Report of mini project 1

Nazmus Sakib

Introduction

In this project, I was engaged in a series of interconnected tasks that combined collecting flight data by scraping, cleaning the scraped data, performing exploratory data analysis on it, and finally let the end-user filter and sort the dataset. My overarching objective is to leverage web scraping techniques to acquire flight data, process and clean this data to ensure accuracy and consistency, perform exploratory data analysis to extract meaningful insights, and ultimately deliver a solution that offers travelers the cheapest and fastest flight options based on their unique requirements. I have scrapped the flights based on a chosen date and city.

Data collection

I scrapped two sites called `momondo.com` and `cheapoair.com` to get the available flights from Helsinki to Berlin for the date 30-10-2023. To perform web scraping I used a library called playwright [playwright.dev/python/]. I have faced enormous amounts of challenges to perform the initial setup. After trying out a few libraries, I found the Playwright library and it worked. Playwright is very easy to set up and it was successful to bypass the bot detection as well.

Challenges of scraping: Challenges I faced during scraping were as follows:

1. I couldn't set up Selenium on google collab. So I explored other available options and settled with Playwright.
2. Flight booking sites have bot detection mechanisms which prevents us from scraping their sites. I tried changing the user agent of the browser instance to Firefox instead of Chrome and it worked for a few sites. By setting the user agent to Firefox I was able to bypass bot detection in momondo and cheapoair.
3. Next challenge was to properly locate the desired data fields in the DOM. I used CSS selectors to locate them in most cases. Sometimes data doesn't become available right after the initial render of the webpage. In those cases, I had to perform user interaction events such as mouse click and scroll etc. to bring the data to DOM.

Data Fields: I collected the following data from the above mentioned sites:

1. Airline names
2. Departure time
3. Arrival time
4. Price in USD
5. Total duration of the journey
6. Number of stops
7. Layover time between flights
8. Flights and Airports of each stops (max 2 stops)

Data collection Method: I collected the data through web scraping using Playwright library.

Data collection Storage: I stored the data in a csv file. I have also mounted google drive with the project and saved the csv in google drive. In the next steps, I retrieved the csv from google drive.

Data analysis:

After collecting the desired fields of flights, I started cleaning the data so that I can perform analysis on it.

Data cleaning: There were following problems in the initial dataset:

1. Airline names were inconsistent between booking sites. I have replaced the names of the same airlines with a single title. For example, Air Baltic were found as airBaltic in some sites. I replaced all of them with 'Air Baltic' to make it consistent.
2. I converted all the duration to minutes. In the original data, they were in hours and minutes.
3. Price had a dollar symbol with it which was cleared and converted to float.
4. 'Number of stops' field was converted to a single integer value(eg, 2).
5. Stops airport and layover were in a single cell. I split and formatted layover and airport field.
6. I converted the departure and arrival times from string to numpy datetime64 object so that it can be analyzed and compared.
7. Finally I merged the clean dataset in a single dataframe

Data visualization:

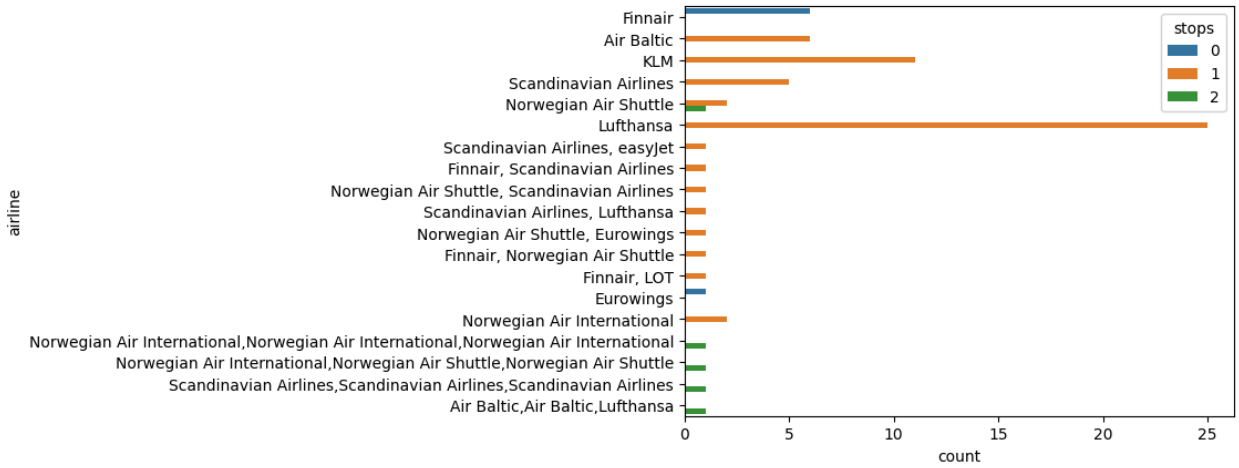

Fig 1: Count plot of airlines

In Figure 1, the data reveals that 'Lufthansa' offers the most extensive selection of flights, with the highest number of options available. Additionally, both 'Air Baltic' and 'KLM' also provide a substantial number of flight choices. Following closely are 'Scandinavian Airlines' and 'Finnair', which offer a notable number of flight options as well. Within the hue colors displayed, it is apparent that the majority of flights are categorized as 1-stop flights. The remaining flight combinations, in contrast, exhibit considerably lower counts and do not display any discernible patterns.

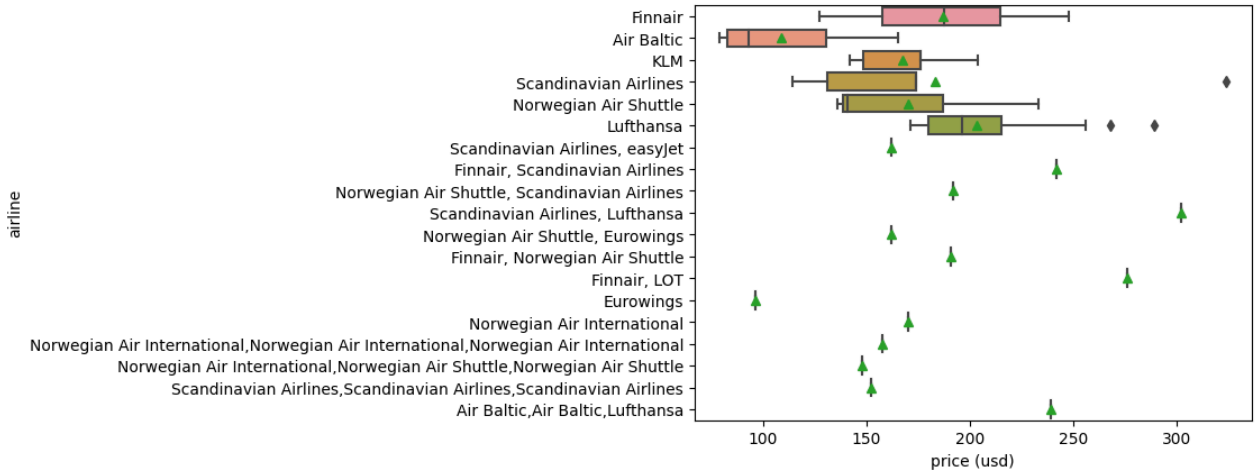

Fig 2: Boxplot of airline and price

Figure 2 paints a clear picture of flight pricing based on airlines, with AirBaltic emerging as the most budget-friendly option. Following AirBaltic, Scandinavian and Norwegian airlines also offer competitively priced flights. In terms of mean prices, AirBaltic maintains its position as the most cost-effective choice, followed by KLM, Norwegian, Scandinavian, Finnair, and Lufthansa. Notably, mixed flights involving multiple airlines were present in the data, but their limited representation prevented us from extracting meaningful insights from this category.

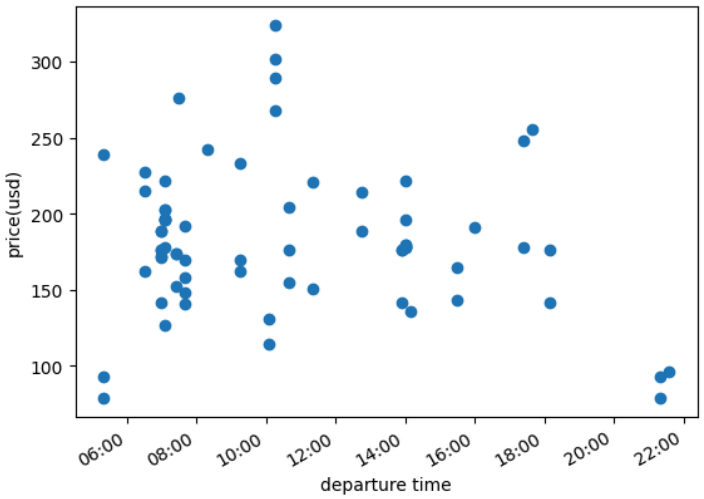

Fig 3: Scatter plot between price and departure

Figure 3 depicts a clear trend in ticket pricing, revealing that the lowest starting prices are observed at 6:00 and 22:00, while the starting prices are notably higher during the hours of 12:00 to 18:00. Interestingly, the peak in ticket prices is evident at approximately 10:00.

For instance, in our Helsinki to Berlin flights, tickets can be purchased for as low as $100 during the early morning hours at around 6:00. However, during the midday to late afternoon hours from 12:00 to 18:00, the starting prices begin at approximately $130.

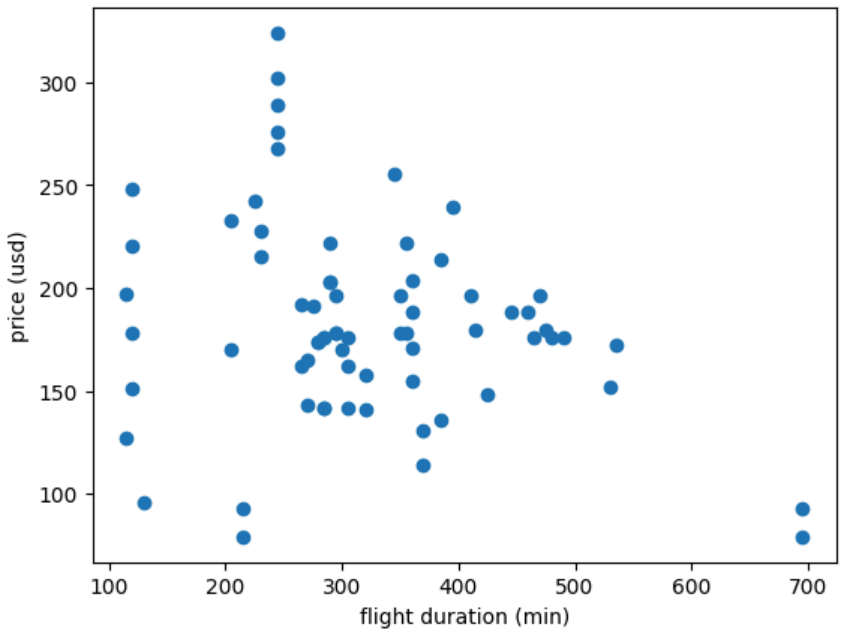

Fig 4: Scatter plot between price and duration

From figure 4, we can see that there is no clear correlation between duration and price. Every duration comes with a varying range of price. However, tickets priced under $100 are typically available for flights with a duration of around 220 minutes or less, and it's noteworthy that extended flight durations do not consistently result in lower ticket prices. Intriguingly, there are two exceptions where flights have durations of 700 minutes yet feature lower prices.

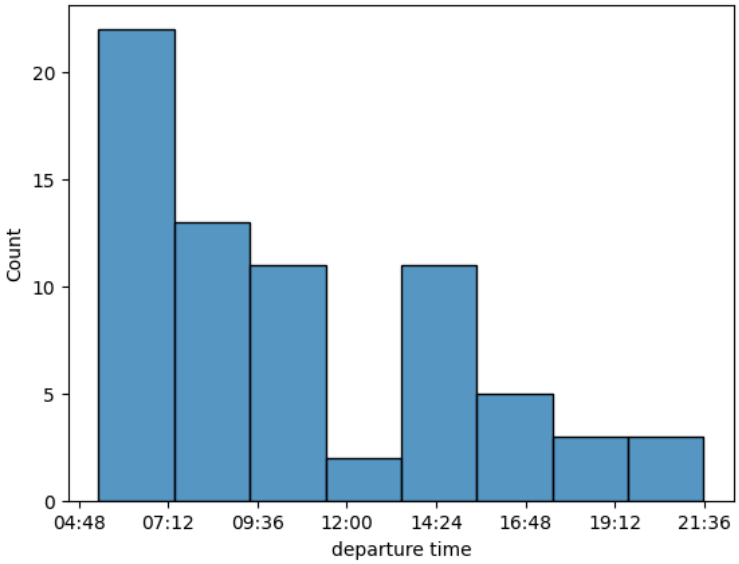

Fig 5: Histogram of departure time

The histogram illustrates that the highest frequency of flights occurs during the early morning hours, specifically between 4:00 and 7:00, while the fewest flights are available during the nighttime period, spanning from 15:00 to 21:00.

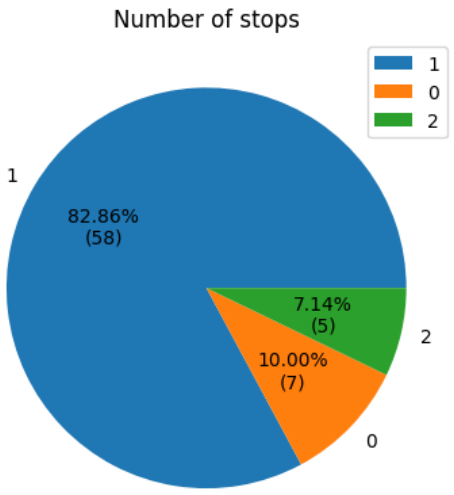

Fig 6: Pie diagram of stop count

The pie plot provides a comprehensive overview of flight distribution categorized by the number of stops. It is evident that one-stop flights dominate the landscape, accounting for a substantial majority at 82% of all available flights. Direct flights, while the second most prevalent, still command a significant share, comprising 10% of the total number of available flights. In contrast, flights with two stops are relatively scarce, representing only 7% of the overall flight options.

User interaction: I have filtered and sorted the flights data and also printed the fastest and cheapest flight based on given criteria.

Conclusion

Our quest to return the cheapest and fastest flight based on traveler-specific criteria has finally turned into a solution. We've gained insights about the impact of airlines, departure time and duration on price and flight availability.

However, along the way, I encountered a significant scientific bottleneck: the dynamic nature of airline websites and the anti-scraping measures they employ. These measures posed a challenge to the automated data collection process and required innovative solutions. To overcome this bottleneck, I employed the user-agent rotation technique in my web scraping code. This strategy allowed me to navigate the evolving website structures and bot detection issue.

As I got into the data analysis phase, another scientific bottleneck emerged. The amount of flight data with some inconsistency, presented challenges in terms of processing and drawing meaningful insights. Additionally, creating informative and visually appealing plots to convey these insights proved to be a formidable task. The need to balance data depth with interpretability posed a scientific challenge in itself. To address these bottlenecks, I implemented data processing steps that efficiently cleaned and transformed the data for analysis. Furthermore, I used powerful data visualization libraries like seaborn and matplotlib to generate insightful plots and charts, ensuring that the data-driven narrative was both informative and accessible.