Homework 2

(Due: October 10, 2025)

The purposes of this homework are to give you experience in working with Pandas and GeoPandas, and Pythons support for filtering and transforming tabular data. Submit to gradescope a pdf writeup containing the problem description for each problem, your Python code, and approprite textual and graphical output.

Question 1: 10 points. A laboratory experiment is conducted on 1030 specimens to determine the compressive strength (MPa) of concrete as a function of age (days) and various ingredients (e.g., cement, blast furnace slag, fly ash, water, superplasticizer, coarse aggregate, and fine aggregate).

The experimental data (see python-code.d/data/materials/concrete-strength-data.csv) comprises eight (quantitative) input parameters:

```
Parameter             Description
-----------------------------------------------
Cement                kg in a m3 mixture.
Blast Furnace Slag    kg in a m3 mixture.
Fly Ash               kg in a m3 mixture.
Water                 kg in a m3 mixture.
Superplasticizer      kg in a m3 mixture.
Coarse Aggregate      kg in a m3 mixture.
Fine Aggregate        kg in a m3 mixture.
Age                   day (1 -- 365).
-----------------------------------------------
```

and one output:

```
Parameter             Description
-------------------------------------------------------
Concrete strength     Concrete compressive strength (MPa).
-------------------------------------------------------
```

What to do? Write a Python program that will:

1. Read the experimental test results from a file `concrete-strength-data.csv` into a Pandas dataframe.
2. Extract numpy arrays from the dataframe for age (days) and concrete compressive strength (MPa).

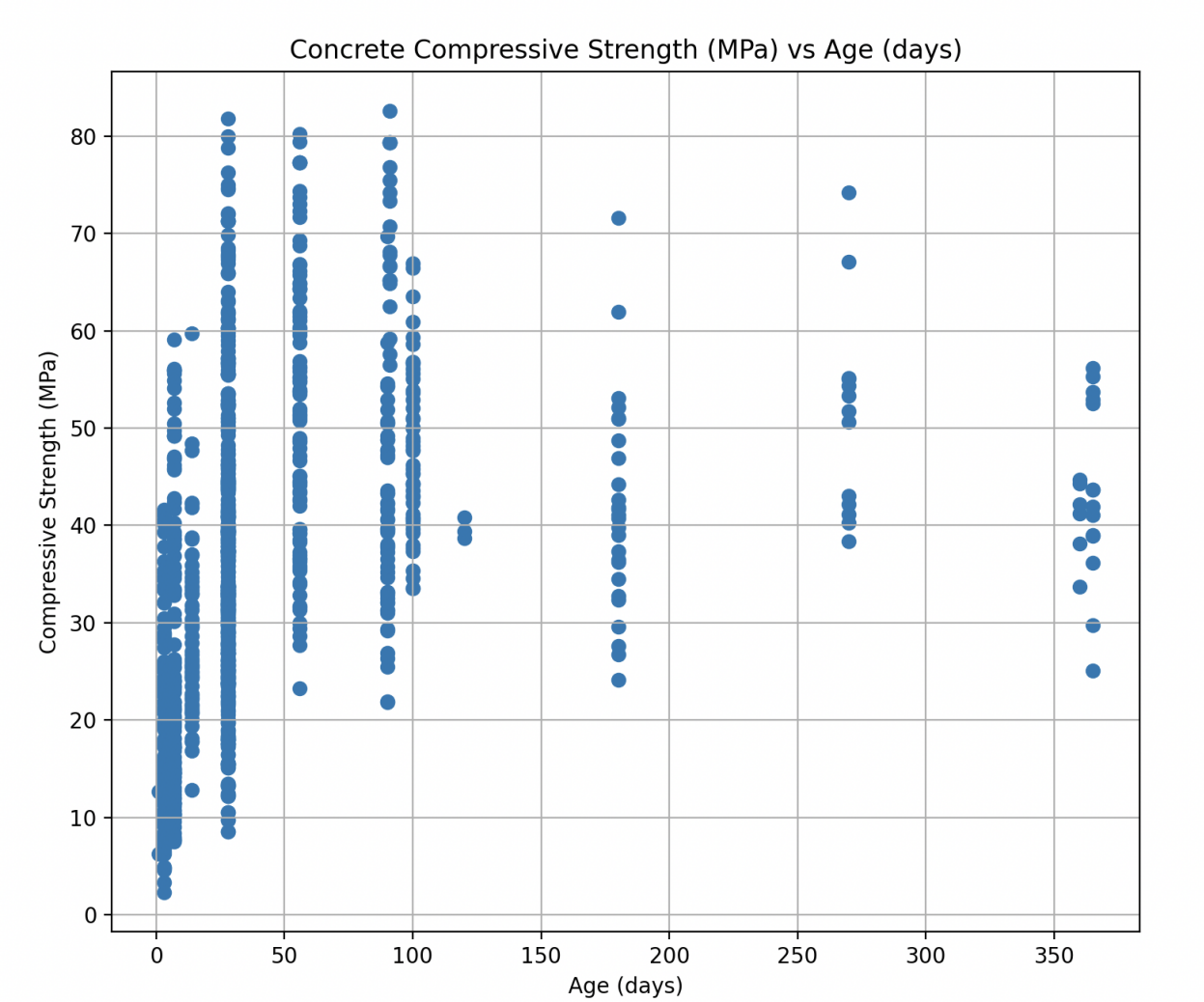

Figure 1: Scatter chart of concrete compressive strength (MPa) vs age (days).

3. Compute and print the range of parameter values for each input (e.g., Cement content, Blast Furnace Slag, Fly Ash, Water, Superplasticizer, Coarse Aggregate, Fine Aggregate and Age.

4. Compute and print the maximum, minimum, and average concrete compressive strengths.

5. Create a scatter chart of concrete strength (MPa) vs age (days). See Figure 1.

6. Use the Pandas cut function (i.e., google `pd.cut()`) to organize the strength data into intervals: 0–25, 25–50, 50–75, and 75–100.

7. Generate a histogram of concrete compressive strength (MPa).

8. Generate the cumulative probability distribution for concrete compressive strength, and then create a stair-step graph of "cumulative frequency" versus "compressive strength."

Note 1. The average value of the experimental results can be computed using Python's buildin functions. The "cumulative frequency" versus "compressive strength" is given by

$$\text{Cumulative frequency(y)} = \int_0^y p(x)dx \tag{1}$$

where $p(x)$ is the probability distribution of concrete compressive strengths. The matplotlib functions `plt.hist(.)` and `plt.step(.)` create histogram and stair-step graphs.

Note 2. See python-code.d/pandas/TestMaterialsA36Steel.py for a very similar problem setup and solution.

Question 2: 20 points.

Problem Statement. The purpose of this question is to take a first step in understanding the spatial and temporal nature of air traffic from BWI to international destinations, 1990 through 2020. Solutions to this problem are complicated by the constantly evolving nature of resources at BWI – for our purposes, it is important to note that Southwest Airlines did not have a presence at BWI until the mid 1990s, and Concourse A/B only opened in 2005 (source: Google).

International Air Traffic at BWI. Compared to volumes of domestic traffic to/from Baltimore-Washington International (BWI) Airport (175,000 flights in 2015 alone; 70% of domestic flights are Southwest), the number of international flights is small. Figure 2 shows, for example, a draft visualization of only 6,935 international flights to/from BWI, 1990 through 2020.

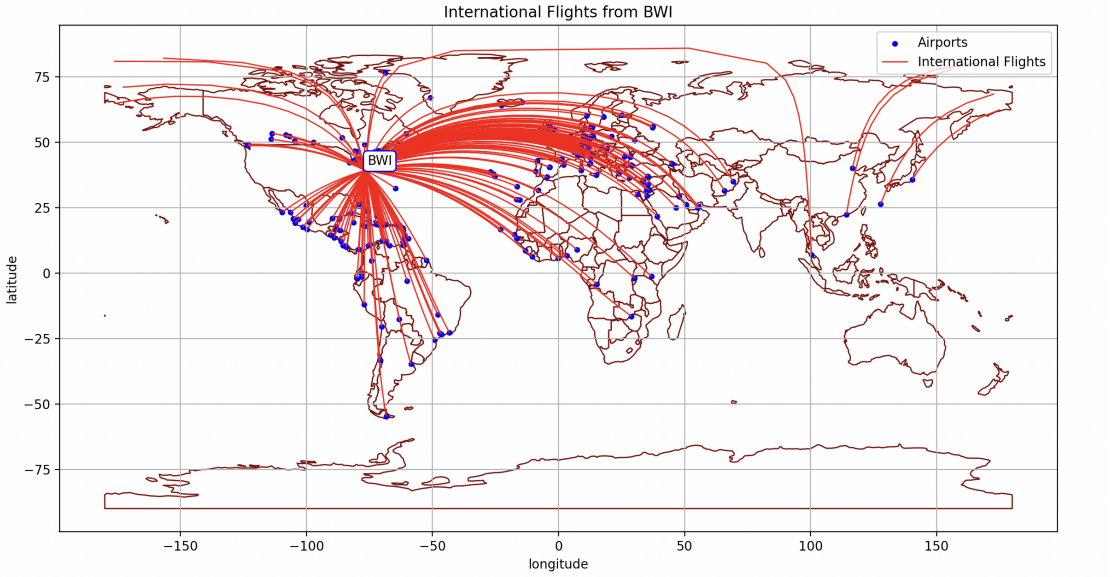

Figure 2: Draft visualization for international flights from BWI, 1990 through 2020.

A preliminary analysis of the flight data (see python-code.d/data/transportation/air/flights-international-bwi1990-2020.csv) indicates that planes departed BWI to 246 destinations, the most popular routes being:

- BWI to CUN (Cancún International Airport, Mexico) $\longrightarrow$ 803 flights.
- BWI to YYZ (Lester Pearson International Airport, Toronto) $\longrightarrow$ 527 flights.
- BWI to NAS (Lynden Pindling International Airport, Nassau, Bahamas) $\longrightarrow$ 509 flights.
- BWI to MBJ (Sangster International Airport, Jamaica) $\longrightarrow$ 484 flights.
- BWI to PUJ (Punta Cana International Airport, Dominican Republic) $\longrightarrow$ 388 flights.

- BWI to KEF (Keflavik International Airport, Iceland) $\longrightarrow$ 355 flights.
- BWI to LHR (London Heathrow Airport, England) $\longrightarrow$ 266 flights.
- BWI to AUA (Queen Beatrix International Airport, Aruba) $\longrightarrow$ 265 flights.

At the other end of the spectrum, approximately half (135) of the flight destinations report a flight count of three or less. For example, during the past thirty years (1990-2020) there is only one recorded flight from BWI to YYJ (Victoria International Airport, Victoria, Canada).

Four Sources of Data: To understand the nature of international flights from BWI, we will gather data from four sources:

Data Source 1: The data file data/transportation/air/airline-iata-codes-small.csv contains iata codes and countries of origin for the main airlines using BWI, IAD and DCA. The columns of data are:

```
   Code -- IATA code for airline.
Airline -- Name of ailine.
Country -- Ownership of the airline ...
```

For example, the first entry is: AA, American Airlines, United States.

Data Source 2: The data file data/airports/airports-worldwide-large.csv is a listing of 57,422 airports worldwide. For our purposes, the columns of interest are:

```
Col  1   ident         --
Col  2   type          -- Type of airport (e.g., large_airport, heliport, closed).
Col  3   name          -- Name of airport.
Col  4   elevation_ft  -- Airport elevation (ft).
Col  5   continent     --
Col  6   iso_country   -- Code for country of airport (e.g., CA for Canada).
Col  7   iso_region    -- Maryland, British Columbia, etc ...
Col  8   municipality  --
Col  9   gps_code      --
Col 10   iata_code     -- International Air Transportation Code (e.g., BWI, YYZ).
Col 11   local_code    --
Col 12   coordinates   -- Airport longitude and latitude.
```

Data Source 3: The data file data/transportation/air/flights-international-bwi1990-2020.csv contains details of 6,935 flight departures from BWI to international destinations. The data is organized into 16 columns, namely:

```
Col  1   data_dte    --
Col  2   Year        --
Col  3   Month       --
Col  4   usg_apt_id  --
Col  5   usg_apt     --
Col  6   usg_wac     --
Col  7   fg_apt_id   --
Col  8   fg_apt      --
Col  9   fg_wac      --
Col 10   airlineid   --
Col 11   carrier     --
Col 12   carriergroup --
Col 13   type        --
Col 14   Scheduled   --
Col 15   Charter     --
Col 16   Total       --
```

Data Source 4: The shape data file data/geography/world/ne_110m_admin_0_countries.shp provides details on a world map.

What to do?

Write a Python program that will:

1. Load the contents of airline-iata-codes-small.csv into a Pandas dataframe. Create and print a dictionary of airline code, i.e.,

   ```
   Dictionary: Airlines:
   -------------------------------------------------
   key: AA --> value: [' American Airlines', ' United States'] ...
   key: AC --> value: [' Air Canada', ' Canada'] ...
   key: AF --> value: [' Air France', ' France'] ...

   ... etc, etc, etc ...
   -------------------------------------------------
   ```

   For a hint, see the pandas method `to_dict(..)`.

   Also see python-code.d/applications/airports/TestAirportMaryland01.py for a similar problem setup and solution.

2. Load the contents of airports-worldwide-large.csv into a Pandas dataframe. Remove all entries that do not have a valid `iata` code. Create and print a dictionary of airports that can be accessed by their iata code. A sample of text output might look like:

   ```
   key: CUN --> value: ['Cancún International Airport', nan, 'MX', '-86.87709, 21.03650'] ...
   ```

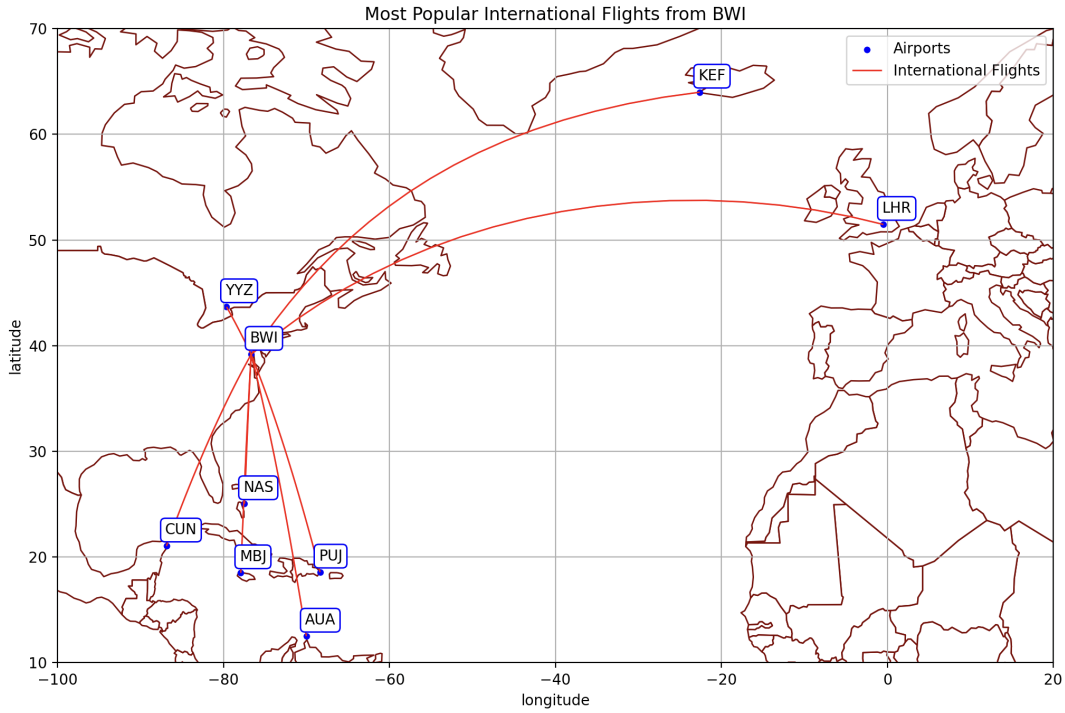

Figure 3: Most popular flights from BWI to international destinations, 1990 through 2020.

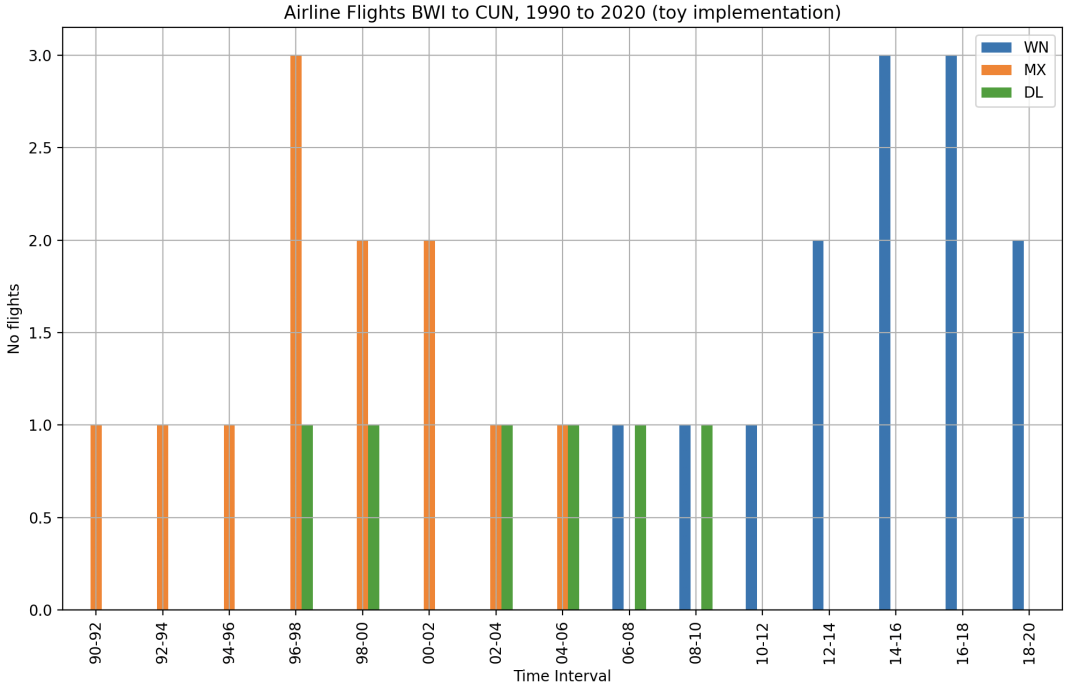

Figure 4: Bar chart of airlines and flights, BWI to CUN (with toy data).

3. Load the contents of flights-international-bwi1990-2020.csv into a Pandas dataframe. Create and print a dictionary of airport destinations (i.e., the key) along with a count of the number of flights (i.e., the value). Sort this dictionary by flight count, then print details of the flight destination such as the city, country, and coordinates. For example, textual output for the BWI (Baltimore) to CUN (Cancun) flight might look like:

```
--- Flight: BWI to CUN     --> count: 803 ...
--- City: Cancún International Airport ...
--- Region: nan ...
--- Country: MX ...
--- Coordinates: (long, lat) = (-86.87709,  21.03650) ...
```

4. Replicate Figure 2, and then customize the visualization to highlight the most frequently traveled international routes from BWI (listed at the top of the question). Something along the lines of Figure 3 would be good.

5. Create a bar chart or histogram of flights from BWI to CUN (Cancun, MX), 1990 to 2020.

My original thought, which by the way turned out to be completely wrong, was that flights from BWI to CUN would be dominated by Southwest Airlines. However, when you examine the data, a completely different story emerges.

Identify the most important airlines that have flown to Cancun, and then create a single diagram that shows the importance of their services as a function of time. For hints on extensions to Figure 4, see python-code.d/charts/TestHistogram03.py.

Question 3: 20 points.

The purpose of this question is to give you **practice at using Pandas** to work with – i.e., filter, transform, group, query, – a datafile that reports on 5.8 million flights across in US during 2015. In case you are wondering, this file is too large to fit into Microsoft Excel.

There are many questions one might ask: How many airlines and planes operate within the domestic US? How many planes does each airline have? Which airlines have the most flights per year? What are the busiest airports? With respect to individual flights, what are the shortest and longest flights? How many flights (on average) does a plane make during a year? What proportion of the day is a plane actually flying? How far does a plane travel during a year?

Data Source: The data file python-code.d/data/transportation/air/flights-usa-large2015.csv contains data on 5.8 million flights across in US during 2015. The data is organized into 31 columns, namely:

```
Cols 1-3   YEAR, MONTH, DAY    -- Date of the flight.
Col    4   DAY_OF_WEEK         -- Monday (1), Tuesday (2), etc ...
Col    5   AIRLINE             -- For example, UA = United Airlines ...
Col    6   FLIGHT_NUMBER       --
Col    7   TAIL_NUMBER         --
Col    8   ORIGIN_AIRPORT      --
Col    9   DESTINATION_AIRPORT --
Col   10   SCHEDULED_DEPARTURE --
Col   11   DEPARTURE_TIME      --
Col   12   DEPARTURE_DELAY     --
Col   13   TAXI_OUT            --
Col   14   WHEELS_OFF          --
Col   15   SCHEDULED_TIME      --
Col   16   ELAPSED_TIM         --
Col   17   AIR_TIME            --
Col   18   DISTANCE            --
Col   19   WHEELS_ON           --
Col   20   TAXI_IN             --
Col   21   SCHEDULED_ARRIVAL   --
Col   22   ARRIVAL_TIME        --
Col   23   ARRIVAL_DELAY       --
Col   24   DIVERTED            --
Col   25   CANCELLED           --
Col   26   CANCELLATION_REASON --
Col   27   AIR_SYSTEM_DELAY    --
Col   28   SECURITY_DELAY      --
Col   29   AIRLINE_DELAY       --
Col   30   LATE_AIRCRAFT_DELAY --
Col   31   WEATHER_DELAY       --
```

Notice that this file makes reference to the origin and destination airports and the airline providing the flight, so analysis of its contents will require the support of airline-iata-codes-small.csv and airports-worldwide-large.csv (covered in Question 2).

What to do? Write a Python program that will:

1. Load the contents of airports-worldwide-large.csv into a Pandas dataframe. Remove all entries that do not have a valid `iata` code. Create and print a dictionary of airports. For a hint, see the pandas method `to_dict(..)`.

2. Load the contents of python-code.d/data/transportation/air/flights-usa-large2015.csv into a Pandas dataframe.

Preliminary analysis indicates that while the overall quality of this datafile is very high, it's not perfect. Identify and print the number of missing values for each of the 31 columns. What areas of the datafile have the lowest overall quality?

3. Compute and print the total number of flights for each of the participating airlines (i.e., AA, WN, etc).

4. The flights-usa-large2015.csv contains flights from 930 airports, but by design, numbers of airline flights are concentrated at hubs (this is the well known hub-and-spoke model).

Identify and print the number of flights leaving the five busiest airports in the US. You should find that the five busiest airports account for more than 20% of all flight departures in the US.

5. Create a dictionary that contains a list of the planes operated by each airline. Compute and print a summary of each airline and the size of its aircraft fleet. Your output might look something like:

```
AIRLINE        NO PLANES
========================
 AA -->  1045 planes ...
 AS -->   147 planes ...

 ... etc, etc, etc ...
========================
```

See python-code.d/pandas/TestDataGroupBy01.py and TestDataGroupBy02.py.

6. For all flights in the US, compute: (1) the total air time (it's more than 600 million minutes), (2) the minimum and maximum flight durations, (3) the total distance flown (think billions of miles), and (4) the average flight time.

7. At this point you should see that the way United Airlines uses its planes is completely different to Southwest Airlines.

Choose a representative plane from each airline and compute and print: (1) the total number of flights during 2015, (2) the minimum, maximum and average flight times, (3) the total air time, and (4) the total distance traveled during 2015.

8. Briefly summarize your conclusions in a few bullet points + text.