assignment-3

February 19, 2024

Q1) Perform the following operations on Age-Income dataset (Age- Income-Dataset.csv)

Provide summary statistics (mean, median, minimum, maximum, standard deviation) for numeric variables with and without using any library functions. Provide summary statistics of income grouped by the age groups. Create a list that contains a numeric value for each response to the categorical variable.

[2]:
```
import numpy as np
import pandas as pd

df = pd.read_excel("/content/Age-Income-Dataset.xlsx")
df.head()
```

[2]:
```
           Age  Income
0        Young   25000
1   Middle Age   54000
2          Old   60000
3        Young   15000
4        Young   45000
```

[3]:
```
df.isnull().sum()
```

[3]:
```
Age       0
Income    0
dtype: int64
```

[4]:
```
df['Age'].unique()
```

[4]:
```
array(['Young', 'Middle Age', 'Old'], dtype=object)
```

0.0.1 Calculating Measures of Central Tendancy

[5]:
```
# Calcualting mean by formula
mean_score = sum(df['Income'])/len(df['Income'])
print(mean_score)
```

```
50966.0
```

[6]:

```
#Using Pandas function
mean_score = df['Income'].mean()
print(mean_score)
```

```
50966.0
```

Therefore from given data the mean Income is Rs 50966.0

[7]:

```
# Calculating Median by formula
n = len(df['Income'])
if n % 2:
    income_median = sorted(df['Income'])[round(0.5*(n-1))]
else:
    x_ord, index = sorted(df['Income']), round(0.5 * n)
    income_median = 0.5 * (x_ord[index-1] + x_ord[index])

print(income_median)
```

```
46850.0
```

[8]:

```
# Using Pandas function
df['Income'].median()
```

[8]:

```
46850.0
```

The median is simply the middle value of the sorted dataset.The value 46850.0 splits the dataset in half.

[9]:

```
# Finding the mode
df['Income'].mode()
```

[9]:

```
0    23000
1    25600
2    45000
3    65400
4    80000
Name: Income, dtype: int64
```

The above are the values that appears most frequently in the dataset.

[12]:

```
income_grouped_by_age = df.groupby('Age')['Income'].describe()
print(income_grouped_by_age)
```

```
            count          mean           std      min      25%      50%  \
Age
Middle Age   15.0  52453.333333  20497.800114  25600.0  36900.0  53200.0
Old          19.0  53942.105263  20868.165968  24500.0  38700.0  45300.0
Young        16.0  46037.500000  22356.859499  15000.0  28750.0  41500.0
```

```
                75%      max
Age
Middle Age  61200.0  93000.0
Old         71400.0  89700.0
Young       65850.0  87000.0
```

Above is the summary statistics of income of people which are grouped by age groups.

```
[14]: age_numeric_values = {'Young': 25, 'Middle Age': 45, 'Old': 65}
      # Create a list with numeric values for each response to the categorical␣
       ↪variable
      numeric_for_age = [age_numeric_values[age] for age in df['Age']]

      print("\nNumeric Values for Categorical Variable 'Age':")
      print(numeric_for_age)
```

```

Numeric Values for Categorical Variable 'Age':
[25, 45, 65, 25, 25, 25, 25, 25, 45, 25, 25, 65, 25, 65, 65, 65, 45, 45, 65, 45,
65, 65, 65, 65, 45, 45, 25, 25, 25, 25, 45, 45, 65, 45, 45, 65, 65, 65, 25, 65,
45, 65, 25, 45, 65, 65, 45, 65, 25, 45]
```

Numeric_for_age contains a numeric values for each response to the categorical variable(Age).

Q2) Write a Python program to display some basic statistical details

like percentile, mean, standard deviation etc. of the species of ‘Iris- setosa’, ‘Iris-versicolor’ and ‘Iris-virginica’ of iris.csv dataset.

Calculate the measures of variability. Calculate and provide the visualization of the Correlation among the variables.

```
[15]: iris_df = pd.read_csv('/content/Iris.csv')
      iris_df.head()
```

```
[15]:    Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm      Species
      0   1            5.1           3.5            1.4           0.2  Iris-setosa
      1   2            4.9           3.0            1.4           0.2  Iris-setosa
      2   3            4.7           3.2            1.3           0.2  Iris-setosa
      3   4            4.6           3.1            1.5           0.2  Iris-setosa
      4   5            5.0           3.6            1.4           0.2  Iris-setosa
```

```
[17]: iris_df.isnull().sum()
```

```
[17]: Id               0
      SepalLengthCm    0
      SepalWidthCm     0
      PetalLengthCm    0
      PetalWidthCm     0
      Species          0
```

```
dtype: int64
```

0.0.2 Filter data for each species

```
[18]: setosa_data = iris_df[iris_df['Species'] == 'Iris-setosa']
      versicolor_data = iris_df[iris_df['Species'] == 'Iris-versicolor']
      virginica_data = iris_df[iris_df['Species'] == 'Iris-virginica']
```

```
[19]: virginica_data.head()
```

```
[19]:        Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm  \
      100  101            6.3           3.3            6.0           2.5
      101  102            5.8           2.7            5.1           1.9
      102  103            7.1           3.0            5.9           2.1
      103  104            6.3           2.9            5.6           1.8
      104  105            6.5           3.0            5.8           2.2

                  Species
      100  Iris-virginica
      101  Iris-virginica
      102  Iris-virginica
      103  Iris-virginica
      104  Iris-virginica
```

```
[20]: versicolor_data.head()
```

```
[20]:     Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm  \
      50  51            7.0           3.2            4.7           1.4
      51  52            6.4           3.2            4.5           1.5
      52  53            6.9           3.1            4.9           1.5
      53  54            5.5           2.3            4.0           1.3
      54  55            6.5           2.8            4.6           1.5

                  Species
      50  Iris-versicolor
      51  Iris-versicolor
      52  Iris-versicolor
      53  Iris-versicolor
      54  Iris-versicolor
```

```
[21]: setosa_data.head()
```

```
[21]:    Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm      Species
      0   1            5.1           3.5            1.4           0.2  Iris-setosa
      1   2            4.9           3.0            1.4           0.2  Iris-setosa
      2   3            4.7           3.2            1.3           0.2  Iris-setosa
      3   4            4.6           3.1            1.5           0.2  Iris-setosa
```

```
4   5             5.0            3.6            1.4            0.2  Iris-setosa
```

[22]:

```python
setosa_data.mean()
```

```
<ipython-input-22-4264295ec158>:1: FutureWarning: The default value of
numeric_only in DataFrame.mean is deprecated. In a future version, it will
default to False. In addition, specifying 'numeric_only=None' is deprecated.
Select only valid columns or specify the value of numeric_only to silence this
warning.
  setosa_data.mean()
```

[22]:

```
Id                 25.500
SepalLengthCm       5.006
SepalWidthCm        3.418
PetalLengthCm       1.464
PetalWidthCm        0.244
dtype: float64
```

[23]:

```python
versicolor_data.median()
```

```
<ipython-input-23-c4adffb41e64>:1: FutureWarning: The default value of
numeric_only in DataFrame.median is deprecated. In a future version, it will
default to False. In addition, specifying 'numeric_only=None' is deprecated.
Select only valid columns or specify the value of numeric_only to silence this
warning.
  versicolor_data.median()
```

[23]:

```
Id                 75.50
SepalLengthCm       5.90
SepalWidthCm        2.80
PetalLengthCm       4.35
PetalWidthCm        1.30
dtype: float64
```

[29]:

```python
from scipy.stats import percentileofscore

percentileofscore(virginica_data['SepalLengthCm'], 6.3)
```

[29]:

```
33.0
```

The percentile of value 6.3 in virginica_data[‘SepalLength’] is 33.0

0.0.3 Measures of variability

Measures of variability are capable of quantifying the spread of data points.

[32]:

```python
print("Variance for Iris-setosa SepalWidthCm:")
print(setosa_data['SepalWidthCm'].var())
```

```
Variance for Iris-setosa SepalWidthCm:
0.1451795918367347
```

The variance quantifies the spread of the data. It signifies how far are the data points from the mean.

[33]:

```
print("Standard Deviation for versicolor_data PetalLengthCm:")
print(versicolor_data['PetalLengthCm'].std())
```

```
Standard Deviation for versicolor_data PetalLengthCm:
0.46991097723995795
```

Standard deviation is the positive square root of the sample variance. Here low standard deviation for a PetalLengthCm indicates that the data points tend to be close to its mean

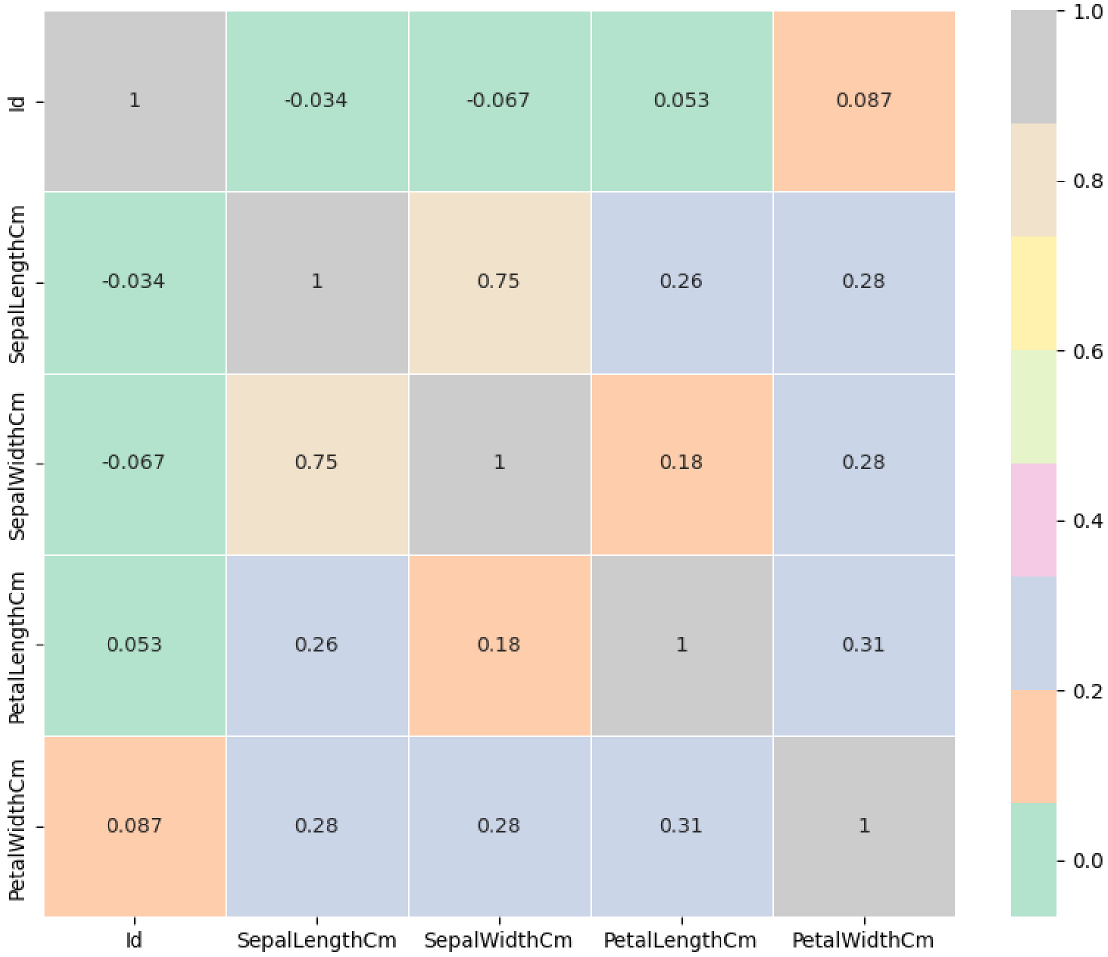
0.0.4 Visualization of Correlation

[42]:

```
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(10, 8))
correlation_matrix = setosa_data.corr()
sns.heatmap(correlation_matrix, annot=True, cmap="Pastel2", linewidths=.5)
plt.title("Correlation Matrix of Setosa Data")
plt.show()
```

```
<ipython-input-42-b7dcdef79e65>:5: FutureWarning: The default value of
numeric_only in DataFrame.corr is deprecated. In a future version, it will
default to False. Select only valid columns or specify the value of numeric_only
to silence this warning.
  correlation_matrix = setosa_data.corr()
```



Correlation Matrix of Setosa Data