INFO370 Problem Set: Python, data frames (100 pt)

March 30, 2025

Instructions

This is the first problem set. It is worth 100 "PS points", that is equivalent to 10 points of your final grade. The goal of it is to get you going with the basic python and basic data frames.

Some background reading will be very helpful for those of you less familiar with Python: the **?** book, chapters 2 (data types), 3 (lists, dicts, sets), 4 (code structures); and **?**, Ch 4 and 5. Alternatively, the basics are also explained in python notes http://faculty.washington.edu/otoomet/machinelearning-py/python.html in a much less thorough manner. But be aware that python notes are not a substitute to a thorough background reading (like Lubanovic), and google search is a substitute for neither (the opposite is also true). In order to know a coding language well you need to understand the basic concepts, in class we will have time for a limited set of tools only. Google is great when you know what to ask, but to know what to ask–this is what books are for.

General requirements:

- Write well. Your submission should be easily readable. Readability is part of your grade!
- Add question numbers to your solutions. Question text is not necessary.
- Be sure that each graph or table adds value to your written explanation; avoid redundancy – you do no need four different visualizations of the same pattern.
- Don't output irrelevant information, or too much of relevant information. To print a few lines of data as illustration may be helpful. Printing thousands of lines of data will make your submission carbage.
- Working together is fun and useful but you have to submit your own work. Discussing the solutions and problems with your classmates is all right but do not copy-paste their solution! First understand it, and thereafter create your own solution. The **same applies for AI**! Please list all your collaborators!

  1. ...
  2. ...

  ...
- Attempt each question and document your reasoning process even if you cannot quite get there! In this way you can get at least partial credit!
- As the final submission, you should submit both your original notebook file (so you grader can actually run the code), and an html version of it (which is much faster to check).

1 Base python (32pt)

Often, we ask you to write textual answers and explain the results. However, in this basic programming task this may not be necessary.

Here is an example of an acceptable solution:

```
## 1. Temperature is 51F.  Compute temperature in C.
f = 51
c = (f - 32)/1.8
c

## 10.555555555555555
```

(Not much to explain here– 51F = 10.6C and that's it).

1.1 Computing (6pt)

Compute the following numbers, assign those to suitably named variables, and print:

1. (1pt) How many seconds are there in year?
2. (1pt) What is your age in seconds? Compute this based on your age.
3. (2pt) Create a logical variable that is true if you are more than 700M seconds old. Use logical operators, not if/else!
4. (2pt) Compute the following mathematical formulas:

$$p(1-p) \qquad \text{where } p = 0.4 \tag{1}$$

$$\frac{\alpha}{x_0}\left(1+\frac{x}{x_0}\right)^{-\alpha-1} \qquad \text{where } \alpha = 3,\ x_0 = 2, \text{ and } x = 2.5 \tag{2}$$

(This is more about understanding math and less about coding.)

Hint: the answer for (2) is approximately 0.0585.

1.2 Lists (6pt)

Consult python notes, ch 2.4.1 *Lists*.

1. (1pt) Create a list 'movies' that contains the names of at least six movies you like
2. (1pt) Using indexing and **slicing**, Create a list of three first movies in the list
3. (2pt) Use slicing and list concatenation to create a list of the first two and the last two movies
4. (2pt) Use *slicing* to list the movies in the opposite order

---

1.3 Loops/dicts (10pt)

1. (1pt) Create a list "numbers", numbers 70 through 79 in the following manner: create an empty list and add numbers to it in a loop (do not use list comprehension or 'list' function here!)

   Hint: Consult python notes, ch 2.4.1 *Lists*.

2. (1pt) Use a loop to compute the mean value of the list: add the values to their sum in a loop, and at the end divide the sum by the number of the values. (Do not use `mean` function!)

3. (1pt) Use loop to compute sum of all integers from 1 till 100.

4. (2pt) Assign people to seats. Consider names *Adam*, *Ashin*, *Inukai*, *Tanaka*, and *Ikki*; and seats 33, 12, 45, 2 and 17. Print seat assignments by assigning each name to the corresponding seat. For example, you can output

   Adam: 33
   Ashin: 12
   ...

   Hint: loop over the integer range of the length of names (here from 0 to 4, remember–python indexing is 0-based and thus counting doesn't start at 1!), and use indexing to access the corresponding name and seat number.

5. (2pt) You are given a list of numbers:

```
l = [1, 3, 5, 7, 11, 13, 17, 19]
```

   Use for-loop to create a dict that that links the numbers to their corresponding squares. The result should look like

```
{1: 1, 3: 9, 5: 25, ...}
```

   Hint: Consult python notes and ch 2.4.3 *Dicts*

6. (3pt) Use a for loop to compute the sum of all *values* in the dict above.

1.4 Functions (10pt)

1. (10pt) Write a function that takes in time in the numeric form (i.e. not a string) of HHMM (hours-minutes), and returns it in the numeric form of HH.HH (hours + fractions of hours). For instance, 1015 → 10.25 (10 hrs 15 mins → 10.25 hrs). Demonstrate it works using values 1015 and 345.

   The function should *return* (not print) the result as a *number*. You'll use this function below.

   Hint: use modulo operator % and integer division operator //. Modulo of 100 gives you minutes and integer division by 100 gives you hours

---

2 Basic data frames (68pt)

Your second task is to analyze flights out of NYC airports in 2013. Some of the questions only require a single number as answer (e.g. "how many flights to Seattle" will only require a single number). Others need a sufficient and clear explanation. For instance, "how good is the delay variable" needs explanations about what did you do, what did you find, and why do you think the result is good (or bad). In that case just plain output, with no explanation, will not count.

2.1 Setup (5pt)

In this problem set you will work with NYC flights data. The data is copied from the corresponding R package, you can read the documentation at e.g. RDocumentation. I guess the data originates from DOT Open Data Catalog, but seems like it has later been removed from that website.

Make sure you have read the background readings about pandas (see above).

1. (2pt) Load the data
2. (3 × 1pt) Do basic checks:
   (a) How many variables (columns) is there in the data? Ensure you know the variables in the data. Keep the documentation nearby.
   (b) How many rows of data is there?
   (c) print the first few lines of data. Does it look reasonable?

   Through the course we expect that you always do similar checks every time you load data.

2.2 Basic data exploration (18pt)

First, let's do some data exploration. Answer the following questions: show the code, the computation results, and comment the results in the accompanying text as appropriate.

1. (2pt) How many NYC airports are included in these data? Which airports are these?
2. (3pt) Into how many airports did the airlines fly from NYC in 2013?
3. (3pt) How many flights were there from NYC to Seattle (airport code *SEA*)?
4. (3pt) Are there any flights from NYC to Spokane (*GEG*)?
5. (4pt) Your boss wants you to know what is the "typical delay" in these data, but has to run to her next meeting and has no time to explain. Compute the "typical delay" and explain why this is a suitable measure.
6. (3pt) Did you remember to check how good is the delay variable? Are there missings? Are there any implausible or invalid entries? Go and check this if you haven't done it already.

2.3 Let's fly to Austin! (25pt)

Now let's see how is it to fly from NYC to Austin (airport code *AUS*).

1. (1pt) How many flights were there from NYC airports to Austin in 2013?
2. (2pt) How many airlines fly from NYC to Austin?
3. (2pt) Which are these airlines (just find the 2-letter abbreviations)? How many times did each of these go to Austin?
4. (2pt) What percentage of flights to Austin were delayed at arrival by more than 15 minutes?
5. (5pt) Use the hours-minutes function you made in Section 1.4 above to compute a new column: flight duration in hours. Use variables "arr time" and "dep time" to do this, do not use not "air time".
6. (3pt) What is the minimum value of the duration you computed? Explain why do you get negative values!

   Hint: the best way to understand why do you get weird results is to look at a few suspicious cases and replicate the computations manually.
7. (5pt) Change your computations so that you fix the negative durations. What is your shortest duration now?

   Hint: use `np.where`, not loops! (It is similar to `ifelse()` in R.)
8. (5pt) Compare your computed results with what you find in flight schedules (find a few direct flights on any booking website). Do you get similar results?

   Explain why do you get a result that is much shorter than the flight plans show!

2.4 Think about all this (15pt)

Finally, think about the questions and the analysis.

1. (5pt) What are your main concerns related to the analysis above? Were questions, answers, methodology your were able to use, and data good enough?
2. (5pt) Can you envision any business-relevant analysis you might do using this data? Hint: it does not have to be transport business, e.g. journalism is a business too.
3. (5pt) Do you see any ethical concerns with your analysis? Is it possible that as a result you put someone in a worse situation, even it they do not deserve it?

How much time did you spend?

And finally-finally, tell us how much time (how many hours) did you spend on this PS!