cse250_project2_Jacob_Padgett.md

Project 2 - Jacob Padgett

Elevator pitch

To work with JSON files that include missing data. The data is about airports and delays. The task of outputing charts and saving t data back to JSON was also a part of this project.

TECHNICAL DETAILS

This chart is part of **question 2**

This chart is part of **question 4**

Here is a video explaining my reasoning and logic behind how I chose to fill in the missing data for **question 2**.

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 t total number of flights, total number of delayed flights, proportion of delayed flights, and average delay time in hours, for eac airport.

- As a passanger on an airplane, I would define "worst delays" as length of the delay in minutes/hours. As an airline company, "v delays" would mean the overall dollar cost of the delay, for example, mechanical delays would be far more costly than weathe delays. Therefore, as a passanger, I would define it as "minutes_delayed_total/num_of_delays_total"

Total flights per airport airport_code ATL 4430047 ORD 3597588 DEN 2513974 SFO 1630945 SLC 1403384 SAN 917862 IAD 85 Name: num_of_flights_total, dtype: int64

Total delays per airport airport_code ATL 902443 ORD 830825 DEN 468519 SFO 425604 SLC 205160 SAN 175132 IAD 168467 N num_of_delays_total, dtype: int64

Delay Percentage airport_code SFO 0.260955 ORD 0.230939 ATL 0.203710 IAD 0.197831 SAN 0.190804 DEN 0.186366 SLC 0.14 dtype: float64

ORD average delay is 66 minutes SFO average delay is 61 minutes IAD average delay is 60 minutes ATL average delay is 59 minut DEN average delay is 53 minutes SLC average delay is 49 minutes SAN average delay is 47 minutes

ATL (Atlanta) has the most flights by 832,459 thousand, which is quite significant. They top out at 4,430,047 flights. ALT also has most number of delays at 902,443, however, it's percentage of delayed flights is 20.37% with an average delay time of 59 minutes delay.

But I dont believe it's the worst airport as far as delays are concerned. That would be ORD (Chicago) with 3,597,588 flights, and 830,825 delays for an average of 23.09% flights delayed at an average length of 66 minutes.

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 ord by month. You also need to explain and justify how you chose to handle the missing Month data.

- Here is a video explaining my reasoning and logic behind how I chose to fill in the missing data.
- As we can see in the chart above, June is the worst month for delays at the airports observed.

GRAND QUESTION 3

According to the BTS website the Weather category only accounts for severe weather delays. Other "mild" weather delays ar 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:

3a. 30% of all delayed flights in the Late-Arriving category are due to weather.

- All delayed flights in the Late-Arriving category, both num_of_delays_nas & num_of_delays_late_aircraft equals 622,016.

3b. From April to August, 40% of delayed flights in the NAS category are due to weather. The rest of the months, the proport rises to 65%.

- From April to August, 40% of flights are delayed, which equals 276,195.
- From September to March, 65% of flights are delayed, which equals 345,820.

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)?

- See the chart above

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 shouldn't have the missing values replaced with a value). Include one record example from your exported JSON file that has missing value (No imputation in this file).

```
import json

# Opening JSON file
f = open('final_product.json',)

# returns JSON object as
# a dictionary
data = json.load(f)

# Show that things have changed... this was `1500+` in the originl file
print(data['num_of_delays_carrier']['0'])

# Closing file
f.close()
```

APPENDIX A (Github)

- Project2.py
- Project2.ipynb The easier to read version.
- total_delayed_to_weather.png
- total_delays_per_month_chart.png
- The `.md` file cse250_project2_Jacob_Padgett.md
- The `.md` file as PDF

APPENDIX B (PYTHON SCRIPT)

```
# To add a new cell, type '# %%'
# To add a new markdown cell, type '# %% [markdown]'
# %% [markdown]
# # Jacob Padgett Project 2
# %% [markdown]
# ## Grand Questions
# ---
#
# ### 1. Which airport has the worst delays? How did you choose to define “worst”? As part of your answer include
#
# ### 2. What is the worst month to fly if you want to avoid delays? Include one chart to help support your answe
#
# ### 3. According to the BTS website the Weather category only accounts for severe weather delays. Other “mild”
#
# #### 3a. 30% of all delayed flights in the Late-Arriving category are due to weather.
#
# #### 3b. From April to August, 40% of delayed flights in the NAS category are due to weather. The rest of the r
#
# ### 4. Create a barplot showing the proportion of all flights that are delayed by weather at each airport. What
#
# ### 5. Fix all of the varied NA types in the data to be consistent and save the file back out in the same forma
#
# ---
#
# %% [markdown]
# # Grand Question 1:
# ## Which airport has the worst delays? How did you choose to define “worst”? As part of your answer include a t

# %%
# Imports
import pandas as pd
import altair as alt
import numpy as np


# Loading data
url = "https://github.com/byuidatascience/data4missing/raw/master/data-raw/flights_missing/flights_missing.json"
df_original = pd.read_json(url)
# df_original

# Check for nulls with `df.isna().sum()`

# From df.isna().sum(), I see there are missing values, so I will instead of
# deleting them, fill them with the mean of the column.
df_original = df_original.fillna(df_original.mean())
df_q1 = df_original.copy()
# print(df_q1.isna().sum())


# %%
# As a passanger on an airplane,  I would define "worst delays" as length of the
# delay in minutes/hours. As an airline company, "worst delays" would mean
# the overall dollar cost of the delay, for example, mechanical delays would be
# far more costly than weather delays. Therefore, as a passanger, I would define
# it as "minutes_delayed_total/num_of_delays_total"

# Total flights per airport
total_flights_per_airport = (
    df_q1.groupby("airport_code")
    .num_of_flights_total.sum()
    .sort_values(ascending=False)
)
print("Total flights per airport\n", total_flights_per_airport, "\n")

# Total delays per airport
total_delays_per_airport = (
```

```
    df_q1.groupby("airport_code").num_of_delays_total.sum().sort_values(ascending=False)
)
print("Total delays per airport\n", total_delays_per_airport, "\n")

# Add proportion of delayed flights column
df_q1["proportion_of_delayed_flights"] = df_q1.apply(
    lambda x: x["num_of_flights_total"] / x["num_of_delays_total"], axis=1
)

# Average delay length
df_q1["delay_length_average"] = df_q1.apply(
    lambda x: x["minutes_delayed_total"] / x["num_of_delays_total"], axis=1
)

# Subset
df_q1 = df_q1[
    [
        "airport_code",
        "num_of_flights_total",
        "num_of_delays_total",
        "proportion_of_delayed_flights",
        "minutes_delayed_total",
        "delay_length_average",
    ]
]

# Individualize the airports
df_q1_ORD = df_q1.query('airport_code == "ORD"')
df_q1_SFO = df_q1.query('airport_code == "SFO"')
df_q1_IAD = df_q1.query('airport_code == "IAD"')
df_q1_ATL = df_q1.query('airport_code == "ATL"')
df_q1_DEN = df_q1.query('airport_code == "DEN"')
df_q1_SLC = df_q1.query('airport_code == "SLC"')
df_q1_SAN = df_q1.query('airport_code == "SAN"')

# Get delay average in minutes
df_q1_ORD_delays = df_q1_ORD["delay_length_average"].sum() / len(df_q1_ORD)
df_q1_SFO_delays = df_q1_SFO["delay_length_average"].sum() / len(df_q1_SFO)
df_q1_IAD_delays = df_q1_IAD["delay_length_average"].sum() / len(df_q1_IAD)
df_q1_ATL_delays = df_q1_ATL["delay_length_average"].sum() / len(df_q1_ATL)
df_q1_DEN_delays = df_q1_DEN["delay_length_average"].sum() / len(df_q1_DEN)
df_q1_SLC_delays = df_q1_SLC["delay_length_average"].sum() / len(df_q1_SLC)
df_q1_SAN_delays = df_q1_SAN["delay_length_average"].sum() / len(df_q1_SAN)

# Delay Percentage/Proportion
delay_percentage = (total_delays_per_airport / total_flights_per_airport).sort_values(
    ascending=False
)
print("Delay Percentage\n", delay_percentage)
print()

# Print delay averages
print(f"ORD average delay is {df_q1_ORD_delays:.0f} minutes")
print(f"SFO average delay is {df_q1_SFO_delays:.0f} minutes")
print(f"IAD average delay is {df_q1_IAD_delays:.0f} minutes")
print(f"ATL average delay is {df_q1_ATL_delays:.0f} minutes")
print(f"DEN average delay is {df_q1_DEN_delays:.0f} minutes")
print(f"SLC average delay is {df_q1_SLC_delays:.0f} minutes")
print(f"SAN average delay is {df_q1_SAN_delays:.0f} minutes")
print()

# Turn delay_percentage to dict
delay_pct_dict = delay_percentage.to_dict()

# Worst delay airport
most_by = 4430047 - 3597588
atl_pct = 902443 / 4430047
```

```
print(
    f"""ATL (Atlanta) has the most flights by {most_by:,} thousand, which is
quite significant. They top out at {4430047:,} flights. ALT also has
the most number of delays at {902443:,}, however, it's percentage
of delayed flights is {delay_pct_dict['ATL']*100:.02f}% with an average delay time of
{df_q1_ATL_delays:.0f} minutes per delay.\n
But I dont believe it's the worst airport as far as delays are concerned. That
would be ORD (Chicago) with {3597588:,} flights, and {830825:,} delays for an
average of {delay_pct_dict['ORD']*100:.02f}% flights delayed at an average length of
{df_q1_ORD_delays:.0f} minutes."""
)

# %% [markdown]
# # Grand Question 2:
# ## What is the worst month to fly if you want to avoid delays? Include one chart to help support your answer, v
#
# %% [markdown]
# [Here is a video](https://youtu.be/Rc-O55r9LKA) explaining my reasoning and logic behind how I chose to fill in

# %%
# Loading data
df_q2 = df_original.copy()

# Convert month column to numeric
## After examining the data, I should make the `n/a` data to match the above month
ls = []

for i in df_q2["month"]:
    ls.append(i)

for i in range(len(ls)):
    if ls[i] == "n/a":
        ls[i] = ls[i - 1]

df_q2["month"] = ls

# Convert string month to numeric
def monthToNum(shortMonth):
    return {
        "January": 1,
        "Febuary": 2,
        "March": 3,
        "April": 4,
        "May": 5,
        "June": 6,
        "July": 7,
        "August": 8,
        "September": 9,
        "October": 10,
        "November": 11,
        "December": 12,
    }[shortMonth]


# Convert the data to numeric data
df_q2["month"] = df_q2["month"].apply(lambda x: monthToNum(x))

# Organize with groupby month
total_delays_per_month = (
    df_q2.groupby("month").num_of_delays_total.sum().sort_values(ascending=False)
)

# Convert to DF for charting
dfx = pd.DataFrame([total_delays_per_month]).T

# Fix indexing issue
dfx["month"] = dfx.index
dfx.reset_index(drop=True, inplace=True)
```

```
# Format the way I want
dfx = dfx[["month", "num_of_delays_total"]]

# Just in case it comes in handy later
# months = ['January','Febuary','March','April','May','June','July','August','September','October','November','D

# Chart
total_delays_per_month_chart = (
    alt.Chart(dfx)
    .encode(x="month", y="num_of_delays_total")
    .mark_bar()
    .properties(width=500, title="June is the worst month for delays")
)

# Results
print(
    "As we can see in the chart below, June is the worst month for delays at the airports observed.\n"
)

# SAVE CHART
total_delays_per_month_chart.save("total_delays_per_month_chart.png")
total_delays_per_month_chart

# %% [markdown]
# # Grand Question 3:
# ## According to the BTS website the Weather category only accounts for severe weather delays. Other “mild” wea
# ### 3a. 30% of all delayed flights in the Late-Arriving category are due to weather.
# ### 3b. From April to August, 40% of delayed flights in the NAS category are due to weather. The rest of the m
# %% [markdown]
# #### 30% of all delayed flights in the Late-Arriving category are due to weather.

# %%
# Understand the question
# “mild” weather delays are included as part of the NAS (minutes_delayed_nas, num_of_delays_nas)
# category and the Late-Arriving Aircraft (minutes_delayed_late_aircraft, num_of_delays_late_aircraft)


# %%
# Using "mild" weather, which is a combination of num_of_delays_nas & num_of_delays_late_aircraft, 30% are due t

# Copy data
df_q3 = df_original.copy()
df_q3 = df_q3[
    ["airport_code", "month", "num_of_delays_nas", "num_of_delays_late_aircraft"]
]

# This shows me there is a problem, negative amounts of flights in the num_of_delays_late_aircraft category, 40 
# Therefore, I'll have to preform surgical removal of these 40 observations, after all,
# they only make up 4.3% of the data, which is significant, but not critical to someones life, since this is onl
# an assignment, and I haven't seen anything that says to do anything specific.
print(df_q3[["num_of_delays_late_aircraft"]].value_counts().head(1))
print()

# Preform the operation to extract the undesired infection
df_q3 = df_q3[df_q3["num_of_delays_late_aircraft"] != -999]

# There are no longer 40 instances of -999 values in the dataframe
print(df_q3[["num_of_delays_late_aircraft"]].value_counts().head(1))

# Make a "combined" column
df_q3["combined"] = df_q3["num_of_delays_nas"] + df_q3["num_of_delays_late_aircraft"]

# Total flights delayed by weather is 30% of "combined"
df_q3["total_delayed_to_weather"] = df_q3["combined"] * 0.3

print()
print("-" * 75)
```

```
print()

################################################################################
# Now, sum it up to answer the first question, 3a
a3_answer = int(df_q3["total_delayed_to_weather"].sum().round())
print(
    f"""All delayed flights in the Late-Arriving category, both num_of_delays_nas
& num_of_delays_late_aircraft equals {a3_answer:,}."""
)
################################################################################
print()
print("-" * 75)

# %% [markdown]
# #### From April to August, 40% of delayed flights in the NAS category are due to weather. The rest of the mont

# %%
# Fill in the n/a months using the code above
ls = []
for i in df_q3["month"]:
    ls.append(i)
for i in range(len(ls)):
    if ls[i] == "n/a":
        ls[i] = ls[i - 1]
df_q3["month"] = ls

# Set it up
## Use "combined" column to get real numbers, not "total_delayed_to_weather" as
## it's already calculated.
b3 = df_q3.groupby(by="month").combined.sum().round()

# Make the DF to subset from
b3 = pd.DataFrame([b3]).T

# Fix indexing issue and make months numeric
b3["month"] = dfx.index + 1
b3.reset_index(drop=True, inplace=True)

month_nums_a = [4, 5, 6, 7, 8]  # 30%
month_nums_b = [1, 2, 3, 9, 10, 11, 12]  # 65%

# month_nums_a
b3a = b3.loc[b3["month"].isin(month_nums_a)]
b3a_answer = int(b3a["combined"].sum() * 0.3)
print(
    f"From April to August, 40% of flights are delayed, which equals {b3a_answer:,}.\n"
)

# month_nums_b
b3b = b3.loc[b3["month"].isin(month_nums_b)]
b3b_answer = int(b3b["combined"].sum() * 0.3)
print(
    f"From September to March, 65% of flights are delayed, which equals {b3b_answer:,}."
)

# %% [markdown]
# # Grand Question 4:
# ## Create a barplot showing the proportion of all flights that are delayed by weather at each airport. What do

# %%
# Group things together
delay_per_airport_to_weather = (
    df_q3.groupby("airport_code")
    .total_delayed_to_weather.sum()
    .sort_values(ascending=False)
)

# Convert to DF for charting
```

```
dfx = pd.DataFrame([delay_per_airport_to_weather]).T

# Fix indexing issue
dfx["airport_code"] = dfx.index
dfx.reset_index(drop=True, inplace=True)
dfx = dfx[["airport_code", "total_delayed_to_weather"]]


# %%
chart = (
    alt.Chart(dfx)
    .encode(x=alt.X("airport_code"), y=alt.Y("total_delayed_to_weather"))
    .mark_bar()
    .properties(width=400, title="Total Weather Delays")
)

# Save chart
chart.save("total_delayed_to_weather.png")
chart

# %% [markdown]
# # 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 tl

# %%
df_original = pd.read_json(url)

df_q5 = df_original.copy()


# %%
df_q5


# %%
# Check airport_code for shenanigans
print(df_q5["airport_code"].unique())  # Nothing suspicious going on (as printed)


# %%
# Check airport_code for shenanigans
for i in df_q5["airport_name"].unique():
    print(i)  # Something is blank... let's fix that
df_q5["airport_name"] = df_q5["airport_name"].replace("", np.nan)

print()
print("-" * 75)
print()

# See update of what's going on
print(df_q5.isna().sum())


# %%
# Normalize everything the hard way
df_q5["year"] = df_q5["year"].replace("nan", np.nan)
df_q5["year"] = df_q5["year"].replace("n/a", np.nan)
df_q5["year"] = df_q5["year"].replace("", np.nan)

df_q5["month"] = df_q5["month"].replace("nan", np.nan)
df_q5["month"] = df_q5["month"].replace("n/a", np.nan)
df_q5["month"] = df_q5["month"].replace("", np.nan)

df_q5["num_of_flights_total"] = df_q5["num_of_flights_total"].replace("nan", np.nan)
df_q5["num_of_flights_total"] = df_q5["num_of_flights_total"].replace("n/a", np.nan)
df_q5["num_of_flights_total"] = df_q5["num_of_flights_total"].replace("", np.nan)

df_q5["num_of_delays_carrier"] = df_q5["num_of_delays_carrier"].replace("nan", np.nan)
```

```
df_q5["num_of_delays_carrier"] = df_q5["num_of_delays_carrier"].replace("n/a", np.nan)
df_q5["num_of_delays_carrier"] = df_q5["num_of_delays_carrier"].replace("", np.nan)
df_q5["num_of_delays_carrier"] = df_q5["num_of_delays_carrier"].replace("1500+", np.nan)

df_q5["num_of_delays_late_aircraft"] = df_q5["num_of_delays_late_aircraft"].replace(
    "nan", np.nan
)
df_q5["num_of_delays_late_aircraft"] = df_q5["num_of_delays_late_aircraft"].replace(
    "n/a", np.nan
)
df_q5["num_of_delays_late_aircraft"] = df_q5["num_of_delays_late_aircraft"].replace(
    -999, np.nan
)
df_q5["num_of_delays_late_aircraft"] = df_q5["num_of_delays_late_aircraft"].replace(
    "", np.nan
)

df_q5["num_of_delays_nas"] = df_q5["num_of_delays_nas"].replace("nan", np.nan)
df_q5["num_of_delays_nas"] = df_q5["num_of_delays_nas"].replace("n/a", np.nan)
df_q5["num_of_delays_nas"] = df_q5["num_of_delays_nas"].replace("", np.nan)

df_q5["num_of_delays_security"] = df_q5["num_of_delays_security"].replace("nan", np.nan)
df_q5["num_of_delays_security"] = df_q5["num_of_delays_security"].replace("n/a", np.nan)
df_q5["num_of_delays_security"] = df_q5["num_of_delays_security"].replace("", np.nan)

df_q5["num_of_delays_weather"] = df_q5["num_of_delays_weather"].replace("nan", np.nan)
df_q5["num_of_delays_weather"] = df_q5["num_of_delays_weather"].replace("n/a", np.nan)
df_q5["num_of_delays_weather"] = df_q5["num_of_delays_weather"].replace("", np.nan)

df_q5["num_of_delays_total"] = df_q5["num_of_delays_total"].replace("nan", np.nan)
df_q5["num_of_delays_total"] = df_q5["num_of_delays_total"].replace("n/a", np.nan)
df_q5["num_of_delays_total"] = df_q5["num_of_delays_total"].replace("", np.nan)

df_q5["minutes_delayed_carrier"] = df_q5["minutes_delayed_carrier"].replace(
    "nan", np.nan
)
df_q5["minutes_delayed_carrier"] = df_q5["minutes_delayed_carrier"].replace(
    "n/a", np.nan
)
df_q5["minutes_delayed_carrier"] = df_q5["minutes_delayed_carrier"].replace("", np.nan)

df_q5["minutes_delayed_late_aircraft"] = df_q5["minutes_delayed_late_aircraft"].replace(
    "nan", np.nan
)
df_q5["minutes_delayed_late_aircraft"] = df_q5["minutes_delayed_late_aircraft"].replace(
    "n/a", np.nan
)
df_q5["minutes_delayed_late_aircraft"] = df_q5["minutes_delayed_late_aircraft"].replace(
    "", np.nan
)

df_q5["minutes_delayed_nas"] = df_q5["minutes_delayed_nas"].replace("nan", np.nan)
df_q5["minutes_delayed_nas"] = df_q5["minutes_delayed_nas"].replace("n/a", np.nan)
df_q5["minutes_delayed_nas"] = df_q5["minutes_delayed_nas"].replace("", np.nan)

df_q5["minutes_delayed_security"] = df_q5["minutes_delayed_security"].replace(
    "nan", np.nan
)
df_q5["minutes_delayed_security"] = df_q5["minutes_delayed_security"].replace(
    "n/a", np.nan
)
df_q5["minutes_delayed_security"] = df_q5["minutes_delayed_security"].replace(
    "", np.nan
)

df_q5["minutes_delayed_weather"] = df_q5["minutes_delayed_weather"].replace(
    "nan", np.nan
)
```

```
df_q5["minutes_delayed_weather"] = df_q5["minutes_delayed_weather"].replace(
    "n/a", np.nan
)
df_q5["minutes_delayed_weather"] = df_q5["minutes_delayed_weather"].replace("", np.nan)

df_q5["minutes_delayed_total"] = df_q5["minutes_delayed_total"].replace("nan", np.nan)
df_q5["minutes_delayed_total"] = df_q5["minutes_delayed_total"].replace("n/a", np.nan)
df_q5["minutes_delayed_total"] = df_q5["minutes_delayed_total"].replace("", np.nan)


# %%
# df_original = df_original.fillna(df_original.mean())
df_q5.isna().sum()


# %%
df_q5_view_json = df_q5.to_json()
print(df_q5_view_json)
df_q5.to_json(r"final_product.json")


# %%
import json

# Opening JSON file
f = open(
    "final_product.json",
)

# returns JSON object as
# a dictionary
data = json.load(f)

# Show that things have changed... this was `1500+` in the originl file
print(data["num_of_delays_carrier"]["0"])

# Closing file
f.close()
# %%%
```