H. Diana McSpadden (hdm5s)

Lab Assignment 1: How to Get Yourself Unstuck

DS 6001: Practice and Application of Data Science

Instructions

Problem 0

Import the following libraries

In [ ]:
```
# import common libraries
import numpy as np
import pandas as pd
import os
import math
```

Problem 1

Python is open-source, and that's beautiful: it means that Python is maintained by a world-wide community of volunteers, that Python develops at the same rate as advancements in science, and that Python is completely free of charge. But one downside of being open-source is that different people design many alternative ways to perform the same task in Python.

Read the following Stack Overflow post: https://stackoverflow.com/questions/11346283/renaming-columns-in-pandas/46912050. The question is simply how to rename the columns of a dataframe using Pandas. Count how many unique different solutions were proposed, and write this number in your lab report. (Hint: the number of solutions is not the number of answers to the posted question.)

Remember: your goal as a data scientist needs to be to process/clean/wrangle/manage data as quickly as possible while still doing it correctly. A big part of that job is knowing how to seek help to find the right answer quickly. Given the number of proposed solutions

on this Stack Overflow page, what's the problem with developing a habit of using Google and Stack Overflow as your first source for seeking help? (2 points)

Answer:

Short Answer: There are at least 26 answers to this question. The problem with using Google/StackOverflow/ChatGPT first before trying to write pseodocode or an initial algorithm first is that you may not understand the basics of your problem and your may over or under engineer your problem by using another developer's code. Some of the answers presented on this Stack Overflow post are just developers having fun by trying to be as overly complex as possible, or trying to come up with special cases. It is unclear in the original question whether the user was trying to write re-usable, future-proof code, or just a one-time column rename. How to write the code would depend on that context. Only you may know the specific context of your data science task, and beware asking for help from people who may now appreciate the nuance of your task.

TLDR Answer: The question is how to rename columns in a pandas dataframe. Pretty straightforward question. I'll try to count the answers in different proposed solutions in the stackoverflow post:

1. There is one response to check the official pydata documentation: pandas.pydata.org/pandas-docs/stable/user_guide/text.html
2. There is an option to use df.rename with only the columns parameter.
3. There is a proposal to use pd.DataFrame function pd.DataFrame(...) with the columns='' parameter.
4. There is a proposal to use df.rename with a dictionary of key value pairs of old column name new columns name with axis = 1.
5. There is a proposal to use df.rename with a dictionary of key value pairs of old column name new columns name with axis = 'columns'.
6. There is a proposal to use df.rename with a dictionary of key value pairs of old column name new columns name with no axis parameter.
7. The three previous methods are described with and without inplace=True, so that you do or do not need to assign back to a new dataframe variable.
8. A option ot us df.set_axis() with axis=1 and inplace=False to return a copy is proposed.
9. Also, the method of just using df.columns = lst_of_column_names is also described.
10. There is also a description of how to rename columns that need to replace a value using a lambda function.
11. Futher down in the stackoverflow chain there are responses that show that the lambda function method has changed syntax a few times since the original version.
12. There is a description of how to rename columns that need to strip a character/characters from the column names by using str.lstrip

13. There is a description of how to rename columns that need to replace part of the column name string using the string replace function using df.columns = df.columns.str.replace('$', '')
14. There is a method proposes that uses pd.concat([c for _, c in df.items()], axis=1, keys=new)
15. There is a method that proposes reconstructing the dataframe using: pd.DataFrame(df.values, df.index, new). The author says this should only work if there is a single dtype for all columns.
16. If there are multiple dtypes for columns the author says this method may work: pd.DataFrame(df.values, df.index, new).astype(dict(zip(new, df.dtypes)))
17. The same author also proposes this solution using transpose() and set_index() for single dtype dataframes: df.T.set_index(np.asarray(new)).T
18. And this for multi dtype dataframes: df.T.set_index(np.asarray(new)).T.astype(dict(zip(new, df.dtypes)))
19. And this different type using lambda: df.rename(columns=lambda x, y=iter(new): next(y))
20. A list comprehension method is described: df.columns = [col.strip('$') for col in df.columns]
21. Using str.slice(1) is offered as a solution: df.columns = df.columns.str.slice(1)
22. Ooo, regex is described: df = df.rename(columns=lambda x: re.sub('$','',x))
23. Someone describes using a delimiter and "future proofing" the solution. (responders are just getting silly at this point)
24. A Multi Index solution is offered with comma separated indices supported so that characters could be stripped from column headers that weren't sequential.
25. Someone provided an entire function that will replace column header names that does error checking on whether column names exist.
26. Further down a respondent describes creating a dictionary map using the code:

```
new_cols = ['a', 'b', 'c', 'd', 'e']
new_names_map = {df.columns[i]:new_cols[i] for i in range(len(new_cols))}

df.rename(new_names_map, axis=1, inplace=True)
```

Problem 2

There are several functions implemented in Python to calculate a logarithm. Both the numpy and math libraries have a log() function. Your task in this problem is to calculate log base 3 (7) directly (without using the change-of-base formula). Note that this particular log has a base of 3, which is unusual. For this problem:

- Write code to display the docstrings for each function.

- Read the docstrings and explain, in words in your lab report, whether it is possible to use each function to calculate log or not. Why did you come to this conclusion?

If possible, use one or both functions to calculate log and display the output. (2 points)

Answer:

As shown in the documentation shown in the docstring print out shown below, math.log accepts an argument for base, while numpy.log does not. Because of this, I calculate the log base 3 of 7 using the python math library, but I do not calculate it using the numpy library.

In [ ]:
```
# display the docstring for the math.log function
print('----' * 10)
print('Docstring for math.log')
print(math.log.__doc__)

# using the math.log
print('Using math.log: The log base 3 of 7 is: ',math.log(7,3))
```

```
----------------------------------------
Docstring for math.log
log(x, [base=math.e])
Return the logarithm of x to the given base.

If the base not specified, returns the natural logarithm (base e) of x.
Using math.log: The log base 3 of 7 is:  1.7712437491614221
```

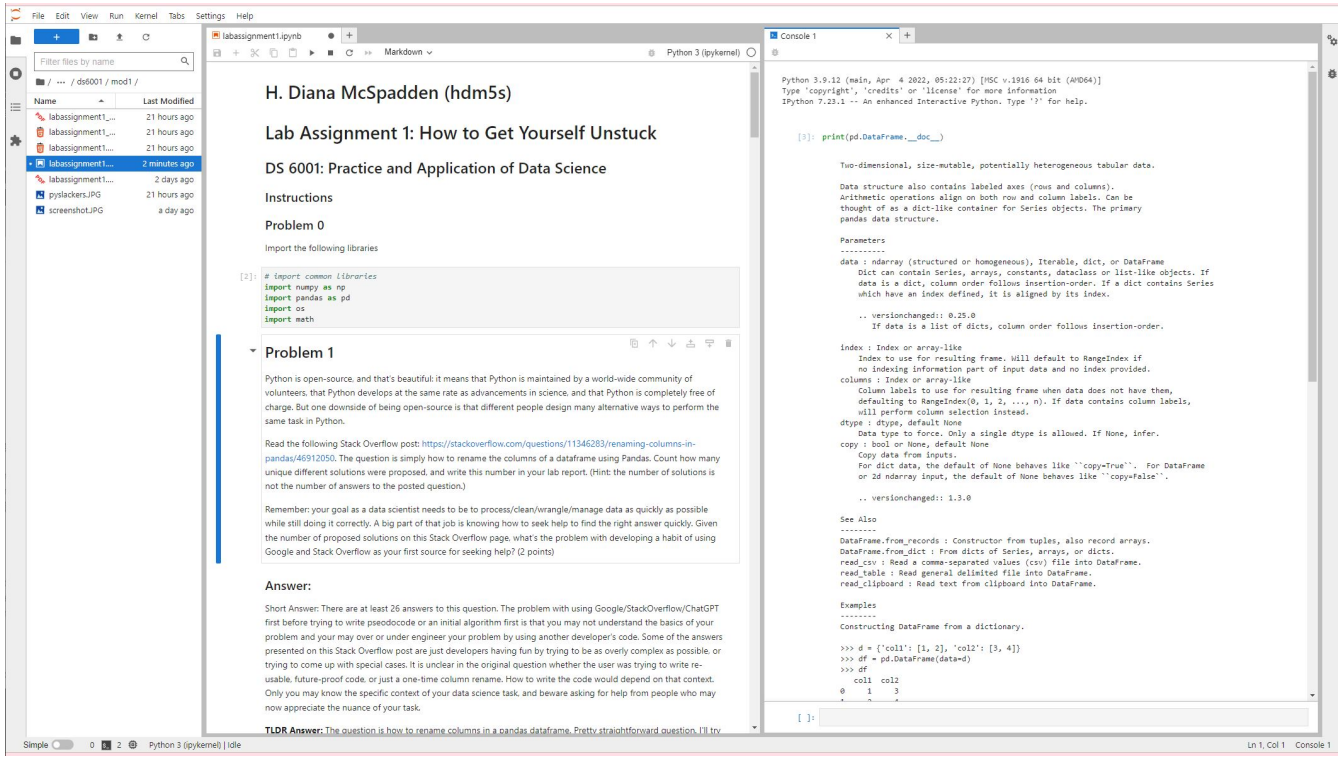
In [ ]:
```
# display the docstring for the numpy.log function
print('----' * 10)
print('Docstring for np.log')
print(np.log.__doc__)
```

```
----------------------------------------
Docstring for np.log
log(x, /, out=None, *, where=True, casting='same_kind', order='K', dtype=None, subok=True[, signature, extobj])

Natural logarithm, element-wise.

The natural logarithm `log` is the inverse of the exponential function,
so that `log(exp(x)) = x`. The natural logarithm is logarithm in base
`e`.

Parameters
----------
x : array_like
    Input value.
out : ndarray, None, or tuple of ndarray and None, optional
    A location into which the result is stored. If provided, it must have
    a shape that the inputs broadcast to. If not provided or None,
    a freshly-allocated array is returned. A tuple (possible only as a
    keyword argument) must have length equal to the number of outputs.
where : array_like, optional
    This condition is broadcast over the input. At locations where the
    condition is True, the `out` array will be set to the ufunc result.
    Elsewhere, the `out` array will retain its original value.
    Note that if an uninitialized `out` array is created via the default
    ``out=None``, locations within it where the condition is False will
    remain uninitialized.
**kwargs
    For other keyword-only arguments, see the
    :ref:`ufunc docs <ufuncs.kwargs>`.

Returns
-------
y : ndarray
    The natural logarithm of `x`, element-wise.
    This is a scalar if `x` is a scalar.

See Also
--------
log10, log2, log1p, emath.log

Notes
-----
Logarithm is a multivalued function: for each `x` there is an infinite
number of `z` such that `exp(z) = x`. The convention is to return the
`z` whose imaginary part lies in `[-pi, pi]`.
```

```
For real-valued input data types, `log` always returns real output. For
each value that cannot be expressed as a real number or infinity, it
yields ``nan`` and sets the `invalid` floating point error flag.

For complex-valued input, `log` is a complex analytical function that
has a branch cut `[-inf, 0]` and is continuous from above on it. `log`
handles the floating-point negative zero as an infinitesimal negative
number, conforming to the C99 standard.

References
----------
.. [1] M. Abramowitz and I.A. Stegun, "Handbook of Mathematical Functions",
       10th printing, 1964, pp. 67.
       https://personal.math.ubc.ca/~cbm/aands/page_67.htm
.. [2] Wikipedia, "Logarithm". https://en.wikipedia.org/wiki/Logarithm

Examples
--------
>>> np.log([1, np.e, np.e**2, 0])
array([  0.,   1.,   2., -Inf])
```

Problem 3

Open a console window and place it next to your notebook in Jupyter labs. Load the kernel from the notebook into the console, then call up the docstring for the pd.DataFrame function. Take a screenshot and include it in your lab report. (To include a locally saved image named screenshot.jpg , for example, create a Markdown cell and paste `<img src="screenshot.jpg" width=600>` (2 points)



Problem 4

Search through the questions on Stack Overflow tagged as Python questions: https://stackoverflow.com/questions/tagged/python. Find a question in which an answerer exhibits passive toxic behavior as defined in this module's notebook. Provide a link, and describe what specific behavior leads you to identify this answer as toxic. (2 points)

Answer:

The first response in this post is pretty rude: https://stackoverflow.com/questions/44134979/python-code-for-parsing-mysql-dump-file-and-extract-useful-data-from-it.

The response is "This is completely incomprehensible. Try adding the supposed input, the expected output, and a pseudocode, at least. – Riccardo Petraglia May 23, 2017 at 12:41"

Yes, the question's grammar is not perfect. Yes, the question includes an image instead of code which is not good Stack Overflow etiquette. However, the phrase "completely incomprehensible" is rude, and is "...at least". If the comment at removed the first sentence

and the ", at least." and asked the original poster for the input, expected output, pseudocode, and intended purpose, then the comment would have served its purpose without possibly causing shame to the original poster.

Problem 5

Search through the questions on Stack Overflow tagged as Python questions: https://stackoverflow.com/questions/tagged/python.

Find a question in which a questioner selfsabotages by asking the question in a way that the community does not appreciate. Provide a link, and describe what the questioner did specifically to annoy the community of answerers. (2 points)

Answer:

Here is an example of self-sabotage: https://stackoverflow.com/questions/75133238/when-i-copy-path-then-this-error-is-found-please-help-and-gave-me-solutions-than

The original poster only includes an error, but not the code that generates the error. Repeatedly the poster is asked to provide the code the creates the error and is provided links to the "How To Ask" documentation. If the original poster had included code that causes the error, and the purpose of the code, there were at least three StackOverflow users who had been willing to look at their question.

Problem 6

These days there are so many Marvel superheros, but only six superheros count as original Avengers: Hulk, Captain America, Iron Man, Black Widow, Hawkeye, and Thor. I wrote a function, `is_avenger()` , that takes a string as an input.

The function looks to see if this string is the name of one of the original six Avengers. If so, it prints that the string is an original Avenger, and if not, it prints that the string is not an original Avenger. Here's the code for the function:

In [ ]:
```
def is_avenger(name):
    if name=="Hulk" or "Captain America" or "Iron Man" or "Black Widow" or "Hawkeye" or "Thor":
        print(name + "'s an original Avenger!")
    else:
        print(name + " is NOT an original Avenger.")
```

To test whether this function is working, I pass the names of some original Avengers to the function:

In [ ]:
```
is_avenger("Black Widow")
```

```
Black Widow's an original Avenger!
```

In [ ]:
```
is_avenger("Iron Man")
```

```
Iron Man's an original Avenger!
```

In [ ]:
```
is_avenger("Hulk")
```

```
Hulk's an original Avenger!
```

Looks good! But next, I pass some other strings to the function:

In [ ]:
```
is_avenger("Spiderman")
```

```
Spiderman's an original Avenger!
```

In [ ]:
```
is_avenger("Beyonce")
```

```
Beyonce's an original Avenger!
```

Beyonce is a hero, but she was too busy going on tour to be in the Avengers movie. Also, Spiderman definitely was NOT an original Avenger. It turns out that this function will display that any string we write here is an original Avenger, which is incorrect. To fix this function, let's turn to Stack Overflow.

Part a

The first step to solving a problem using Stack Overflow is to do a comprehensive search of available resources to try to solve the problem. There is a post on Stack Overflow that very specifically solves our problem. Do a Google search and find this post. In your lab report, write the link to this Stack Overflow page, and the search terms you entered into Google to find this page. Then apply the solution on this Stack Overflow page to fix the is_avenger() function, and test the function to confirm that it works as we expect. (2 points)

Part a Answer:

My Google Search: "StackOverflow python if statement with or's always returns true"

Here was the StackOverflow article: https://stackoverflow.com/questions/69520008/if-statement-returns-true-when-it-is-not-python

which is exactly the problem with the `is_avenger()` function.

In [ ]:
```
def is_avenger(name):
    if name in ["Hulk","Captain America","Iron Man","Black Widow","Hawkeye","Thor"]:
        print(name + "'s an original Avenger!")
    else:
        print(name + " is NOT an original Avenger.")
```

In [ ]:
```
test_names = ["Hulk","Captain America","Iron Man","Black Widow","Hawkeye","Thor","Spiderman","Beyonce"]
for name in test_names:
    is_avenger(name)
```

```
Hulk's an original Avenger!
Captain America's an original Avenger!
Iron Man's an original Avenger!
Black Widow's an original Avenger!
Hawkeye's an original Avenger!
Thor's an original Avenger!
Spiderman is NOT an original Avenger.
Beyonce is NOT an original Avenger.
```

Yay! It seems to be working as expected now.

Part b

Suppose that no Stack Overflow posts yet existed to help us solve this problem. It would be time to consider writing a post ourselves. In your lab report, write a good title for this post. Do NOT copy the title to the posts you found for part a. (Hint: for details on how to write a good title see the slides or https://stackoverflow.com/help/how-to-ask) (3 points)

Title: Why does my python if statement always evaluate to true using python v 3.9.5?

Tags: [python]

Body: My function takes a string and if the string is one of six possible values the function prints [phrase A], otherwise the function should print [phrase B]. However, the function always prints [phrase A]. In this case, [phrase A] is [name]'s an original Avenger! and phrase B is [name] is NOT an original Avenger. Where [name] is my input string. I am using python v. 3.9.5, Windows, and VSCode IDE.

```
def is_avenger(name):
    if name=="Hulk" or "Captain America" or "Iron Man" or "Black Widow" or "Hawkeye" or "Thor":
        print(name + "'s an original Avenger!")
    else:
        print(name + " is NOT an original Avenger.")

test_names = ["Hulk","Captain America","Iron Man","Black
Widow","Hawkeye","Thor","Spiderman","Beyonce"]
for name in test_names:
    is_avenger(name)
```

EXPECTED RESULTS:

Hulk's an original Avenger!

Captain America's an original Avenger!

Iron Man's an original Avenger!

Black Widow's an original Avenger!

Hawkeye's an original Avenger!

Thor's an original Avenger!

Spiderman is NOT an original Avenger.

Beyonce is NOT an original Avenger.

ACTUAL RESULTS

Hulk's an original Avenger!

Captain America's an original Avenger!

Iron Man's an original Avenger!

Black Widow's an original Avenger!

Hawkeye's an original Avenger!

Thor's an original Avenger!

Spiderman's an original Avenger.

Beyonce's' an original Avenger.

As you can see Spiderman and Beyonce are somehow passing the if statement when I would not expect them to.

Thanks!

Part c

One characteristic of a Stack Overflow post that is likely to get good responses is a minimal working example. A minimal working example is code with the following properties:

- It can be executed on anyone's local machine without needing a data file or a hard-to-get package or module
- It always produces the problematic output.
- . It using as few lines of code as possible, and is written in the simplest way to write that code

Write a minimal working example for this problem. (3 points)

In [ ]:
```
# I believe I did that above, but here it is again:
def is_avenger(name):
    if name=="Hulk" or "Captain America" or "Iron Man" or "Black Widow" or "Hawkeye" or "Thor":
        print(name + "'s an original Avenger!")
    else:
        print(name + " is NOT an original Avenger.")

test_names = ["Hulk","Captain America","Iron Man","Black Widow","Hawkeye","Thor","Spiderman","Beyonce"]
for name in test_names:
    is_avenger(name)


#EXPECTED RESULTS:
#Hulk's an original Avenger!
#Captain America's an original Avenger!
#Iron Man's an original Avenger!
#Black Widow's an original Avenger!
#Hawkeye's an original Avenger!
#Thor's an original Avenger!
```

```
#Spiderman is NOT an original Avenger.
#Beyonce is NOT an original Avenger.
```

```
Hulk's an original Avenger!
Captain America's an original Avenger!
Iron Man's an original Avenger!
Black Widow's an original Avenger!
Hawkeye's an original Avenger!
Thor's an original Avenger!
Spiderman's an original Avenger!
Beyonce's an original Avenger!
```

Problem 7

Sign on to the PySlackers slack page and send me a private message in which you tell me which three channels on that Slack workspace look most interesting to you. (2 points)

Answer: Done.