Week1

September 10, 2024

0.1 *Question 1*

```
[1]: import pandas as pd

     # Load the dataset
     url = 'https://web.stanford.edu/class/archive/cs/cs109/cs109.1166/stuff/titanic.
       ↪csv'
     titanic_data = pd.read_csv(url)

     # Check for missing values
     print(titanic_data.isnull().sum())
```

```
Survived                   0
Pclass                     0
Name                       0
Sex                        0
Age                        0
Siblings/Spouses Aboard    0
Parents/Children Aboard    0
Fare                       0
dtype: int64
```

ChatGPT Chatlog Summary

Here's a summary of this chat session:

1. **Dataset Requests**: You asked for links to datasets that can be imported into Python for analysis. I provided you with several datasets:
   - **Iris Dataset**: A classic dataset for flower classification.
   - **Titanic Dataset**: Data on Titanic passengers, which is useful for survival prediction.
   - **Wine Quality Dataset**: A dataset about wine characteristics, with no missing values.
   - **World Happiness Report**: Data on global happiness rankings.
2. **Missing Values**: You requested a dataset that contains missing values. Initially, I mistakenly provided links that didn't work, but I later corrected that with a working **Titanic Dataset** link that contains missing values in the **Age**, **Cabin**, and **Embarked** columns.
3. **Python Import Instructions**: For each dataset, I provided code snippets for importing the data into Python and checking for missing values using `pandas`. For instance, checking for missing values in the Titanic dataset can be done with:

```
import pandas as pd
url = 'https://web.stanford.edu/class/archive/cs/cs109/cs109.1166/stuff/titanic.csv'
titanic_data = pd.read_csv(url)
print(titanic_data.isnull().sum())
```

In summary, this session focused on finding datasets that you can use in Python for analysis, particularly focusing on datasets with missing values for data cleaning practice.

link: https://chatgpt.com/share/0506ae9e-1d60-43e2-a50d-bc282e2eada2

0.2 *Question 2*

```
[3]: import pandas as pd

     # Load the dataset
     df = pd.read_csv('https://web.stanford.edu/class/archive/cs/cs109/cs109.1166/
      ↪stuff/titanic.csv')
     # Get the number of rows and columns
     num_rows, num_columns = df.shape

     print(f'Number of rows: {num_rows}')
     print(f'Number of columns: {num_columns}')
```

```
Number of rows: 887
Number of columns: 8
```

Observation - In statistics, an observation is a single piece of qualitative or quantitative information about a given variable. A dataset is comprised of these observations. For example, in the Titanic dataset above, the female gender of a specific passenger would be a observation. These observations are often recorded in tables or graphs.

Variable - A variable is a measurable characteristic or attribute that varies between observations. The observations of a variable can be numerical such as height, or it can be categorical such as species. Variables allow you to compare and contrast the different attributes of different entities.

ChatGPT Chatlog Summary

Here's a summary of our chat:

1. **Dataset Information**: You asked about the Titanic dataset and the typical number of rows and columns. I explained that the dataset usually has 887 rows and 12 columns, detailing the column names.
2. **Observations in Statistics**: I clarified that in statistics, "observations" are individual data points or records in a dataset, represented by rows in a table, and are essential for analysis.
3. **Variables in Statistics**: I described that a variable is a characteristic or attribute that can vary between observations, such as age or gender. Variables can be categorical or numerical and are crucial for organizing and analyzing data.
4. **Code for Data Dimensions**: I provided Python code to determine the number of rows and columns in your dataset using pandas.

Feel free to use or modify this summary as needed for your homework!

Link: https://chatgpt.com/share/e32c1223-d3c4-43ca-9f87-1d0d1c88e510

0.3 *Question 3*

[4]:
```
import pandas as pd

# Load data
url = 'https://web.stanford.edu/class/archive/cs/cs109/cs109.1166/stuff/titanic.
 ↪csv'
titanic_data = pd.read_csv(url)

# Get statistical summary of numerical columns
summary = titanic_data.describe()
print(summary)
```

```
         Survived      Pclass         Age  Siblings/Spouses Aboard  \
count  887.000000  887.000000  887.000000               887.000000
mean     0.385569    2.305524   29.471443                 0.525366
std      0.487004    0.836662   14.121908                 1.104669
min      0.000000    1.000000    0.420000                 0.000000
25%      0.000000    2.000000   20.250000                 0.000000
50%      0.000000    3.000000   28.000000                 0.000000
75%      1.000000    3.000000   38.000000                 1.000000
max      1.000000    3.000000   80.000000                 8.000000

       Parents/Children Aboard       Fare
count               887.000000  887.00000
mean                  0.383315   32.30542
std                   0.807466   49.78204
min                   0.000000    0.00000
25%                   0.000000    7.92500
50%                   0.000000   14.45420
75%                   0.000000   31.13750
max                   6.000000  512.32920
```

[5]:
```
categorical_summary = titanic_data['Pclass'].value_counts()
print("Categorical Summary:")
print(categorical_summary)
```

```
Categorical Summary:
Pclass
3    487
1    216
2    184
Name: count, dtype: int64
```

ChatGPT Chat Summary

Here's a summary of our chat:

1. **Code Explanation:**
   - You provided a code snippet that loads a Titanic dataset from a URL into a Pandas DataFrame and prints the count of missing values for each column.
   - The `print(titanic_data.isnull().sum())` line shows the number of missing values per column in the dataset.
2. **Using `df.describe()`:**
   - `df.describe()` provides a statistical summary of numerical columns, including count, mean, standard deviation, min, 25th percentile, median, 75th percentile, and max.
3. **Using `df['column'].value_counts()`:**
   - `df['column'].value_counts()` provides the frequency of each unique value in a specific categorical column, helping understand the distribution of different categories.
4. **Combining Both Approaches:**
   - Use `df.describe()` for numerical columns to get a summary of their distribution.
   - Use `df['column'].value_counts()` for categorical columns to get the frequency distribution of unique values.

By using these methods, you can gain insights into both numerical and categorical data within your dataset.

link: https://chatgpt.com/share/aefe9fe9-b4d1-4427-86e1-4f6a78169548

0.4 *Question 4*

Using the df.shape function in pandas, the reported number of rows for the Titanic dataset is 887 while the reported number of collumns in the dataset is 7. However, using the df.describe() function only two collumns appear: Age and Fare. This is because the other collumns/variables contain non-numerical values which cannot be counted. The count value for Age is also less than the count value for Fare because there are missing data entries in the Age dataset.

[7]:
```python
import pandas as pd

# Load data
url = 'https://web.stanford.edu/class/archive/cs/cs109/cs109.1166/stuff/titanic.
 ↪csv'
titanic_data = pd.read_csv(url)

# Get the shape of the DataFrame
shape = titanic_data.shape
print(shape)
```

```
(887, 8)
```

[6]:
```python
import pandas as pd

# Load data
url = 'https://web.stanford.edu/class/archive/cs/cs109/cs109.1166/stuff/titanic.
 ↪csv'
```

```python
titanic_data = pd.read_csv(url)

# Get summary statistics for numerical columns
summary = titanic_data.describe()
print(summary)
```

```
         Survived      Pclass         Age  Siblings/Spouses Aboard  \
count  887.000000  887.000000  887.000000               887.000000   
mean     0.385569    2.305524   29.471443                 0.525366   
std      0.487004    0.836662   14.121908                 1.104669   
min      0.000000    1.000000    0.420000                 0.000000   
25%      0.000000    2.000000   20.250000                 0.000000   
50%      0.000000    3.000000   28.000000                 0.000000   
75%      1.000000    3.000000   38.000000                 1.000000   
max      1.000000    3.000000   80.000000                 8.000000   

       Parents/Children Aboard       Fare  
count               887.000000  887.00000  
mean                  0.383315   32.30542  
std                   0.807466   49.78204  
min                   0.000000    0.00000  
25%                   0.000000    7.92500  
50%                   0.000000   14.45420  
75%                   0.000000   31.13750  
max                   6.000000  512.32920  
```

ChatGPT chatlog

Here's a summary of our chat:

1. **Code Explanation:**
   - Your code snippet loads the Titanic dataset from a URL into a Pandas DataFrame and prints the count of missing values for each column.
2. **Using `df.describe()`:**
   - `df.describe()` provides a statistical summary of numerical columns in the DataFrame, including count, mean, standard deviation, minimum, percentiles, and maximum values.
3. **Using `df['column'].value_counts()`:**
   - `df['column'].value_counts()` shows the frequency of each unique value in a specific categorical column.
4. **Combining Both Approaches:**
   - Use `df.describe()` for numerical columns to understand their distribution.
   - Use `df['column'].value_counts()` for categorical columns to understand the distribution of categories.
5. **`df.shape` Method:**
   - `df.shape` provides the dimensions of the DataFrame, returning a tuple with the number of rows and columns.
6. **`df.describe()` Details:**
   - `df.describe()` includes count, mean, standard deviation, minimum, percentiles, and maximum values for numerical columns. You can use `df.describe(include='all')` to

include categorical columns as well.

link: https://chatgpt.com/share/aefe9fe9-b4d1-4427-86e1-4f6a78169548

0.5 *Question 5*

In pandas, **attributes** usually display some feature of the dataset. For example, the df.shape function shows us how big the particular dataset is. Attributes are called without parenthesis. On the other hand, methods usually perform some action on the dataset or manipulate the dataset in some way. For example, the df.describe() function calculates the mean and variance of variables. Methods are called with parenthesis.

ChatGPT Chat Log Sure! Here's a summary of our chat:

- **Difference Between Attribute and Method in Pandas**:
  - **Attribute**: Provides information about a pandas object (e.g., `df.shape` gives the dimensions of a DataFrame) and is accessed without parenthesis.
  - **Method**: Performs actions or operations on a pandas object (e.g., `df.head()` returns the first few rows) and is called with parenthesis.

Link: https://chatgpt.com/share/82d70231-3d87-4292-91ba-cf68403d2811

0.6 *Question 6*

In the summary statistics of the df.describe() method, the **count** of a column refers to the number of non-null entries in that particular column. The **mean** of a column refers to the mathematical average for all entries into that column. It can be calculated by summing up all entries in the column and then dividing that sum by the count of the column. The **std** of a column refers to the standard deviation of the column, or how spread out the data is. A higher **std** implies that the data is more spread out. The **min** of a column refers to the value of the smallest data entry in that column and likewise, the **max** of a column refers to the value of the biggest data entry in that column. **25%** refers to the first quartile value. This means that exactly 25% of the entries in the data set are smaller than the first 25% value. Similairly exactly 50% of the data set is smaller than the **50%** value and exactly 75% of the data set is smaller than the **75%** value.

0.7 Question 7

1. Using df.dropna() would be preferred over using del df['col'] when we want to know all the attributes of complete data entries. For example, if there was a class survey and we wanted to know the name, age, and sex of all students who fully completed the class survey we would use df.dropna() instead of del df['col'].

2. Using del df['col'] would be preferred over using df.dropna() if we want to know only certain attributes about the data entries. For example, if we only wanted to know the names of the students who did the survey we would use del df['age'] and del df['sex'].

3. del df['col'] is more time efficient than df.dropna() because it does not require the computer to look through every single row and collumn for missing values. If we know that we already

don't need a column, we can delete it first so that the computer doesn't search through that column's values when we are deleting the necessary rows.

4.

```
[8]: import pandas as pd
     url = "https://raw.githubusercontent.com/KeithGalli/pandas/master/pokemon_data.
      ↪csv"
     data = pd.read_csv(url)
     del data['Type 2']
     print(data.isnull().sum())
```

```
#             0
Name          0
Type 1        0
HP            0
Attack        0
Defense       0
Sp. Atk       0
Sp. Def       0
Speed         0
Generation    0
Legendary     0
dtype: int64
```

ChatGPT Chat Logs

In our conversation, you initially asked about the function of `del df['col']` in pandas. I explained that this command deletes the specified column from a DataFrame, regardless of whether it contains missing data. An example demonstrated how this removes a column entirely from the DataFrame.

Next, you inquired about `df.dropna()`, which removes rows or columns with `NaN` values. By default, it drops rows containing any missing data, but parameters like `axis` and `how` allow for more control, such as dropping columns or handling rows where all values are missing.

You then asked if `del df['name']` would remove columns with incomplete data entries, and I clarified that it does not. Instead, to remove columns with missing data, you would use `df.dropna(axis=1)`. This method ensures columns with `NaN` values are specifically targeted, unlike the manual deletion with `del`.

link: https://chatgpt.com/share/a874f911-b940-4a85-84ef-72f0a644e0e2

0.8 *Question 8*

1. First, the groupby('col1') method groups all rows with the same 'col1' value together in the pokemon dataset. Second, the ['col2'] attribute selects the second column of all data entries in all the groups created above, and performs a describe() method where it outputs the mean, count, std, max, min, 25%, 50%, and 75% of the collumn 2s of each group.

[14]:

```python
titanic = pd.read_csv("https://raw.githubusercontent.com/KeithGalli/pandas/
 ↪master/pokemon_data.csv")
grouped = titanic.groupby('Type 1')
grouped['Attack'].describe()
```

[14]:

```
          count        mean        std   min    25%    50%     75%    max
Type 1
Bug        69.0   70.971014  37.040904  10.0  45.00   65.0   90.00  185.0
Dark       31.0   88.387097  25.774247  50.0  65.00   88.0  100.00  150.0
Dragon     32.0  112.125000  33.742622  50.0  86.25  113.5  134.25  180.0
Electric   44.0   69.090909  23.764169  30.0  53.75   65.0   85.00  123.0
Fairy      17.0   61.529412  29.751298  20.0  45.00   52.0   72.00  131.0
Fighting   27.0   96.777778  28.290163  35.0  80.00  100.0  120.00  145.0
Fire       52.0   84.769231  28.769275  30.0  62.25   84.5  101.00  160.0
Flying      4.0   78.750000  37.500000  30.0  60.00   85.0  103.75  115.0
Ghost      32.0   73.781250  29.629687  30.0  53.75   66.0   92.75  165.0
Grass      70.0   73.214286  25.380520  27.0  55.00   70.0   93.50  132.0
Ground     32.0   95.750000  33.059087  40.0  72.00   85.0  121.00  180.0
Ice        24.0   72.750000  27.289511  30.0  50.00   67.0   87.50  130.0
Normal     98.0   73.469388  30.295862   5.0  55.00   70.5   85.00  160.0
Poison     28.0   74.678571  19.630010  43.0  60.00   74.0   90.50  106.0
Psychic    57.0   71.456140  42.309265  20.0  45.00   57.0   95.00  190.0
Rock       44.0   92.863636  35.325458  40.0  59.75   95.0  120.25  165.0
Steel      27.0   92.703704  30.388276  24.0  77.50   89.0  110.00  150.0
Water     112.0   74.151786  28.377192  10.0  53.00   72.0   92.00  155.0
```

2. df.describe() displays the overall number of non-null values in each collumn, but df.groupby('col1')['col2'].describe() tells you the count of non-null values of col2 in each group of rows with the same 'col1' value.
   /
3. I believe that a google search is more efficient for fixing errors which are generic such as errors A and C, but ChatGPT is a better way to fix errors that are more specific to your code such as errors E and G. This is because Google will usually give a short, straight to the point answer while ChatGPT outputs a long winded detailed answer to your question. If the error is generic Google will solve my issue quicker, but if my issue is very specific to my code then I need a more detailed fix.

0.9 *Question 9*

Yes, I have.