Kiana Dane

Lab 1 Notebook - 06/22/24

DS 6001 Kropko

Problem 0

In [ ]:

```
import numpy as np
import pandas as pd
import os
import math
```

Problem 1

In [ ]:

```
## I would argue that ten or eleven different solutions were proposed,
## depending on whether we count certain identical first steps from differen
### as the same solution or not.
```

In [ ]:

```
## An issue with developing a habit of using stack overflow is that there ar
## so often many ways to approach and solve a problem, some of which may
## be better than others - and we need to know how to think critically
## about the problem rather than apply a one-sized solution.
```

Problem 2

In [ ]:

```
np.log.__doc__
```

Out[ ]:

```
'log(x, /, out=None, *, where=True, casting=\'same_kind\', order=\'K\', dty
pe=None, subok=True[, signature])\n\nNatural logarithm, element-wise.\n\nTh
e natural logarithm `log` is the inverse of the exponential function,\nso t
hat `log(exp(x)) = x`. The natural logarithm is logarithm in base\n`e`.\n\n
Parameters\n----------\nx : array_like\n    Input value.\nout : ndarray, No
ne, or tuple of ndarray and None, optional\n    A location into which the r
esult is stored. If provided, it must have\n    a shape that the inputs bro
adcast to. If not provided or None,\n    a freshly-allocated array is retur
ned. A tuple (possible only as a\n    keyword argument) must have length eq
ual to the number of outputs.\nwhere : array_like, optional\n    This condi
tion is broadcast over the input. At locations where the\n    condition is 
True, the `out` array will be set to the ufunc result.\n    Elsewhere, the 
`out` array will retain its original value.\n    Note that if an uninitiali
zed `out` array is created via the default\n    ``out=None``, locations wit
hin it where the condition is False will\n    remain uninitialized.\n**kwar
gs\n    For other keyword-only arguments, see the\n    :ref:`ufunc docs <uf
uncs.kwargs>`.\n\nReturns\n-------\ny : ndarray\n    The natural logarithm 
of `x`, element-wise.\n    This is a scalar if `x` is a scalar.\n\nSee Also
\n--------\nlog10, log2, log1p, emath.log\n\nNotes\n-----\nLogarithm is a m
ultivalued function: for each `x` there is an infinite\nnumber of `z` such 
that `exp(z) = x`. The convention is to return the\n`z` whose imaginary par
t lies in `(-pi, pi]`.\n\nFor real-valued input data types, `log` always re
turns real output. For\neach value that cannot be expressed as a real numbe
r or infinity, it\nyields ``nan`` and sets the `invalid` floating point err
or flag.\n\nFor complex-valued input, `log` is a complex analytical functio
n that\nhas a branch cut `[-inf, 0]` and is continuous from above on it. `l
og`\nhandles the floating-point negative zero as an infinitesimal negative 
\nnumber, conforming to the C99 standard.\n\nIn the cases where the input h
as a negative real part and a very small\nnegative complex part (approachin
g 0), the result is so close to `-pi`\nthat it evaluates to exactly `-pi
`.\n\nReferences\n----------\n.. [1] M. Abramowitz and I.A. Stegun, "Handbo
ok of Mathematical Functions",\n       10th printing, 1964, pp. 67.\n
https://personal.math.ubc.ca/~cbm/aands/page_67.htm\n.. [2] Wikipedia, "Log
arithm". https://en.wikipedia.org/wiki/Logarithm\n\nExamples\n--------\n>>>
np.log([1, np.e, np.e**2, 0])\narray([  0.,   1.,   2., -inf])'
```

In [ ]:

```
math.log.__doc__
```

Out[ ]:

```
'log(x, [base=math.e])\nReturn the logarithm of x to the given base.\n\nIf
the base is not specified, returns the natural logarithm (base e) of x.'
```

In [ ]:

```
tar = 7
bse = 3
ans = math.log(tar,bse)
ans
```

Out[ ]:

```
1.7712437491614221
```

In [ ]:

```
np.version.version
```

Out[ ]:

```
'2.0.0'
```

Problem 3

screenshot of Dane's screen showing docstring for DataFrame function in pandas

No description has been provided for this image

Problem 4

In this question, OP states they "have been asked to write a script" and asks for advice on which modules to use to achieve their goal. Most comments offered the Python subprocess module to call git, which OP eventually marked as the answer. Our commenter, however, had to "whip out [their] dick and knowledge drop" how to achieve a solution using a shell script. Commenter writes: "If you're on Linux or Mac, why use python at all for this task? Write a shell script." ** **GROAN** **

https://stackoverflow.com/a/11114295

Problem 5

https://stackoverflow.com/questions/49579846/how-to-combine-numbers

It is clear that this question asker did not put in much, if any effort to find the answer to their problem before posting the thread on Stack Overflow. This is a pretty easy way to annoy the community at large and get nasty responses. The lack of clarity in the question asker's communication of the objective to (we think) concatenate numbers, from the title to the example, probably annoyed those "neckbeards" and ultimately made them downvote the asker's question.

Problem 6

In [ ]:

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

is_avenger("Beyonce")
```

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

Problem 6 Part a

- Link to Stack Overflow: https://stackoverflow.com/questions/48465536/using-or-in-if-statement-python

- The search terms I used to find the article were "use or and if together python."

In [ ]:

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

is_avenger("Beyonce")
```

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

Problem 6 Part b

My Stack Overflow question title for this question would be:

"How do I use "or" in an if statement to check if a value is in a list?"

Problem 6 Part c

In [ ]:

```
def is_primary(color):
    if color  == "Red" or "Blue" or "Yellow":
        print(color + "'s a primary color!")
    else:
        print("color + is NOT a primary color.")
```

Problem 7

In [ ]:

```
jobs = pd.read_csv("jobs_in_data.csv")
jobs
```

Out[ ]:

	work_year	job_title	job_category	salary_currency	salary	salary_in_usd	er
0	2023	Data DevOps Engineer	Data Engineering	EUR	88000	95012	
1	2023	Data Architect	Data Architecture and Modeling	USD	186000	186000	
2	2023	Data Architect	Data Architecture and Modeling	USD	81800	81800	
3	2023	Data Scientist	Data Science and Research	USD	212000	212000	
4	2023	Data Scientist	Data Science and Research	USD	93300	93300	
...	...	...	...	...	...	...	
9350	2021	Data Specialist	Data Management and Strategy	USD	165000	165000	
9351	2020	Data Scientist	Data Science and Research	USD	412000	412000	
9352	2021	Principal Data Scientist	Data Science and Research	USD	151000	151000	
9353	2020	Data Scientist	Data Science and Research	USD	105000	105000	
9354	2020	Business Data Analyst	Data Analysis	USD	100000	100000	

9355 rows × 12 columns

Problem 7 Part a

Code from ChatGPT:

In [ ]:

```python
import pandas as pd
# Load the dataset

jobs_df = pd.read_csv("jobs_in_data.csv")

# Filter the DataFrame to include only the relevant job titles
job_titles = ['Data Analyst', 'Data Engineer', 'Data Scientist', 'Machine Le
filtered_df = jobs_df[jobs_df['job_title'].isin(job_titles)]

# Group by job title and work year, then calculate the average salary
```

```
average_salaries = filtered_df.groupby(['job_title', 'work_year'])['salary_i

# Select only the years 2022 and 2023
average_salaries = average_salaries[[2022, 2023]]

# Display the resulting table
print(average_salaries)
```

```
work_year                      2022      2023
job_title
Data Analyst               108658.0  110988.0
Data Engineer              139803.0  149945.0
Data Scientist             138529.0  163714.0
Machine Learning Engineer  151775.0  191026.0
```

Correspondence with ChatGPT

kiana: hi

```
GPT: Hello! How can I assist you today?
```

kiana: i've got some python homework

```
GPT: Sure, I'd be happy to help with your Python homework.
What specific problems or concepts are you working on?
```

kiana: working with dataframes and pandas

```
GPT: Great! Pandas is a powerful library for data
manipulation and analysis in Python. What specific tasks or
problems are you trying to solve with DataFrames? If you have
any code or data that you are working with, feel free to
share it!
```

kiana: i'd like to share a dataset

```
GPT: Sure! Please go ahead and upload your dataset. Once I
have the data, I can help you with any questions or tasks you
have related to it.
```

* Kiana shares the "jobs_in_data.csv" file with the attachment tool.

```
GPT: Thanks for sharing your dataset! Here's an overview of
the columns in your `jobs_in_data.csv` file:
```

```
1. **work_year**: Year of the job entry.
2. **job_title**: Title of the job.
3. **job_category**: Category of the job.
4. **salary_currency**: Currency of the salary.
5. **salary**: Salary in the given currency.
6. **salary_in_usd**: Salary converted to USD.
7. **employee_residence**: Country of residence of the
employee.
8. **experience_level**: Experience level required for the
job.
9. **employment_type**: Type of employment (e.g., Full-time).
10. **work_setting**: Work setting (e.g., Hybrid, In-person).
11. **company_location**: Location of the company.
12. **company_size**: Size of the company (e.g., L for Large,
M for Medium).

How would you like to proceed? Do you have any specific
analyses or operations you'd like to perform on this data?
```

Kiana (copied and pasted from Kropko's assignment):

Our goal is to manipulate the jobs dataframe to create a table with four rows: one for each of the job titles Data Analyst, Data Engineer, Data Scientist, and Machine Learning Engineer; and two columns: one for the year 2022 and one for 2023.

Inside each cell should be the average salary (salary_in_usd) for that job title and year, rounded to the nearest dollar.

```
GPT: Here's the table showing the average salary (in USD) for
the job titles Data Analyst, Data Engineer, Data Scientist,
and Machine Learning Engineer for the years 2022 and 2023:

| Job Title                 | 2022      | 2023      |
|---------------------------|-----------|-----------|
| Data Analyst              | $108,658  | $110,988  |
| Data Engineer             | $139,803  | $149,945  |
| Data Scientist            | $138,529  | $163,714  |
| Machine Learning Engineer| $151,775  | $191,026  |

If you need any further analysis or modifications, let me
know!
```

kiana: thanks for the table! I'm looking for the python code to execute on this dataset in order to get this table as the output.

```
GPT: Sure! Here is the complete Python code to create the
table with the average salaries for the specified job titles
```

and years:

In [ ]:

```
import pandas as pd

# Load the dataset
file_path = 'path_to_your_dataset/jobs_in_data.csv'
jobs_df = pd.read_csv(file_path)

# Filter the DataFrame to include only the relevant job titles
job_titles = ['Data Analyst', 'Data Engineer', 'Data Scientist', 'Machine Le
filtered_df = jobs_df[jobs_df['job_title'].isin(job_titles)]

# Group by job title and work year, then calculate the average salary
average_salaries = filtered_df.groupby(['job_title', 'work_year'])['salary_i

# Select only the years 2022 and 2023
average_salaries = average_salaries[[2022, 2023]]

# Display the resulting table
print(average_salaries)
```

GPT: Replace 'path_to_your_dataset/jobs_in_data.csv' with the actual path to your dataset file. Running this code will give you the desired table with average salaries rounded to the nearest dollar.

kiana: love you

GPT: I'm glad I could help! If you have any more questions or need further assistance, feel free to ask. Happy coding!

Problem 7 Part b

I think using ChatGPT for data wrangling before analysis has tremendous potential to assist myself and colleagues by providing us with a place to start, or a skeleton (in some cases with gpt code, zombies) with which to fill up with organs and blood, or data sources and legitimate logic in this metaphor. It should not be relied upon as a source of truth, ever (just like solutions one may find on stack overflow) . ALL code that comes from a generative AI tool should be checked, tested and amended before ever becoming part of any larger chunk of code.