Contents

Assignment 2: Practicing Python and Accessing Data

Due: 2020-09-20

Objective & Evaluation

This assignment is an opportunity to earn level 1 or 2 achievements in `python`, `process` and `access` and begin working toward level 1 in `summarize`.

Accept the assignment on GitHub Classroom. It contains a notebook with some template structure (and will set you up for grading). The template will also convert notebooks that are added to markdown, which makes reading on GitHub for easier grading. If you want to incorporate feedback you receive back into a notebook file, Jupytext can do that.

To work with this notebook you can either:

- download the repository as .zip from the green code button, unzip, and re-upload, OR
- clone the repository with git and the push your changes. See Git/GitHub help on cloning, committing, and pushing, for example this tutorial on git to learn more about git.

Accessing Data with Python and pandas

(for python and access)

Find 3 datasets of interest to you that are provided in different file formats. Choose datasets that are not too big, so that they do not take more than a few second to load. At least one dataset, must have non numerical (eg string or boolean) data in at least 1 column. Complete a dictionary for each with the url, a name, and what function should be used to load the data into a `pandas.DataFrame`.

Use a list of those dictionaries to iterate over the datasets and build a table that describes them, with the following columns `['name','source','num_rows', 'num_columns','source_file_name']`. The source column should be the url where you loaded the data from or the source if you downloaded it from a website first The `source_file_name` should be the part of the url after the last `/`, you should extract this programmatically. Display that summary table as a dataframe and save it as a csv, named `dataset_summary.csv`.

For one dataset (must include nonnumerical data):

- display the heading with the last seven rows
- make and display a new data frame with only the non numerical columns
- was the format that the data was provided in a good format? why or why not?

> Tip
>
> You can put an if statement in a list comprehension.
>
> Even more pythonic: you can index pandas objects (DataFrames, Series, and Index– remember that's what the df.columns was–) with logical expressions.

> Tip
>
> Urls are strings. The `string` class in python has methods for manipulating strings, like `split`.

> Note
>
> If you download the datasets (or find them as .zip and need to) you can use the local path instead of the url, but include a markdown cell with links to where you got your data from.

> Tip
>
> You can create a `pandas DataFrame` using the

For a second dataset:

- display the heading and the first three rows
- display the datatype for each column
- Are there any variables where pandas may have read in the data as a datatype that's not what you expect (eg a numerical column mistaken for strings)?

For the third dataset:

- display the first 5 even rows of the data for three columns of your choice

For any dataset:

- try reading it in with the wrong `read_` function. If you had done this by accident, how could you tell?

Data Science Process

(for the `process` skill)

Make a list of a data science pipeline and denote which types of programming might be helpful at each staged.
Include this in a markdown cell in the same notebook with your analysis

---

By Professor Sarah M Brown