Lab 03 – pandas I

Submission: `lab03.ipynb`, pushed to your Git repo on `master`.
Points: 10
Due: Friday, January 25, 9:59am

Objectives

- `pandas`

Introduction

Numpy is a core, foundational library for use in many domains. It works well. It is efficient, having largely been written in the C language. And, it works quite well when your data is clean, structured, and uniform types. However, most real-world data is large, messy, heterogeneous, and often incomplete, missing values. Pandas has been developed for the data science community to aid in dealing with real world data.

Create a **`lab03.ipynb`** file. Create your header cell, then your first cell to set up your imports:

```
Import sys
import numpy as np
import pandas as pd
```

A large portion of this lab is taken from snippets scattered throughout the enormous documentation and tutorials from the pandas website at http://pandas.pydata.org/pandas-docs/stable/index.html . In particular, this short intro will get you up to speed: http://pandas.pydata.org/pandas-docs/stable/10min.html pretty quickly with the most common tasks you'll be doing. It doesn't do a great job really explaining the "how" and "why" that's happening behind the scenes, but you'll learn those through experience. I strongly recommend that you work through the 10 minute introduction first, and don't just skim it, but try out the exercises presented there. Then come back here to work through the exercises. More advanced pandas exercises are coming!

I also recommend the lynda.com (now Linkedin Learning) courses, which are available for free. Simply go there and log in through Bucknell, and have fun! Search for any skill you are after. There are no lack of material for searching `"Python data science"`.

Exercises

1) [P] Report the Python, Numpy and Pandas version numbers

2) [P] Show the result of `pd.show_versions()`.

3) [M] Explain the relationship between `numpy` and `pandas`. How are they tied together? What core functionality does `pandas` add to `numpy`?

4) [M] The two primary data structures in `pandas` are `Series` and `DataFrame`. Compare and contrast these two types

5) [M] What are the data structures that can be used to create a `Series` object in `pandas`?

6) [M] What are the data structures that can be used to create a `DataFrame` object in `pandas`?

7) [M] What role does the `index` parameter play when creating a `Series` object? Does the index always need to be specified? If not, under what conditions is it needed? When can it be implied?

Pay close attention to this NOTE from their Intro to Data Structures documentation.

> **Note:** When the data is a dict, and an index is not passed, the `Series` index will be ordered by the dict's insertion order, if you're using Python version >= 3.6 and Pandas version >= 0.23.
>
> If you're using Python < 3.6 or Pandas < 0.23, and an index is not passed, the `Series` index will be the lexically ordered list of dict keys.

We're going to work with a very simple set of data, just to get you started. Enter the following Python lists in a cell in your notebook. They will represent some fictitious daily quiz scores for a couple of weeks of some course you are taking:

```
days = ["Mon","Tue","Wed","Thu","Fri"]
scores_1 = [9.5, 8.75, 8, 10, 7.75]
scores_2 = [9, 8, 10, 8.75, 7.25]
```

8) [P] Convert `scores_1` and `scores_2` into two `Series` objects. Use `days` as your index. You should name each Series object using the name parameter as "week_1", and "week_2"

9) [P] Create a pandas `DataFrame` called `scores`, that represents the above data. Show your data frame. Your results should be arranged as follows: (NOTE: depending on how you construct your `DataFrame`, you might need to transpose your data frame using the `T` operator):

	Mon	Tue	Wed	Thu	Fri
week_1	9.5	8.75	8.0	10.00	7.75
week_2	9.0	8.00	10.0	8.75	7.25

10) [P] Append one more week of data to scores. The data you are appending is `[8.5,8,9.75,9,6]`. The row label on the new week is `"week_3"`. Show your updated data frame. It should look like the following:

	Mon	Tue	Wed	Thu	Fri
week_1	9.5	8.75	8.00	10.00	7.75
week_2	9.0	8.00	10.00	8.75	7.25
week_3	8.5	8.00	9.75	9.00	6.00

You are about to practice a lot of data selection and manipulation techniques. Developing the ability to quickly select data you are looking for is an important skill. Some exericses will be presented here, just to get you thinking about the powerful flexibility you have in this feature. Remember, this is a simple toy example so you

can observed that your output indeed matches the correct value.

11) [M] Pandas has an ENORMOUS number of ways you can select your data. At first, the flexibility will confuse and drive you nuts. In time, it because amazing and intuitive (with less nuts.) From their 10 minute tutorial:

> **Note:** While standard Python / Numpy expressions for selecting and setting are intuitive and come in handy for interactive work, for production code, we recommend the optimized pandas data access methods, `.at`, `.iat`, `.loc` and `.iloc`.

Write yourself a quick one sentence reference for each access method listed above, with one example for each using the `scores` DataFrame.

Of course, in addition to the above access methods, you can use the `[ ]` operator to access your data. Oh, and DataFrames create attributes for each column of data you have. For example, scores["Mon"], or scores.Mon could be used, respectively. From the 10 minute tutorial, briefly summarize what you can in your own notes. Find some good cheat sheets, or make your own. AND – PAY CLOSE ATTENTION TO THE RETURN TYPES! Some return DataFrame objects, and other techniques return Series objects.

12) [P] Show at least two different techniques to select scores for Monday using the string `"Mon"`

13) [P] Show how to retrieve the scores for Tuesday using the named attribute `Tue`

14) [P] Show at least three techniques to select scores for Wednesday using an integer.

15) [P] Select the data for the first week using the string `"week_1"`

16) [P] Select the data for the first week using a slice (HINT: `0:1`)

17) [P] Show at least two techniques to select the data for the second week using an integer

18) [P] Select Monday and Friday of the first and third week.

19) [P] Report the mean for each week

20) [P] For each week, report how much each day's score for that week differed from the mean for the week

21) [P] Report the maximum score for each week

22) [P] For each week, report which day had the largest score

23) [P] Report the scores rescaled to fall between 0 and 100, instead of 0 to 10 as they are now.

Deliverables

Commit and push `lab03.ipynb`. Be sure you have every cell run, and output generated. Verify that your file is pushed properly on Gitlab.