Client Report - Are we missing JASON on our flight?

Course CSE 250 Joshua Sapp

Elevator pitch

We have been asked to run some calculations for a travel agency to help them pick out the best flights for their clients. Some of the factors considered were local weather, airline delay history, time of the year, and more. The dataset provided was inperfect, so there was some cleanup that had to be done first however.

GRAND QUESTION 1

Which airport has the worst delays? How did you choose to define "worst"? As part of your answer include a table that lists the total number of flights, total number of delayed flights, proportion of delayed flights, and average delay time in hours, for each airport.

Delay table

Airport Code	Total Flights	Delay Percentage	Average delay time
ATL	2.0	2.0%	59.82 Minuites
DEN	1.9	1.9%	53.73 Minuites
IAD	2.0	2.0%	61.04 Minuites
ORD	2.3000000000000003	2.3000000000000003%	67.83 Minuites
SAN	1.9	1.9%	47.26 Minuites
SFO	2.6	2.6%	62.38 Minuites
SLC	1.5	1.5%	49.34 Minuites

TECHNICAL DETAILS

GRAND QUESTION 2

What is the worst month to fly if you want to avoid delays? Include one chart to help support your answer, with the x-axis ordered by month. You also need to explain and justify how you chose to handle the missing Month data.

TECHNICAL DETAILS

GRAND QUESTION 3

According to the BTS website the Weather category only accounts for severe weather delays. Other "mild" weather delays are included as part of the NAS category and the Late-Arriving Aircraft category. Calculate the total number of flights delayed by weather (either severe or mild) using these two rules:

-A: 30% of all delayed flights in the Late-Arriving category are due to weather.

-B: From April to August, 40% of delayed flights in the NAS category are due to weather. The rest of the months, the proportion rises to 65%.

TECHNICAL DETAILS

GRAND QUESTION 4

Create a barplot showing the proportion of all flights that are delayed by weather at each airport. What do you learn from this graph (Careful to handle the missing Late Aircraft data correctly)?

TECHNICAL DETAILS

GRAND QUESTION 5

Fix all of the varied NA types in the data to be consistent and save the file back out in the same format that was provided (this file shouldn't have the missing values replaced with a value). Include one record example from your exported JSON file that has a missing value (No imputation in this file).

TECHNICAL DETAILS

APPENDIX A (PYTHON SCRIPT)

```
#%%
from altair.vegalite import data
from altair.vegalite.v4.schema.channels import Color
import numpy as np
import pandas as pd
import altair as alt
from altair_saver import save

class Are_we_missing_JSON:

    def main(self):
        self.settup()
        #self.q5()
        #self.q1()
        self.q2()
        #self.q3()
        #self.q4()

    def settup(self):
        #do any needed initializing
        #alt.data_transformers.enable('json')
        self.data = pd.read_json('flights_missing.json')
        self.airportCodeList = self.data['airport_code'].unique()
        self.months_list =
["January",'February','March','April','May','June','July','August','September','Oc
tober','November','December']
```

```
        self.year_list = [2000,2001,2002,2003,2004,2005,2006,2007,2008]
        #print(self.data)

    def q1(self):
        #Which airport has the worst delays? How did you choose to define “worst”? As part of your answer include a table that lists the total number of flights,
        #total number of delayed flights, proportion of delayed flights, and average delay time in hours, for each airport.

        # "worst Delays" is defined as having the highest average delay time per flight: (score = (delay time/delayed flights)/total flights)  with a low score being good.
        # The code below generates a table compatible with the .md file that contains the information
        airport_table = "| Airport Code | Total Flights | Delay Percentage | Average delay time |\n|-------------|--------------|-----------------|--------------------|\n"
        for airport in self.airportCodeList:
            airport_data = self.data.query(f"airport_code == '{airport}'")
            airport_table = airport_table+ f"| {airport} "
            flights = airport_data['num_of_flights_total'].sum()
            delays = airport_data['num_of_delays_total'].sum()
            delay_ratio = round((delays/flights),2)*10
            airport_table = airport_table+ f"| {delay_ratio} "
            airport_table = airport_table+ f"| {delay_ratio}% "
            total_delay_time = flights = airport_data['minutes_delayed_total'].sum()
            average_delay_time = round((total_delay_time/delays),2)
            airport_table = airport_table+ f"| {average_delay_time} Minuites |\n"
        print(airport_table)

    def q2(self):
        #What is the worst month to fly if you want to avoid delays? Include one chart to help support your answer, with the x-axis ordered by month.
        #You also need to explain and justify how you chose to handle the missing Month data.


        for airport in self.airportCodeList:
            airport_data = self.data.query(f"airport_code == '{airport}'")
            years = airport_data['year'].unique()
            recorded_years = []
            for item in years:
                try:
                    item = int(item)
                    recorded_years.append(item)
                except:
                    pass
            data_dictionary = {'month':self.months_list,'flight delays':[]}
            for year in recorded_years:
                airport_year_data = self.data.query(f"airport_code == '{airport}' & year == {year}")
                months = airport_year_data['month'].unique()
                recorded_months = []
```

```
                for item in months:
                    try:
                        item = str(item)
                        if item != 'n/a':
                            recorded_months.append(item)
                    except:
                        pass
                for month in recorded_months:
                    airport_year_month_data =  self.data.query(f"airport_code ==
'{airport}' & year == {year} & month == '{month}'")
                    index_choice = data_dictionary['month'].get(month)
                    print(index_choice)

    def q3(self):
        #According to the BTS website the Weather category only accounts for
severe weather delays. Other “mild” weather delays are included as part of the NAS
category and the Late-Arriving Aircraft category.
        #Calculate the total number of flights delayed by weather (either severe
or mild) using these two rules:
        # A: 30% of all delayed flights in the Late-Arriving category are due to
weather.
        # B: From April to August, 40% of delayed flights in the NAS category are
due to weather. The rest of the months, the proportion rises to 65%.
        clean_data = self.data.dropna()

        for month in self.months_list:
            month_data = clean_data.query(f"month == '{month}'")

            for year in self.year_list:
                if month in ['April','May','June','July','August']:
                    late_delays =
(month_data['num_of_delays_late_aircraft'].sum())*.3
                    nas_delays = (month_data['num_of_delays_nas'].sum())*.4
                    weather_delays = month_data['num_of_delays_weather'].sum()
                else:
                    late_delays =
(month_data['num_of_delays_late_aircraft'].sum())*3
                    nas_delays = (month_data['num_of_delays_nas'].sum())*.65
                    weather_delays = month_data['num_of_delays_weather'].sum()
                total_weather_delays = late_delays + nas_delays + weather_delays
                print(f"In {month} there were approximately {total_weather_delays}
weather relayted delays")

    def q4(self):
        #Create a barplot showing the proportion of all flights that are delayed
by weather at each airport. What do you learn from this graph (Careful to handle
the missing Late Aircraft data correctly)?
        pass

    def q5(self):
        #Fix all of the varied NA types in the data to be consistent and save the
file back out in the same format that was provided (this file shouldn’t have the
missing values replaced with a value).
        #Include one record example from your exported JSON file that has a
```

```
missing value (No imputation in this file).

        #for (column_name,column_data)in self.data.iteritems():
        #    for item in column_data:
        #        if item.
        pass

j = Are_we_missing_JSON()
j.main()



# %%
```