STA 130 W1 HW1

September 23, 2024

[5]:
```
import pandas as pd

# URL for the dataset
url = "https://raw.githubusercontent.com/rfordatascience/tidytuesday/master/
 ↪data/2020/2020-05-05/villagers.csv"

# Load the dataset into a pandas DataFrame
df = pd.read_csv(url)

# Check for missing values
missing_values = df.isna().sum()

# Output the missing values
missing_values
```

[5]:
```
row_n           0
id              1
name            0
gender          0
species         0
birthday        0
personality     0
song           11
phrase          0
full_id         0
url             0
dtype: int64
```

[6]:
```
# Get the number of rows and columns in the DataFrame
rows, columns = df.shape

# Print out the result
f"The dataset has {rows} rows (observations) and {columns} columns (variables)."
```

[6]:
```
'The dataset has 391 rows (observations) and 11 columns (variables).'
```

[ ]:
```
Observations: Each row in the dataset is called an observation. It represents a␣
 ↪single entry or record. For example, in the villagers dataset, each row␣
 ↪represents one character from the game.
Variables: Each column in the dataset is a variable. It represents a␣
 ↪characteristic or attribute of the observations, such as the name, species,␣
 ↪personality, or birthday of a character.
```

[8]:
```
# Summary of numeric columns
df.describe()
```

[8]:
```
            row_n
count  391.000000
mean   239.902813
std    140.702672
min      2.000000
25%    117.500000
50%    240.000000
75%    363.500000
max    483.000000
```

[7]:
```
# Count the unique values in the 'species' column
df['species'].value_counts()
```

[7]:
```
species
cat          23
rabbit       20
frog         18
squirrel     18
duck         17
dog          16
cub          16
pig          15
bear         15
mouse        15
horse        15
bird         13
penguin      13
sheep        13
elephant     11
wolf         11
ostrich      10
deer         10
eagle         9
gorilla       9
chicken       9
koala         9
goat          8
```

```
hamster       8
kangaroo      8
monkey        8
anteater      7
hippo         7
tiger         7
alligator     7
lion          7
bull          6
rhino         6
cow           4
octopus       3
Name: count, dtype: int64
```

[9]:
```
import pandas as pd

# Load the Titanic dataset
url = "https://raw.githubusercontent.com/mwaskom/seaborn-data/master/titanic.
 ↪csv"
df_titanic = pd.read_csv(url)

# Get a summary of numeric columns
summary_numeric = df_titanic.describe()

print(summary_numeric)
```

```
         survived      pclass         age       sibsp       parch        fare
count  891.000000  891.000000  714.000000  891.000000  891.000000  891.000000
mean     0.383838    2.308642   29.699118    0.523008    0.381594   32.204208
std      0.486592    0.836071   14.526497    1.102743    0.806057   49.693429
min      0.000000    1.000000    0.420000    0.000000    0.000000    0.000000
25%      0.000000    2.000000   20.125000    0.000000    0.000000    7.910400
50%      0.000000    3.000000   28.000000    0.000000    0.000000   14.454200
75%      1.000000    3.000000   38.000000    1.000000    0.000000   31.000000
max      1.000000    3.000000   80.000000    8.000000    6.000000  512.329200
```

[11]:
```
# Check for missing values in the Titanic dataset
df_titanic.isna().sum()
```

[11]:
```
survived      0
pclass        0
sex           0
age         177
sibsp         0
parch         0
fare          0
embarked      2
```

```
class            0
who              0
adult_male       0
deck           688
embark_town      2
alive            0
alone            0
dtype: int64
```

```
[ ]: In programming terms, attributes are more about accessing data, while methods␣
     ↪are about executing functions or operations. This distinction helps in␣
     ↪understanding how to interact with objects in a programming language.
```

Question 6

Count:

Definition: The number of non-null (non-missing) values in each column. Purpose: Helps understand how many valid data points are available for analysis in each column. Mean:

Definition: The average value of the column, calculated as the sum of all values divided by the count of values. Purpose: Provides a measure of central tendency, showing where the center of the data lies. Standard Deviation (std):

Definition: A measure of the amount of variation or dispersion in the column. It indicates how much the values deviate from the mean on average. Purpose: Helps understand the spread of the data. A larger standard deviation means more spread out values. Minimum (min):

Definition: The smallest value in the column. Purpose: Provides the lower bound of the data range, helping identify the lowest data point. 25th Percentile (25%):

Definition: The value below which 25% of the data falls. It's also known as the first quartile (Q1). Purpose: Helps understand the distribution of the data by dividing the dataset into quartiles. 50th Percentile (50%):

Definition: The median value of the column, which is the middle value when the data is sorted. It's also the second quartile (Q2). Purpose: Provides the central value of the dataset, indicating where half of the data points fall below and above. 75th Percentile (75%):

Definition: The value below which 75% of the data falls. It's also known as the third quartile (Q3). Purpose: Helps understand the upper range of the data distribution by dividing the dataset into quartiles. Maximum (max):

Definition: The largest value in the column. Purpose: Provides the upper bound of the data range, helping identify the highest data point.

Question 7

1. Use Case for df.dropna() Example Use Case: Scenario: You have a dataset with customer information for a retail analysis. Some rows have missing values for non-critical fields such as "middle_name" or "nickname," but the rest of the data is complete and crucial for analysis.

Approach: Use df.dropna() to remove rows with missing values if those fields are not critical to the analysis.

Justification: By using df.dropna(), you remove only those rows where the important data is missing, without affecting columns that are crucial for your analysis. This approach is preferred when missing values are scattered across rows and you want to retain as much of the remaining data as possible.

2. Use Case for del df[‘col’] Example Use Case: Scenario: You have a dataset with multiple columns, and one column (e.g., “optional_notes”) contains a significant number of missing values. This column is not important for your analysis and its missing data makes it less reliable.

Approach: Use del df[‘col’] to remove the entire column with missing values.

Justification: Removing the column is preferred when the column does not contribute to the analysis or is deemed irrelevant. This avoids any confusion or extra processing for columns that have too many missing values.

3. Importance of Applying del df[‘col’] Before df.dropna() Reason: Applying del df[‘col’] before df.dropna() can be important if the column with missing data is not relevant to your analysis.

Benefit: This sequence helps avoid wasting computational resources and ensures that missing value handling is applied only to relevant data.

[13]:
```
import pandas as pd

# Load the Titanic dataset
url = "https://raw.githubusercontent.com/mwaskom/seaborn-data/master/titanic.
 ↪csv"
df_titanic = pd.read_csv(url)

# Check the number of missing values before removing
missing_before = df_titanic.isna().sum()
print("Missing values before removing:")
print(missing_before)

# Check the shape of the dataset before removing
shape_before = df_titanic.shape
print("\nShape before removing missing data:")
print(shape_before)

# Remove rows with missing values
df_titanic_cleaned = df_titanic.dropna()

# Check the number of missing values after removing
missing_after = df_titanic_cleaned.isna().sum()
print("\nMissing values after removing:")
print(missing_after)
```

```python
# Check the shape of the dataset after removing
shape_after = df_titanic_cleaned.shape
print("\nShape after removing missing data:")
print(shape_after)
```

```
Missing values before removing:
survived         0
pclass           0
sex              0
age            177
sibsp            0
parch            0
fare             0
embarked         2
class            0
who              0
adult_male       0
deck           688
embark_town      2
alive            0
alone            0
dtype: int64

Shape before removing missing data:
(891, 15)

Missing values after removing:
survived       0
pclass         0
sex            0
age            0
sibsp          0
parch          0
fare           0
embarked       0
class          0
who            0
adult_male     0
deck           0
embark_town    0
alive          0
alone          0
dtype: int64

Shape after removing missing data:
(182, 15)
```

8.2 titanic_df.groupby("class") groups the data by the 'class' column (e.g., First, Second, Third).

["age"].describe() provides summary statistics for the 'age' column within each class group.

8.3 Troubleshooting Common Errors

a. Missing import pandas as pd

Error: NameError: name 'pd' is not defined Solution: Add import pandas as pd at the top of your script. b. Mistyping the file name

Error: FileNotFoundError: [Errno 2] No such file or directory: 'titanics.csv' Solution: Correct the file name to 'titanic.csv'. c. Using an undefined DataFrame variable

Error: NameError: name 'DF' is not defined Solution: Ensure you use the correct variable name, e.g., df. d. Forgetting parentheses

Error: TypeError: pd.read_csv missing 1 required positional argument: 'filepath_or_buffer' Solution: Add the missing parenthesis. e. Mistyping method names

Errors: AttributeError: 'DataFrame' object has no attribute 'group_by' AttributeError: 'DataFrame' object has no attribute 'describle' Solution: Correct method names to groupby and describe. f. Using incorrect column names

Errors: KeyError: 'Sex' or KeyError: 'age' Solution: Use the correct column names as they appear in the DataFrame. g. Forgetting quotes around column names

Errors: NameError: name 'sex' is not defined NameError: name 'age' is not defined Solution: Enclose column names in quotes.

[ ]:
```
ChatBot vs. Google for Error Troubleshooting
ChatBot: It can be helpful for interactive troubleshooting, understanding␣
 ↪errors, and explaining concepts. However, its responses depend on the␣
 ↪clarity of the question and might sometimes be limited in scope.

Google Search: Often faster for finding specific error messages and solutions.␣
 ↪It can provide various forums, documentation, and examples that might solve␣
 ↪the issue quickly.
```

9. YES

Chat GPT chat log histories:https://chatgpt.com/share/5d78920f-57dc-41c9-9d27-18da024df5f8 and:https://chatgpt.com/share/0e7db697-62fa-4c05-9075-381d1fe6fca3