Note that this code assumes that your data is in a CSV file with a header row, and that the column containing the airline names is called "airline" and the column containing the speeds is called "speed". You may need to modify the code to match the structure of your data.

1. Airports most and least busy

Ans :

To find the airports that are most and least busy using Python, PySpark, and Google Colab, you can use the following steps:

1. First, you will need to gather data on the flights and the airports they operate from and arrive at. You can use a dataset from a database, such as a SQL database, or you can scrape data from a website using a tool like Django.
2. Once you have the data, you can use PySpark to perform distributed data processing and analysis.
3. To find the number of flights for each airport, you can use PySpark's **groupBy** and **count** functions to group the flights by airport and count the number of flights for each group.
4. You can then use PySpark's **sort** function to sort the airports by the number of flights.
5. Finally, you can use Python's **print** function to print the airports with the most and least flights.

Here is some example code that demonstrates these steps:

# Import PySpark and create a SparkSession

from pyspark.sql import SparkSession

spark = SparkSession.builder.appName("Airport Busyness").getOrCreate()

# Load the data into a PySpark DataFrame

df = spark.read.csv("/path/to/data.csv", header=True)

# Group the data by airport and count the number of flights for each group

flight\_counts = df.groupBy("airport").count()

# Sort the airports by number of flights

sorted\_counts = flight\_counts.sort("count", ascending=False)

# Print the airport with the most flights

print("Airport with the most flights:")

print(sorted\_counts.first())

# Print the airport with the least flights

print("Airport with the least flights:")

print(sorted\_counts.last())

Note that this code assumes that your data is in a CSV file with a header row, and that the column containing the airport names is called "airport". You may need to modify the code to match the structure of your data.

1. Months During which airports are most and least busy

To find the months during which airports are most and least busy using Python, PySpark, and Google Colab, you can use the following steps:

1. First, you will need to gather data on the flights and their arrival and departure dates. You can use a dataset from a database, such as a SQL database, or you can scrape data from a website using a tool like Django.
2. Once you have the data, you can use PySpark to perform distributed data processing and analysis.
3. To extract the month from the arrival and departure dates, you can use PySpark's **date\_format** function and the **%m** format string, which represents the month as a two-digit number. You can then use the **month** column in your analysis.
4. To find the number of flights for each month at each airport, you can use PySpark's **groupBy** and **count** functions to group the flights by airport and month and count the number of flights for each group.
5. You can then use PySpark's **sort** function to sort the airports by the number of flights in each month.
6. Finally, you can use Python's **print** function to print the months during which each airport is busiest and least busy.

Here is some example code that demonstrates these steps:

# Import PySpark and create a SparkSession

from pyspark.sql import SparkSession

spark = SparkSession.builder.appName("Airport Busyness by Month").getOrCreate()

# Load the data into a PySpark DataFrame

df = spark.read.csv("/path/to/data.csv", header=True)

# Extract the month from the arrival and departure dates

df = df.withColumn("arrival\_month", date\_format("arrival\_date", "%m"))

df = df.withColumn("departure\_month", date\_format("departure\_date", "%m"))

# Group the data by airport and month and count the number of flights for each group

flight\_counts = df.groupBy("airport", "arrival\_month").count()

# Sort the airports by number of flights in each month

sorted\_counts = flight\_counts.sort("count", ascending=False)

# Print the busiest and least busy months for each airport

print("Busiest and least busy months for each airport:")

for airport, flights in sorted\_counts:

print(f"Airport: {airport}")

print(f"Busiest month: {flights.first()[1]}")

print(f"Least busy month: {flights.last()[1]}")

Note that this code assumes that your data is in a CSV file with a header row, and that the columns containing the airport names and dates are called "airport", "arrival\_date", and "departure\_date", respectively. You may need to modify the code to match the structure of your data.

1. Which airline should you fly on to avoid significant delays?

To find the airline that you should fly on to avoid significant delays using Python, PySpark, and Google Colab, you can use the following steps:

1. First, you will need to gather data on the flights and their arrival and departure times. You can use a dataset from a database, such as a SQL database, or you can scrape data from a website using a tool like Django.
2. Once you have the data, you can use PySpark to perform distributed data processing and analysis.
3. To calculate the delay for each flight, you can use PySpark's **withColumn** function to subtract the departure time from the arrival time and create a new column called "delay".
4. To find the mean delay for each airline, you can use PySpark's **groupBy** and **avg** functions to group the flights by airline and calculate the average delay for each group.
5. You can then use PySpark's **sort** function to sort the airlines by their mean delay.
6. Finally, you can use Python's **print** function to print the airline with the lowest mean delay.

Here is some example code that demonstrates these steps:

# Import PySpark and create a SparkSession

from pyspark.sql import SparkSession

spark = SparkSession.builder.appName("Airline Delay").getOrCreate()

# Load the data into a PySpark DataFrame

df = spark.read.csv("/path/to/data.csv", header=True)

# Calculate the delay for each flight

df = df.withColumn("delay", df["arrival\_time"] - df["departure\_time"])

# Group the data by airline and calculate the mean delay for each group

mean\_delays = df.groupBy("airline").avg("delay")

# Sort the airlines by mean delay

sorted\_delays = mean\_delays.sort("avg(delay)", ascending=True)

# Print the airline with the lowest mean delay

print("Airline with lowest mean delay:")

print(sorted\_delays.first())

Note that this code assumes that your data is in a CSV file with a header row, and that the columns containing the airline names and arrival and departure times are called "airline", "arrival\_time", and "departure\_time", respectively. You may need to modify the code to match the structure of your data.