DSBDA Lab Assignment No. 3

Name: Akash Ganesh Padir
Roll No.: TEB04

Assignment 3.1

In [1]:
```
import pandas as pd
import numpy as np
```

In [3]:
```
dataset= pd.read_csv("Mall_Customers.csv")
```

In [4]:
```
dataset
```

Out[4]:

	CustomerID	Genre	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40
...	...	...	...	...	...
195	196	Female	35	120	79
196	197	Female	45	126	28
197	198	Male	32	126	74
198	199	Male	32	137	18
199	200	Male	30	137	83

200 rows × 5 columns

In [5]:
```
dataset.shape
```

Out[5]:
```
(200, 5)
```

In [6]:
```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 5 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   CustomerID              200 non-null    int64
 1   Genre                   200 non-null    object
 2   Age                     200 non-null    int64
 3   Annual Income (k$)      200 non-null    int64
 4   Spending Score (1-100)  200 non-null    int64
dtypes: int64(4), object(1)
memory usage: 7.9+ KB
```

Measures of Central Tendency

1. Mean

In [7]:
```
dataset.mean()
```

```
C:\Users\COMP 549\AppData\Local\Temp\ipykernel_9880\1799472221.py:1: FutureWa
rning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_on
ly=None') is deprecated; in a future version this will raise TypeError.  Sele
ct only valid columns before calling the reduction.
  dataset.mean()
```

Out[7]:
```
CustomerID                100.50
Age                        38.85
Annual Income (k$)         60.56
Spending Score (1-100)     50.20
dtype: float64
```

In [8]:
```
dataset.loc[:,'Age'].mean()
```

Out[8]:
```
38.85
```

In [9]:
```
dataset.loc[:,'Annual Income (k$)'].mean()
```

Out[9]:
```
60.56
```

Calculate mean of rows

In [10]:
```
dataset.mean(axis=1)[0:5]
```

```
C:\Users\COMP 549\AppData\Local\Temp\ipykernel_9880\3443625040.py:1: FutureWa
rning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_on
ly=None') is deprecated; in a future version this will raise TypeError.  Sele
ct only valid columns before calling the reduction.
  dataset.mean(axis=1)[0:5]
```

Out[10]:
```
0    18.50
1    29.75
2    11.25
3    30.00
4    23.25
dtype: float64
```

2. Median

Median represents the 50th percentile or the middle value

In [11]:
```
dataset.median()
```

```
C:\Users\COMP 549\AppData\Local\Temp\ipykernel_9880\4167803218.py:1: FutureWa
rning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_on
ly=None') is deprecated; in a future version this will raise TypeError.  Sele
ct only valid columns before calling the reduction.
  dataset.median()
```

Out[11]:
```
CustomerID               100.5
Age                       36.0
Annual Income (k$)        61.5
Spending Score (1-100)    50.0
dtype: float64
```

Median of particular variable

In [12]:
```
dataset.loc[:,'Age'].median()
```

Out[12]:
```
36.0
```

3.Mode

Mode represents the most recently accessed

In [14]:
```
dataset.mode(axis=0)
```

Out[14]:

	CustomerID	Genre	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Female	32.0	54.0	42.0
1	2	NaN	NaN	78.0	NaN
2	3	NaN	NaN	NaN	NaN
3	4	NaN	NaN	NaN	NaN
4	5	NaN	NaN	NaN	NaN
...	...	...	...	...	...
195	196	NaN	NaN	NaN	NaN
196	197	NaN	NaN	NaN	NaN
197	198	NaN	NaN	NaN	NaN
198	199	NaN	NaN	NaN	NaN
199	200	NaN	NaN	NaN	NaN

200 rows × 5 columns

In [15]:
```
dataset.mode()
```

Out[15]:

	CustomerID	Genre	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Female	32.0	54.0	42.0
1	2	NaN	NaN	78.0	NaN
2	3	NaN	NaN	NaN	NaN
3	4	NaN	NaN	NaN	NaN
4	5	NaN	NaN	NaN	NaN
...	...	...	...	...	...
195	196	NaN	NaN	NaN	NaN
196	197	NaN	NaN	NaN	NaN
197	198	NaN	NaN	NaN	NaN
198	199	NaN	NaN	NaN	NaN
199	200	NaN	NaN	NaN	NaN

200 rows × 5 columns

Measures of Dispersion (or Variability)

1. Standard Deviation

In [16]:
```
dataset.loc[:,'Age'].std()
```

Out[16]:
```
13.969007331558883
```

Standard deviation of first 5 rows

In [17]:
```
dataset.std(axis=1)[0:5]
```

```
C:\Users\COMP 549\AppData\Local\Temp\ipykernel_9880\2666469271.py:1: FutureWa
rning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_on
ly=None') is deprecated; in a future version this will raise TypeError.  Sele
ct only valid columns before calling the reduction.
  dataset.std(axis=1)[0:5]
```

Out[17]:
```
0    15.695010
1    35.074920
2     8.057088
3    32.300671
4    15.413738
dtype: float64
```

2. Variance

In [18]:
```
dataset.var()
```

```
C:\Users\COMP 549\AppData\Local\Temp\ipykernel_9880\2458428038.py:1: FutureWa
rning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_on
ly=None') is deprecated; in a future version this will raise TypeError.  Sele
ct only valid columns before calling the reduction.
  dataset.var()
```

Out[18]:
```
CustomerID               3350.000000
Age                       195.133166
Annual Income (k$)        689.835578
Spending Score (1-100)    666.854271
dtype: float64
```

In [19]:
```
from scipy.stats import iqr
```

In [20]:
```
iqr(dataset['Age'])
```

Out[20]:
```
20.25
```

Skewness

In [21]:
```
dataset.skew()
```

```
C:\Users\COMP 549\AppData\Local\Temp\ipykernel_9880\4231230252.py:1: FutureWa
rning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_on
ly=None') is deprecated; in a future version this will raise TypeError.  Sele
ct only valid columns before calling the reduction.
  dataset.skew()
```

Out[21]:
```
CustomerID              0.000000
Age                     0.485569
Annual Income (k$)      0.321843
Spending Score (1-100)  -0.047220
dtype: float64
```

Describe all the statistics

In [23]:
```
dataset.describe()
```

Out[23]:

	CustomerID	Age	Annual Income (k$)	Spending Score (1-100)
count	200.000000	200.000000	200.000000	200.000000
mean	100.500000	38.850000	60.560000	50.200000
std	57.879185	13.969007	26.264721	25.823522
min	1.000000	18.000000	15.000000	1.000000
25%	50.750000	28.750000	41.500000	34.750000
50%	100.500000	36.000000	61.500000	50.000000
75%	150.250000	49.000000	78.000000	73.000000
max	200.000000	70.000000	137.000000	99.000000

In [24]:
```
dataset.describe(include='all')
```

Out[24]:

	CustomerID	Genre	Age	Annual Income (k$)	Spending Score (1-100)
count	200.000000	200	200.000000	200.000000	200.000000
unique	NaN	2	NaN	NaN	NaN
top	NaN	Female	NaN	NaN	NaN
freq	NaN	112	NaN	NaN	NaN
mean	100.500000	NaN	38.850000	60.560000	50.200000
std	57.879185	NaN	13.969007	26.264721	25.823522
min	1.000000	NaN	18.000000	15.000000	1.000000
25%	50.750000	NaN	28.750000	41.500000	34.750000
50%	100.500000	NaN	36.000000	61.500000	50.000000
75%	150.250000	NaN	49.000000	78.000000	73.000000
max	200.000000	NaN	70.000000	137.000000	99.000000

Prepare groupby

In [25]:
```
grouped= dataset.groupby('Age')
```

In [26]:
```
grouped
```

Out[26]:
```
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x00000189842582E0>
```

In [27]:
```
grouped.groups
```

Out[27]:
```
{18: [33, 65, 91, 114], 19: [0, 61, 68, 111, 113, 115, 138, 162], 20: [2, 17,
39, 99, 134], 21: [1, 31, 35, 84, 105], 22: [5, 15, 87], 23: [3, 7, 29, 78, 1
00, 124], 24: [13, 41, 45, 95], 25: [21, 132, 144], 26: [75, 103], 27: [47, 5
8, 97, 120, 155, 177], 28: [142, 145, 171, 187], 29: [25, 48, 135, 161, 183],
30: [9, 37, 157, 159, 175, 185, 199], 31: [4, 23, 43, 49, 52, 125, 133, 163],
32: [69, 94, 137, 141, 143, 147, 169, 181, 191, 197, 198], 33: [51, 167, 19
2], 34: [88, 148, 149, 158, 190], 35: [6, 11, 16, 19, 20, 27, 139, 179, 195],
36: [38, 165, 168, 172, 173, 189], 37: [14, 156, 180], 38: [81, 112, 121, 12
9, 153, 193], 39: [123, 131, 151], 40: [28, 77, 93, 122, 127, 170], 41: [184,
188], 42: [36, 166], 43: [66, 126, 150], 44: [136, 152], 45: [26, 76, 196], 4
6: [22, 83, 182], 47: [55, 71, 96, 130, 154, 194], 48: [42, 85, 92, 98, 146],
49: [34, 44, 50, 79, 101, 104, 117], 50: [46, 54, 89, 119, 164], 51: [56, 11
8], 52: [18, 174], 53: [32, 59], 54: [24, 63, 107, 186], 55: [86], 56: [160],
57: [80, 140], 58: [12, 176], 59: [53, 74, 128, 178], 60: [30, 72, 73], 63:
[64, 116], 64: [8], 65: [40, 110], 66: [106, 109], 67: [10, 62, 82, 102], 68:
[67, 90, 108], 69: [57], 70: [60, 70]}
```

In [28]:
```
grouped.size()
```

Out[28]:
```
Age
18     4
19     8
20     5
21     5
22     3
23     6
24     4
25     3
26     2
27     6
28     4
29     5
30     7
31     8
32    11
33     3
34     5
35     9
36     6
37     3
38     6
39     3
40     6
41     2
42     2
43     3
44     2
45     3
46     3
47     6
48     5
49     7
50     5
51     2
52     2
53     2
54     4
55     1
56     1
57     2
58     2
59     4
60     3
63     2
64     1
65     2
66     2
67     4
68     3
69     1
70     2
dtype: int64
```

In [29]:
```
grouped["Age"]
```

Out[29]:
```
<pandas.core.groupby.generic.SeriesGroupBy object at 0x0000018984258BE0>
```

In [30]:
```
grouped["Age"].size()
```

Out[30]:
```
Age
18     4
19     8
20     5
21     5
22     3
23     6
24     4
25     3
26     2
27     6
28     4
29     5
30     7
31     8
32    11
33     3
34     5
35     9
36     6
37     3
38     6
39     3
40     6
41     2
42     2
43     3
44     2
45     3
46     3
47     6
48     5
49     7
50     5
51     2
52     2
53     2
54     4
55     1
56     1
57     2
58     2
59     4
60     3
63     2
64     1
65     2
66     2
67     4
68     3
69     1
70     2
Name: Age, dtype: int64
```

Counting number of each category by count()

In [32]:
```
print(dataset.groupby(["Genre"]).count().reset_index())
```

```
    Genre  CustomerID  Age  Annual Income (k$)  Spending Score (1-100)
0  Female         112  112                 112                     112
1    Male          88   88                  88                      88
```

In [33]:
```
print(dataset.groupby(["Genre"]).mean().reset_index())
```

```
    Genre  CustomerID         Age  Annual Income (k$)  Spending Score (1-100)
0  Female   97.562500   38.098214           59.250000               51.526786
1    Male  104.238636   39.806818           62.227273               48.511364
```

In [34]:
```
print(dataset.groupby(["Genre"]).median().reset_index())
```

```
    Genre  CustomerID   Age  Annual Income (k$)  Spending Score (1-100)
0  Female        94.5  35.0                60.0                    50.0
1    Male       106.5  37.0                62.5                    50.0
```

Assignment 3.2

In [36]:
```
import pandas as pd
data= pd.read_csv("iris.csv")
```

In [41]:
```
print('iris-setosa')
setosa= data["Species"]=='iris-setosa'
print(data[setosa].describe())
print('\niris-versicolor')
setosa= data["Species"]=='iris-versicolor'
print(data[setosa].describe())
print('\niris-virginics')
setosa= data["Species"]=='iris-virginica'
print(data[setosa].describe())
```

```
iris-setosa
        Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm
count  0.0            0.0           0.0            0.0           0.0
mean   NaN            NaN           NaN            NaN           NaN
std    NaN            NaN           NaN            NaN           NaN
min    NaN            NaN           NaN            NaN           NaN
25%    NaN            NaN           NaN            NaN           NaN
50%    NaN            NaN           NaN            NaN           NaN
75%    NaN            NaN           NaN            NaN           NaN
max    NaN            NaN           NaN            NaN           NaN

iris-versicolor
        Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm
count  0.0            0.0           0.0            0.0           0.0
mean   NaN            NaN           NaN            NaN           NaN
std    NaN            NaN           NaN            NaN           NaN
min    NaN            NaN           NaN            NaN           NaN
25%    NaN            NaN           NaN            NaN           NaN
50%    NaN            NaN           NaN            NaN           NaN
75%    NaN            NaN           NaN            NaN           NaN
max    NaN            NaN           NaN            NaN           NaN

iris-virginics
        Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm
count  0.0            0.0           0.0            0.0           0.0
mean   NaN            NaN           NaN            NaN           NaN
std    NaN            NaN           NaN            NaN           NaN
min    NaN            NaN           NaN            NaN           NaN
25%    NaN            NaN           NaN            NaN           NaN
50%    NaN            NaN           NaN            NaN           NaN
75%    NaN            NaN           NaN            NaN           NaN
max    NaN            NaN           NaN            NaN           NaN
```

In [ ]: