HW 1
Post lecture

6. The df.describe() function in pandas gives you a quick summary of your data, in my owns words, it feels like a tool that helps you organize you statistics/data and maybe if you have some questions, you could ask by typing code and i will automatically generate the answer for you, for example, “what's the average number?”, and there are few functions that it will privde you in df.describe(), 'count', 'mean', 'std', 'min', '25%', '50%', '75%', and 'max'.

- **Count**: The number of non-missing (non-NaN) values for each column.
    - This tells you how many valid (non-null) data points are present for each variable.
- **Mean**: The average
    - Calculated as the sum of all values divided by the count of the values. It gives a central value for the data.
- **Standard Deviation (std)**: A measure of the dispersion or spread of the values in the column.
    - It quantifies how much the values in the dataset deviate from the mean. A higher standard deviation means the data points are spread out over a wider range of values.
- **Minimum (min)**: The smallest number.
- **25% (1st Quartile or Q1)**: The value below which 25% of the data falls.
    - It is the first quartile, which is the middle value between the smallest number (min) and the median.
- **50% (Median or Q2)**: The middle value of the data when it is sorted in ascending order.
    - The median is the point at which 50% of the data lies below and 50% lies above. It is less sensitive to outliers compared to the mean.
- **75% (3rd Quartile or Q3)**: The value below which 75% of the data falls.
    - It is the third quartile, which is the middle value between the median and the maximum value.
- **Maximum (max)**: The largest number.

7.

1) for example, a data set of --> name, age, favorite food, favorite drinks and income, in a situation where a lot of people enter data for the favorite food, favorite drink, but didn’t

entered their age and income , this is when you can consider removing the entire row of the person since their data is not full, In this case, dropping rows with missing data ensures you keep the useful information.

2) keep using the example from q1, If many people enter data for favorite food, age, and income but didn't provide their favorite drinks, you might want to consider deleting the entire drinks column. Since a lot of people didn't enter their favorite drink, the column becomes less useful, and removing it would help keep only the useful information in the dataset.

3) you delete that column first, you'll clean up your data by getting rid of a big chunk of missing values. It might also delete the missing data for a row which you don't have to delete the row again. Then, when you use df.dropna() after, you're only removing rows that still have missing data in the other useful columns. del df['col'] is more likely a larger scope compared to df.dropna() is more likely a smaller scope.

4) Start by deleting the column with the most missing data using `del df['col']`. This column might also be the least useful. After that, look for rows that are missing important data and remove them using `df.dropna()`. These are the cleaning steps. Before cleaning, you should record how many rows and columns are in your dataset. After cleaning, note how many rows and columns you've deleted and how many are left. Then, compare the quality or accuracy of the data before and after cleaning to see if there's any difference.

8.

1) For groupby("col1"), it will help you automatically separate your data set into sections, for example, in your column 1, you have titanic survivor of 15 male, 50 female, which means it will help you separate it into 2 sections, combining all each gender survivors together.

For ["col2"] part, for example I can enter an integer, and it will help you generate all the following for you(count, mean, min, max, etc). So using the function df.groupby("col1")["col2"].describe() gives you a organized table of your data set, in the case of titanic, ("col1") would be male and female into two section, and for ["col2"], we take the age of all survivors, it automatically calculates the count, mean, min, max...etc for you.

2) df.describe() shows the overall count of non-missing values in each column. df.groupby("col1")["col2"].describe() breaks it down by groups (in col1) and shows how many non-missing values there are for col2 in each group.

3)

A.

What happens:

- If you forget to import pandas, you'll get an error like NameError: name 'pd' is not defined. This error happens because Python doesn't know what "pd" is since pandas hasn't been imported.

How to troubleshoot:

- With the chatbot, you can explain the error, and it will likely suggest you forgot to import pandas.
- If you Google the error, you'll find similar solutions, telling you to include import pandas as pd.

B.

What happens:

- This will cause a `FileNotFoundError: [Errno 2] No such file or directory: 'titanics.csv'`. Python is trying to load a file that doesn't exist because of the typo.

How to troubleshoot:

- Chatbot: Explaining the error will likely prompt the suggestion to check if the file name is correct.
- Google: Searching for the error will also show similar advice to check the file path and name.

C.

What happens:

- If you mistype the variable (like using DF instead of `df`), you'll get an error like `NameError: name 'DF' is not defined`. Python doesn't recognize DF because the variable was defined as `df` (with lowercase letters).

How to troubleshoot:

- Chatbot: You can explain that you got the `NameError`, and it will likely suggest checking your variable names.
- Google: Searching for the `NameError` will show you that Python variables are case-sensitive, so you'll find advice on correcting the variable name.

D.

What happens:

- If you forget a parenthesis (e.g., `pd.read_csv(url` instead of `pd.read_csv(url)`), you'll get a syntax error like `SyntaxError: unexpected EOF while parsing`.

How to troubleshoot:

- Chatbot: Explaining the syntax error will prompt the suggestion to check your parentheses and code structure.
- Google: Searching for `SyntaxError` will show similar advice, likely reminding you to ensure all parentheses are properly closed.

E.

What happens:

- If you mistype a function name, you'll get an error like `AttributeError: 'DataFrame' object has no attribute 'group_by'`. Python can't find the function you're trying to call because it's mistyped.

How to troubleshoot:

- Chatbot: It will suggest checking the spelling of your functions.
- Google: Searching for `AttributeError` will show similar advice about checking function names and attributes.

F.

What happens:

- If you use a column name that isn't in your data (like replacing "sex" with "Sex"), you'll get a `KeyError` (e.g., `KeyError: 'Sex'`). This means Python couldn't find the column you're referencing.

How to troubleshoot:

- Chatbot: You'll be reminded to check if the column name is correct and matches exactly (case-sensitive).
- Google: Searching for `KeyError` will show similar advice, suggesting you check column names.

G.

- **What happens**:
  If you forget to put the column name in quotes, like `titanic_df.groupby(sex)`, you'll get an error like `NameError: name 'sex' is not defined`.

How to troubleshoot:

- Chatbot: It will suggest putting the column name in quotes, reminding you it needs to be treated as a string.
- Google: Searching for `NameError` will also lead you to similar advice, reminding you to use quotes for column names.

9. somewhat

Link to ***ChatBot session:*** https://chatgpt.com/share/e0986294-c73f-463c-8781-5c057affc34a