Turkey, Duck, and Fish

Chat GPT Usage Report

Saving Data

In part 4 of the notebook “api requests”, it is inspired by ChatGPT to import the “os” module to better save and read the .csv files to specific paths.

```
import os

folder_name = 'xxx'
folder_path = os.path.join(os.pardir, folder_name)
file_path = os.path.join(folder_path, 'xxx.csv')
```

Data filtration

In part 2 of the notebook “Data Filteration.ipynb”, we are inspired by ChatGPT in solving the problem of “unneeded information” (e.g. runway, terminal) if there is no delay happening.

The code scrapes everything starting from the 4th comma to the 5th comma inside the column “departure”, while if delay information does not exist at all (no delay happened), this part of the information will become runway information or terminal information.

It is inspired by ChatGPT to write the following code:

```
df['delay in mins'].apply(lambda info: 'None' if info is None or ('Time' in info or 'Runway' in info) else info)
```

Seaborn (sns functions)

Example:
sns.scatterplot(x=x_column, y=y_column, hue='Airline Company', data=df, palette='viridis')

We used chatGPT to navigate the palate function in the x and y axis graph. We realized that we can use some predetermined color scheme to show distinction between the values. For example, there is a cold color scheme for a range of colors. We didn’t have to link the percentage of red with the value in order to mix colors.

Seaborn is a very useful add-on because we can change the color scheme easily within one line of code. It was useful when we wanted to change one airline’s color to be different from other airlines.

Turning off legend for the scatter plot was difficult, as it is a default setting of the plot. We used chatGPT to find the set visible function.

Turkey, Duck, and Fish

Hover over effect

Example:

```
from sklearn.linear_model import LinearRegression
import ipywidgets as widgets
from IPython.display import display
import numpy as np
```

We learned about these add-ons, which enabled us to program a box pop-up when we have our mouse hover over the dots. The Ipython.display enabled a lot more functions like reactions to the mouse hovering over and triggering a pop-up box.

Geopanda functions

We used chatGPT to help generate a world map. Geopanda documentation is very complicated so chatGPT helped me to narrow down the functions that we want to use. We also didn't really know how to generate dots (turns out it's by overlapping another layer on the world map) on geopanda or how to make them colorful.

Scipy.stats

We used scipy.stats for our flight delay prediction model. It calculated the standard deviation of the table and mean under a normal distribution. It also outputs a normal distribution graph, which we wouldn't know how to create without this specific add-on. The function also calculates probabilities automatically, which we used for the final output.

Inserting plot on the website

We used ChatGPT to help us display the scatter plots generated by plotly on our website with the 'write_html()' function, and 'config' to set the width and height of the plot. We initially tried the method of inserting images given by the markdown syntax guide online, but the plot either could not show on the website or was shown with a distorted y-axis. So we asked ChatGPT to generate the method of inserting this scatter plot without losing the zooming in and out function and with adjustable height. This is the code with its help**:**

```
fig_delay.write_html("scatter_plot_1.html", config={'displayModeBar': True}, full_html=False)
```

In markdown:

```
![Scatter Diagram](scatter_plot_1.html){ width=800px height=600px }
```