ASSIGNMENT 4 – CUSTOMER SEGMENTATION ANALYSIS

Assignment Date	03 October 2022
Team ID	PNT2022TMID27812
Project Name	Smart Lender-Application Credibility Prediction for loan Approval
Student Name	Hariharan A G
Student Roll Number	311519104020
Maximum Marks	2 Marks

1. Import Required Libraries.

In [7]:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sbn
```

2. Download the dataset

Dataset was successfully downloaded as Mall_Customers.csv

In [3]:

```
db = pd.read_csv('Mall_Customers.csv')
db
```

Out[3]:

	CustomerID	Gender	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

3. Perform Below Visualizations.

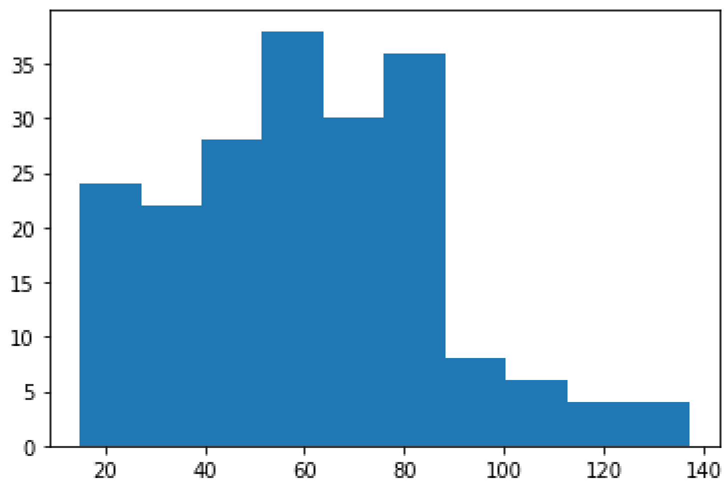
a) Univariate Analysis

In [4]:

```
plt.hist(db['Annual Income (k$)'])
```

Out[4]:

```
(array([24., 22., 28., 38., 30., 36.,  8.,  6.,  4.,  4.]),
 array([ 15. ,  27.2,  39.4,  51.6,  63.8,  76. ,  88.2, 100.4, 112.6,
        124.8, 137. ]),
 )
```

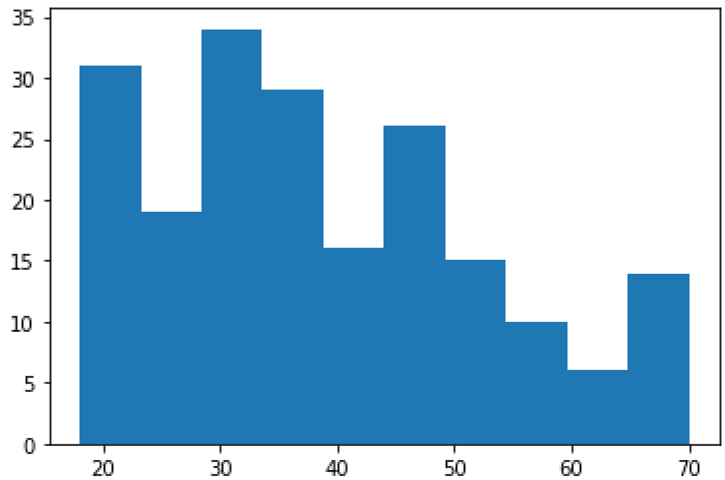


In [5]:

```
plt.hist(db['Age'])
```

Out[5]:

```
(array([31., 19., 34., 29., 16., 26., 15., 10.,  6., 14.]),
 array([18. , 23.2, 28.4, 33.6, 38.8, 44. , 49.2, 54.4, 59.6, 64.8, 70. ]),
 )
```

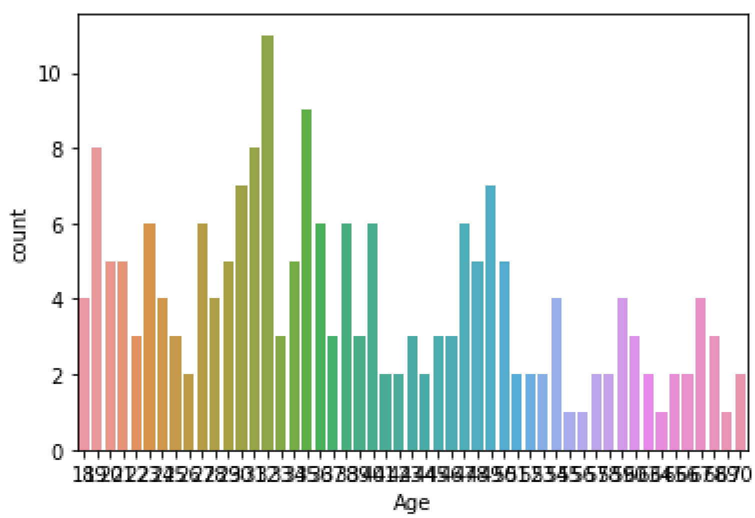


In [8]:

sbn.countplot(db['Age'])

/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  FutureWarning

Out[8]:



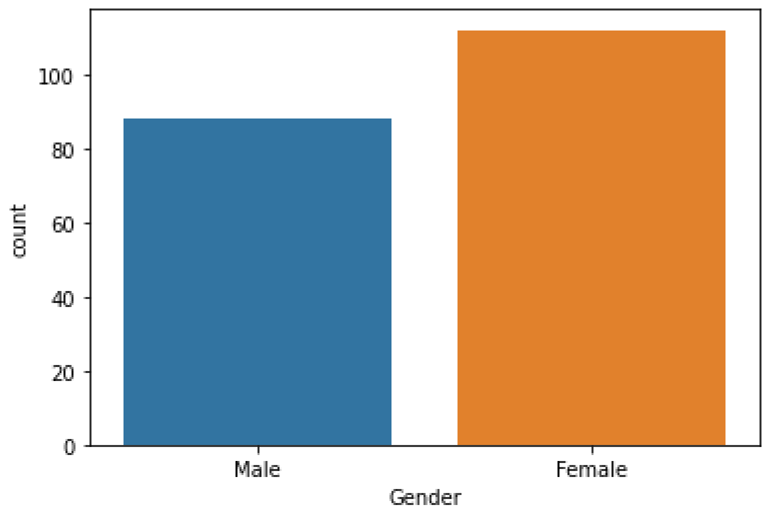
In [9]:

sbn.countplot(db['Gender'])

/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.

FutureWarning

Out[9]:

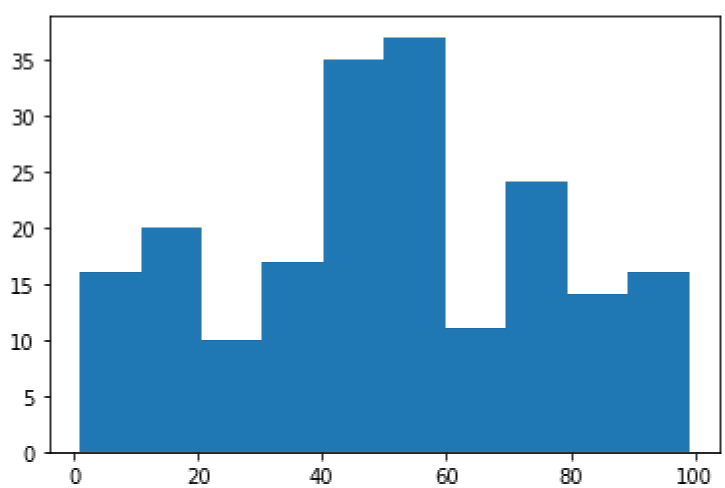


In [10]:

```
plt.hist(db['Spending Score (1-100)'])
```

Out[10]:

```
(array([16., 20., 10., 17., 35., 37., 11., 24., 14., 16.]),
 array([ 1. , 10.8, 20.6, 30.4, 40.2, 50. , 59.8, 69.6, 79.4, 89.2, 99. ]),
 )
```

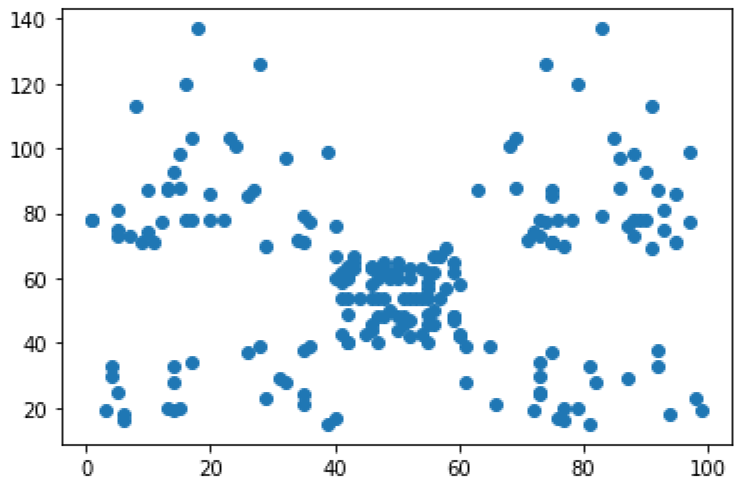


b) Bi- Variate Analysis

In [11]:

```
plt.scatter(db['Spending Score (1-100)'],db['Annual Income (k$)'])
```

Out[11]:



In [12]:

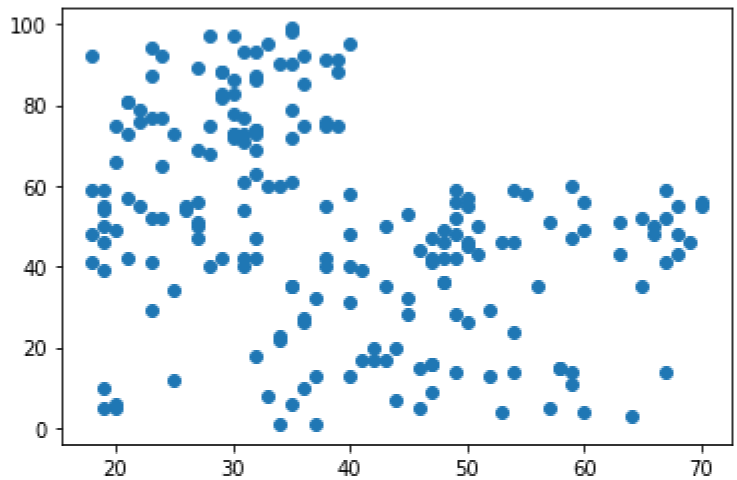
```
plt.scatter(db['Gender'],db['Annual Income (k$)'])
```

Out[12]:

In [13]:

```
plt.scatter(db['Age'],db['Spending Score (1-100)'])
```

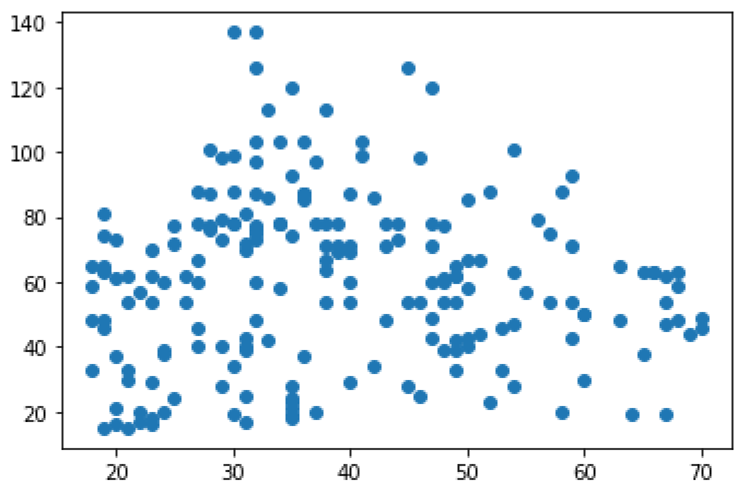
Out[13]:



In [14]:

plt.scatter(db['Age'],db['Annual Income (k$)'])

Out[14]:



In [15]:

sbn.heatmap(db.corr(), annot = **True**)

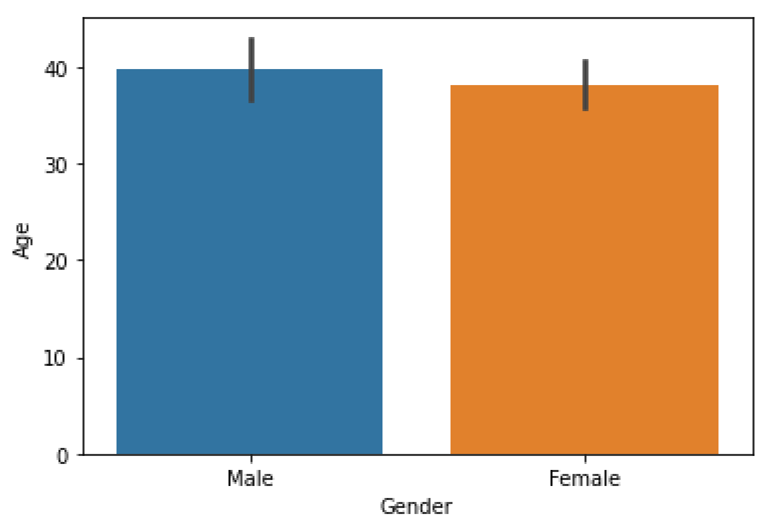
Out[15]:

In [16]:

```
sbn.barplot(db['Gender'], db['Age'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variables as keyword args: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  FutureWarning
```

Out[16]:

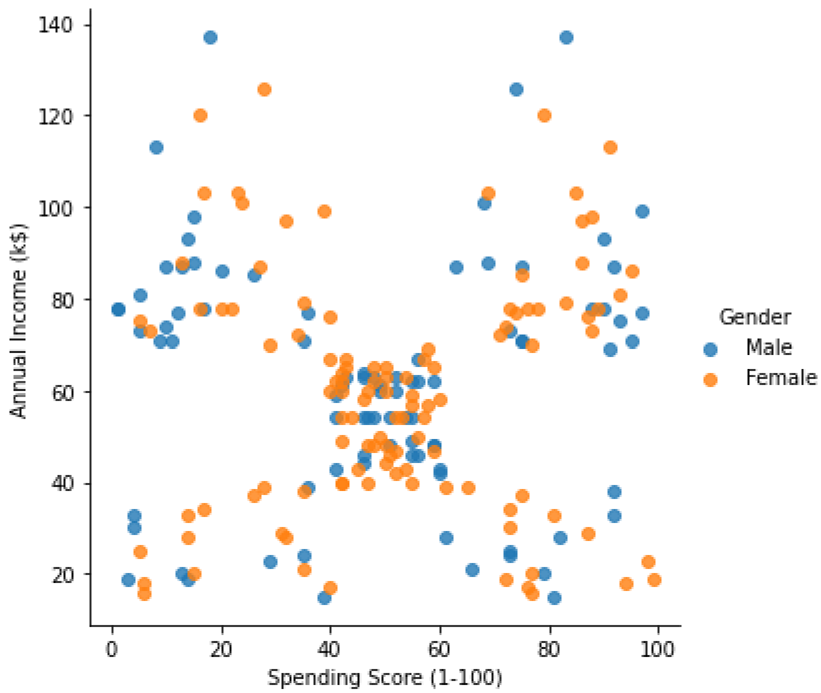


c) Multi-Variate Analysis

In [17]:

```
sbn.lmplot("Spending Score (1-100)","Annual Income (k$)", db, hue="Gender", fit_reg=False);
```

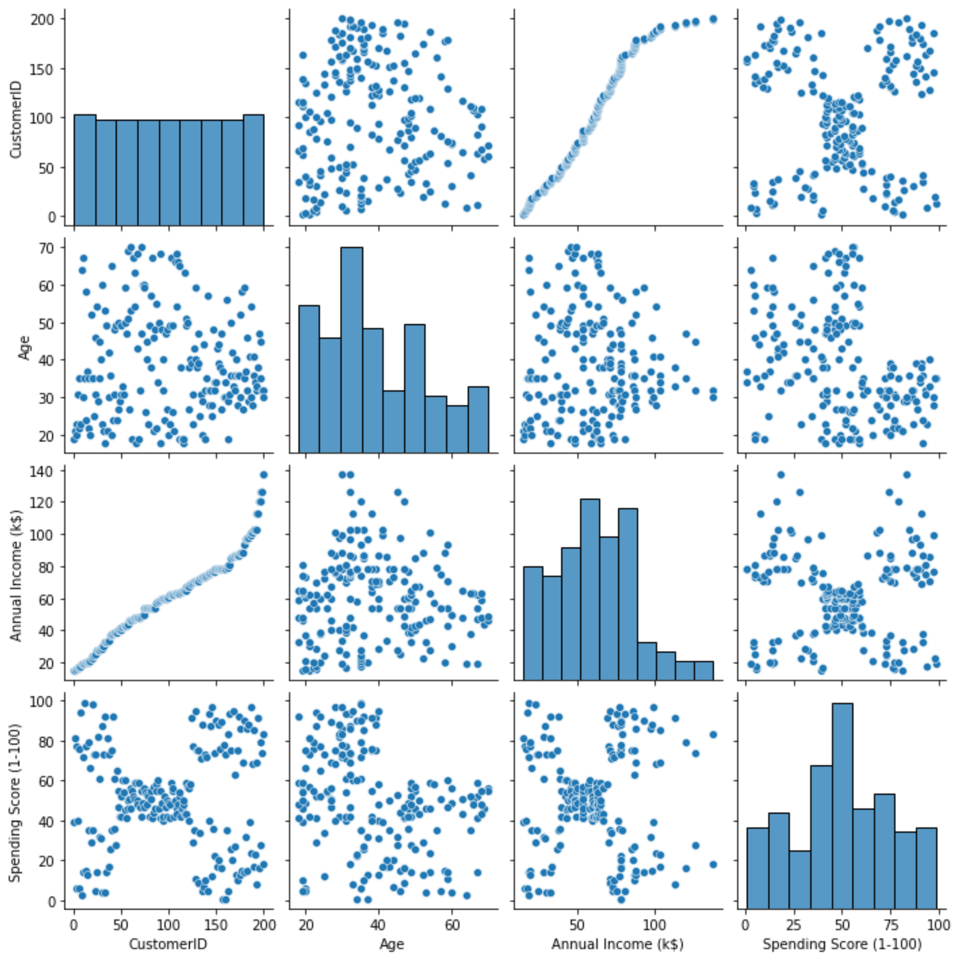
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variables as keyword args: x, y, data. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  FutureWarning



In [18]:

```
sbn.pairplot(db)
```

Out[18]:



4. Perform descriptive statistics on the dataset.

In [19]:

```
db.describe()
```

Out[19]:

	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 [20]:

```
db.describe().T
```

Out[20]:

	count	mean	std	min	25%	50%	75%	max
CustomerID	200.0	100.50	57.879185	1.0	50.75	100.5	150.25	200.0
Age	200.0	38.85	13.969007	18.0	28.75	36.0	49.00	70.0
Annual Income (k$)	200.0	60.56	26.264721	15.0	41.50	61.5	78.00	137.0
Spending Score (1-100)	200.0	50.20	25.823522	1.0	34.75	50.0	73.00	99.0

In [21]:

```
db.dtypes
```

Out[21]:

```
CustomerID                int64
Gender                   object
Age                       int64
Annual Income (k$)        int64
Spending Score (1-100)    int64
dtype: object
```

In [22]:

```
db.var()
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError.  Select only valid columns before calling the reduction.
  """Entry point for launching an IPython kernel.
```

Out[22]:

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

In [23]:

```
db.skew()
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError.  Select only valid columns before calling the reduction.
  """Entry point for launching an IPython kernel.
```

Out[23]:

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

In [24]:

```
db.corr()
```

Out[24]:

	CustomerID	Age	Annual Income (k$)	Spending Score (1-100)
CustomerID	1.000000	-0.026763	0.977548	0.013835
Age	-0.026763	1.000000	-0.012398	-0.327227
Annual Income (k$)	0.977548	-0.012398	1.000000	0.009903
Spending Score (1-100)	0.013835	-0.327227	0.009903	1.000000

In [25]:

```
db.std()
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError.  Select only valid columns before calling the reduction.
  """Entry point for launching an IPython kernel.
```

Out[25]:

```
CustomerID             57.879185
Age                    13.969007
Annual Income (k$)     26.264721
Spending Score (1-100)    25.823522
dtype: float64
```

5. Check for Missing values and deal with them.

In [26]:

```
db.isna().sum()
```

Out[26]:

```
CustomerID             0
Gender                 0
Age                    0
Annual Income (k$)     0
Spending Score (1-100)    0
dtype: int64
```

In [27]:

```
db.isna().sum().sum()
```

Out[27]:

0

In [28]:

```
db.duplicated().sum()
```

Out[28]:

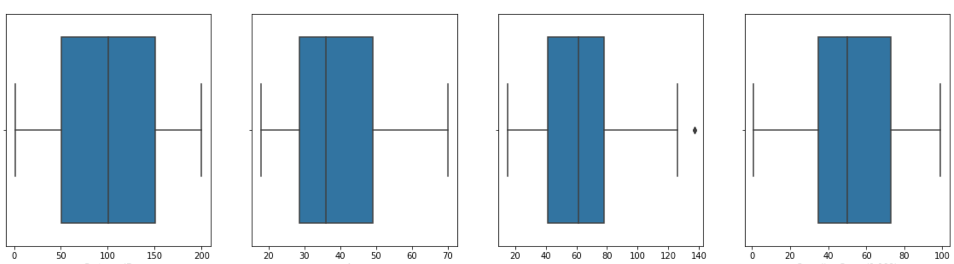
0

6. Find the outliers and replace them outliers

In [29]:

```
fig,ax=plt.subplots(figsize=(25,5))

plt.subplot(1, 5, 2)
sbn.boxplot(x=db['Age'])

plt.subplot(1, 5, 3)
sbn.boxplot(x=db['Annual Income (k$)'])

plt.subplot(1, 5, 4)
sbn.boxplot(x=db['Spending Score (1-100)'])

plt.subplot(1, 5, 1)
sbn.boxplot(x=db['CustomerID'])
```

Out[29]:



In [30]:

```
quantile = db.quantile(q = [0.25, 0.75])
quantile
```

Out[30]:

	CustomerID	Age	Annual Income (k$)	Spending Score (1-100)
0.25	50.75	28.75	41.5	34.75

0.75	150.25	49.00	78.0	73.00

In [31]:

```
quantile.loc[0.75]
```

Out[31]:

```
CustomerID              150.25
Age                      49.00
Annual Income (k$)       78.00
Spending Score (1-100)   73.00
Name: 0.75, dtype: float64
```

In [32]:

```
quantile.loc[0.25]
```

Out[32]:

```
CustomerID              50.75
Age                     28.75
Annual Income (k$)      41.50
Spending Score (1-100)  34.75
Name: 0.25, dtype: float64
```

In [33]:

```
IQR = quantile.iloc[1] - quantile.iloc[0]
IQR
```

Out[33]:

```
CustomerID              99.50
Age                     20.25
Annual Income (k$)      36.50
Spending Score (1-100)  38.25
dtype: float64
```

In [34]:

```
upper = quantile.iloc[1] + (1.5 *IQR)
upper
```

Out[34]:

```
CustomerID              299.500
Age                      79.375
Annual Income (k$)      132.750
Spending Score (1-100)  130.375
dtype: float64
```

In [35]:

```
lower = quantile.iloc[0] - (1.5* IQR)
lower
```

Out[35]:

```
CustomerID             -98.500
Age                     -1.625
Annual Income (k$)     -13.250
Spending Score (1-100)  -22.625
dtype: float64
```

In [36]:

```
db.mean()
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError.  Select only valid columns before calling the reduction.
  """Entry point for launching an IPython kernel.
```

Out[36]:

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

In [37]:

```
db['Annual Income (k$)'].max()
```

Out[37]:

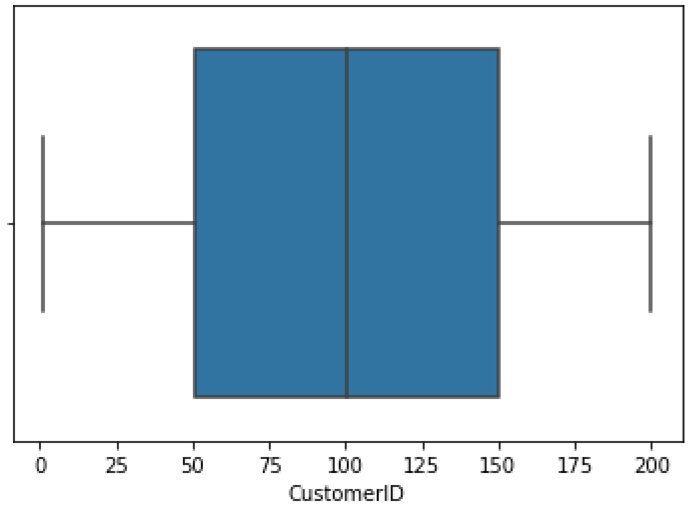
```
137
```

In [38]:

```
sbn.boxplot(db['CustomerID'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  FutureWarning
```

Out[38]:

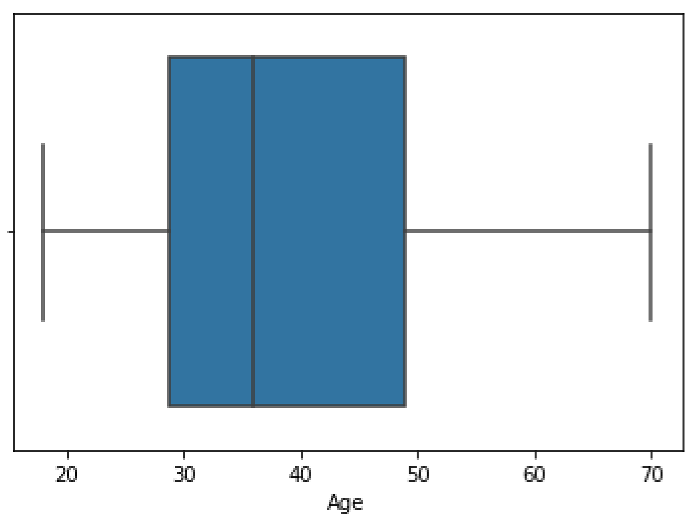


In [39]:

```
sbn.boxplot(db['Age'])
```

/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  FutureWarning

Out[39]:



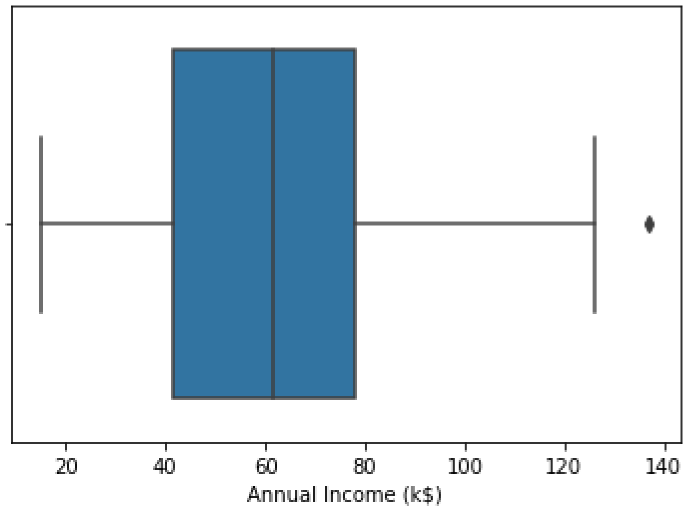
In [40]:

```
sbn.boxplot(db['Annual Income (k$)'])
```

/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.

FutureWarning

Out[40]:



In [41]:

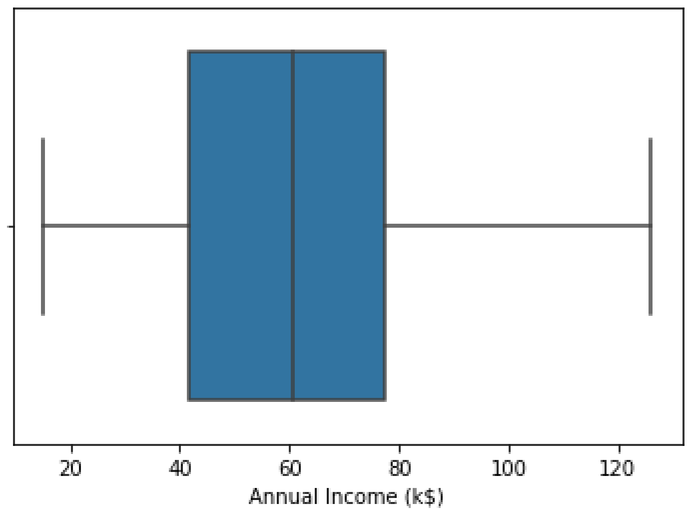
```
db['Annual Income (k$)'] = np.where(db['Annual Income (k$)'] > 132.750, 60.55, db['Annual Income (k$)'])
```

In [42]:

```
sbn.boxplot(db['Annual Income (k$)'])
```

/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  FutureWarning

Out[42]:



In [43]:

db['Annual Income (k$)'].max()

Out[43]:

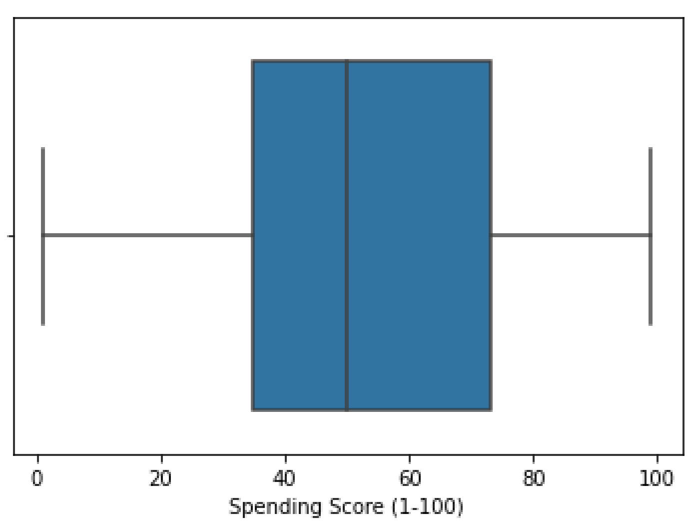
126.0

In [44]:

sbn.boxplot(db['Spending Score (1-100)'])

/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  FutureWarning

Out[44]:



7. Check for Categorical columns and perform encoding.

In [48]:

```
db.select_dtypes(include='object').columns
```

Out[48]:

```
Index(['Gender'], dtype='object')
```

In [46]:

```
db['Gender'].unique()
```

Out[46]:

```
array(['Male', 'Female'], dtype=object)
```

In [49]:

```
db['Gender'].replace({'Male':1,'Female':0},inplace=True)
db
```

Out[49]:

	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)
0	1	1	19	15.00	39
1	2	1	21	15.00	81
2	3	0	20	16.00	6
3	4	0	23	16.00	77
4	5	0	31	17.00	40
...	...	...	...	...	...
195	196	0	35	120.00	79
196	197	0	45	126.00	28
197	198	1	32	126.00	74
198	199	1	32	60.55	18
199	200	1	30	60.55	83

200 rows × 5 columns

In [50]:

```
db.head()
```

Out[50]:

	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)
0	1	1	19	15.0	39
1	2	1	21	15.0	81
2	3	0	20	16.0	6
3	4	0	23	16.0	77
4	5	0	31	17.0	40

8. Scaling the data

In [51]:

```
from sklearn.preprocessing import  StandardScaler
ss = StandardScaler().fit_transform(db)
ss
```

Out[51]:

```
array([[-1.7234121 ,  1.12815215, -1.42456879, -1.78843062, -0.43480148],
       [-1.70609137,  1.12815215, -1.28103541, -1.78843062,  1.19570407],
       [-1.68877065, -0.88640526, -1.3528021 , -1.74850629, -1.71591298],
       [-1.67144992, -0.88640526, -1.13750203, -1.74850629,  1.04041783],
       [-1.6541292 , -0.88640526, -0.56336851, -1.70858195, -0.39597992],
       [-1.63680847, -0.88640526, -1.20926872, -1.70858195,  1.00159627],
       [-1.61948775, -0.88640526, -0.27630176, -1.66865761, -1.71591298],
       [-1.60216702, -0.88640526, -1.13750203, -1.66865761,  1.70038436],
       [-1.5848463 ,  1.12815215,  1.80493225, -1.62873328, -1.83237767],
       [-1.56752558, -0.88640526, -0.6351352 , -1.62873328,  0.84631002],
       [-1.55020485,  1.12815215,  2.02023231, -1.62873328, -1.4053405 ],
       [-1.53288413, -0.88640526, -0.27630176, -1.62873328,  1.89449216],
       [-1.5155634 , -0.88640526,  1.37433211, -1.58880894, -1.36651894],
       [-1.49824268, -0.88640526, -1.06573534, -1.58880894,  1.04041783],
       [-1.48092195,  1.12815215, -0.13276838, -1.58880894, -1.44416206],
       [-1.46360123,  1.12815215, -1.20926872, -1.58880894,  1.11806095],
       [-1.4462805 , -0.88640526, -0.27630176, -1.5488846 , -0.59008772],
       [-1.42895978,  1.12815215, -1.3528021 , -1.5488846 ,  0.61338066],
       [-1.41163905,  1.12815215,  0.94373197, -1.46903593, -0.82301709],
       [-1.39431833, -0.88640526, -0.27630176, -1.46903593,  1.8556706 ],
       [-1.3769976 ,  1.12815215, -0.27630176, -1.42911159, -0.59008772],
       [-1.35967688,  1.12815215, -0.99396865, -1.42911159,  0.88513158],
       [-1.34235616, -0.88640526,  0.51313183, -1.38918726, -1.75473454],
       [-1.32503543,  1.12815215, -0.56336851, -1.38918726,  0.88513158],
       [-1.30771471, -0.88640526,  1.08726535, -1.26941425, -1.4053405 ],
       [-1.29039398,  1.12815215, -0.70690189, -1.26941425,  1.23452563],
       [-1.27307326, -0.88640526,  0.44136514, -1.26941425, -0.7065524 ],
       [-1.25575253,  1.12815215, -0.27630176, -1.26941425,  0.41927286],
       [-1.23843181, -0.88640526,  0.08253169, -1.22948991, -0.74537397],
       [-1.22111108, -0.88640526, -1.13750203, -1.22948991,  1.42863343],
       [-1.20379036,  1.12815215,  1.51786549, -1.18956557, -1.7935561 ],
       [-1.18646963, -0.88640526, -1.28103541, -1.18956557,  0.88513158],
       [-1.16914891,  1.12815215,  1.01549866, -1.06979256, -1.7935561 ],
       [-1.15182818,  1.12815215, -1.49633548, -1.06979256,  1.62274124],
       [-1.13450746, -0.88640526,  0.7284319 , -1.06979256, -1.4053405 ],
       [-1.11718674, -0.88640526, -1.28103541, -1.06979256,  1.19570407],
       [-1.09986601, -0.88640526,  0.22606507, -1.02986823, -1.28887582],
       [-1.08254529, -0.88640526, -0.6351352 , -1.02986823,  0.88513158],
       [-1.06522456, -0.88640526, -0.20453507, -0.91009522, -0.93948177],
       [-1.04790384, -0.88640526, -1.3528021 , -0.91009522,  0.96277471],
       [-1.03058311, -0.88640526,  1.87669894, -0.87017088, -0.59008772],
       [-1.01326239,  1.12815215, -1.06573534, -0.87017088,  1.62274124],
       [-0.99594166,  1.12815215,  0.65666521, -0.83024654, -0.55126616],
```

[-0.97862094, -0.88640526, -0.56336851, -0.83024654, 0.41927286],
[-0.96130021, -0.88640526, 0.7284319 , -0.83024654, -0.86183865],
[-0.94397949, -0.88640526, -1.06573534, -0.83024654, 0.5745591],
[-0.92665877, -0.88640526, 0.80019859, -0.79032221, 0.18634349],
[-0.90933804, -0.88640526, -0.85043527, -0.79032221, -0.12422899],
[-0.89201732, -0.88640526, -0.70690189, -0.79032221, -0.3183368],
[-0.87469659, -0.88640526, -0.56336851, -0.79032221, -0.3183368],
[-0.85737587, -0.88640526, 0.7284319 , -0.71047353, 0.06987881],
[-0.84005514, 1.12815215, -0.41983513, -0.71047353, 0.38045129],
[-0.82273442, -0.88640526, -0.56336851, -0.6705492 , 0.14752193],
[-0.80541369, 1.12815215, 1.4460988 , -0.6705492 , 0.38045129],
[-0.78809297, -0.88640526, 0.80019859, -0.6705492 , -0.20187212],
[-0.77077224, 1.12815215, 0.58489852, -0.6705492 , -0.35715836],
[-0.75345152, -0.88640526, 0.87196528, -0.63062486, -0.00776431],
[-0.73613079, 1.12815215, 2.16376569, -0.63062486, -0.16305055],
[-0.71881007, -0.88640526, -0.85043527, -0.55077619, 0.03105725],
[-0.70148935, 1.12815215, 1.01549866, -0.55077619, -0.16305055],
[-0.68416862, 1.12815215, 2.23553238, -0.55077619, 0.22516505],
[-0.6668479 , 1.12815215, -1.42456879, -0.55077619, 0.18634349],
[-0.64952717, -0.88640526, 2.02023231, -0.51085185, 0.06987881],
[-0.63220645, -0.88640526, 1.08726535, -0.51085185, 0.34162973],
[-0.61488572, 1.12815215, 1.73316556, -0.47092751, 0.03105725],
[-0.597565 , 1.12815215, -1.49633548, -0.47092751, 0.34162973],
[-0.58024427, -0.88640526, 0.29783176, -0.47092751, -0.00776431],
[-0.56292355, -0.88640526, 2.091999 , -0.47092751, -0.08540743],
[-0.54560282, 1.12815215, -1.42456879, -0.47092751, 0.34162973],
[-0.5282821 , -0.88640526, -0.49160182, -0.47092751, -0.12422899],
[-0.51096138, 1.12815215, 2.23553238, -0.43100318, 0.18634349],
[-0.49364065, -0.88640526, 0.58489852, -0.43100318, -0.3183368],
[-0.47631993, -0.88640526, 1.51786549, -0.39107884, -0.04658587],
[-0.4589992 , -0.88640526, 1.51786549, -0.39107884, 0.22516505],
[-0.44167848, 1.12815215, 1.4460988 , -0.23138149, -0.12422899],
[-0.42435775, 1.12815215, -0.92220196, -0.23138149, 0.14752193],
[-0.40703703, -0.88640526, 0.44136514, -0.23138149, 0.10870037],
[-0.3897163 , 1.12815215, 0.08253169, -0.23138149, -0.08540743],
[-0.37239558, -0.88640526, -1.13750203, -0.23138149, 0.06987881],
[-0.35507485, -0.88640526, 0.7284319 , -0.23138149, -0.3183368],
[-0.33775413, 1.12815215, 1.30256542, -0.23138149, 0.03105725],
[-0.3204334 , 1.12815215, -0.06100169, -0.23138149, 0.18634349],
[-0.30311268, 1.12815215, 2.02023231, -0.23138149, -0.35715836],
[-0.28579196, -0.88640526, 0.51313183, -0.23138149, -0.24069368],
[-0.26847123, -0.88640526, -1.28103541, -0.23138149, 0.26398661],
[-0.25115051, 1.12815215, 0.65666521, -0.23138149, -0.16305055],
[-0.23382978, -0.88640526, 1.15903204, -0.11160848, 0.30280817],
[-0.21650906, -0.88640526, -1.20926872, -0.11160848, 0.18634349],
[-0.19918833, -0.88640526, -0.34806844, -0.07168415, 0.38045129],
[-0.18186761, -0.88640526, 0.80019859, -0.07168415, -0.16305055],
[-0.16454688, -0.88640526, 2.091999 , -0.03175981, 0.18634349],
[-0.14722616, 1.12815215, -1.49633548, -0.03175981, -0.35715836],
[-0.12990543, 1.12815215, 0.65666521, 0.00816453, -0.04658587],
[-0.11258471, -0.88640526, 0.08253169, 0.00816453, -0.39597992],
[-0.09526399, -0.88640526, -0.49160182, 0.00816453, -0.3183368],
[-0.07794326, 1.12815215, -1.06573534, 0.00816453, 0.06987881],

```
[-0.06062254, -0.88640526,  0.58489852,  0.00816453, -0.12422899],
[-0.04330181, -0.88640526, -0.85043527,  0.00816453, -0.00776431],
[-0.02598109,  1.12815215,  0.65666521,  0.04808886, -0.3183368 ],
[-0.00866036,  1.12815215, -1.3528021 ,  0.04808886, -0.04658587],
[ 0.00866036, -0.88640526, -1.13750203,  0.0880132 , -0.35715836],
[ 0.02598109, -0.88640526,  0.7284319 ,  0.0880132 , -0.08540743],
[ 0.04330181,  1.12815215,  2.02023231,  0.0880132 ,  0.34162973],
[ 0.06062254,  1.12815215, -0.92220196,  0.0880132 ,  0.18634349],
[ 0.07794326,  1.12815215,  0.7284319 ,  0.0880132 ,  0.22516505],
[ 0.09526399, -0.88640526, -1.28103541,  0.0880132 , -0.3183368 ],
[ 0.11258471, -0.88640526,  1.94846562,  0.12793754, -0.00776431],
[ 0.12990543,  1.12815215,  1.08726535,  0.12793754, -0.16305055],
[ 0.14722616,  1.12815215,  2.091999  ,  0.12793754, -0.27951524],
[ 0.16454688,  1.12815215,  1.94846562,  0.12793754, -0.08540743],
[ 0.18186761,  1.12815215,  1.87669894,  0.12793754,  0.06987881],
[ 0.19918833, -0.88640526, -1.42456879,  0.12793754,  0.14752193],
[ 0.21650906, -0.88640526, -0.06100169,  0.16786187, -0.3183368 ],
[ 0.23382978,  1.12815215, -1.42456879,  0.16786187, -0.16305055],
[ 0.25115051, -0.88640526, -1.49633548,  0.20778621, -0.08540743],
[ 0.26847123, -0.88640526, -1.42456879,  0.20778621, -0.00776431],
[ 0.28579196, -0.88640526,  1.73316556,  0.20778621, -0.27951524],
[ 0.30311268, -0.88640526,  0.7284319 ,  0.20778621,  0.34162973],
[ 0.3204334 , -0.88640526,  0.87196528,  0.28763488, -0.27951524],
[ 0.33775413, -0.88640526,  0.80019859,  0.28763488,  0.26398661],
[ 0.35507485,  1.12815215, -0.85043527,  0.28763488,  0.22516505],
[ 0.37239558, -0.88640526, -0.06100169,  0.28763488, -0.39597992],
[ 0.3897163 , -0.88640526,  0.08253169,  0.36748356,  0.30280817],
[ 0.40703703,  1.12815215,  0.010765  ,  0.36748356,  1.58391968],
[ 0.42435775, -0.88640526, -1.13750203,  0.40740789, -0.82301709],
[ 0.44167848, -0.88640526, -0.56336851,  0.40740789,  1.04041783],
[ 0.4589992 ,  1.12815215,  0.29783176,  0.44733223, -0.59008772],
[ 0.47631993,  1.12815215,  0.08253169,  0.44733223,  1.73920592],
[ 0.49364065,  1.12815215,  1.4460988 ,  0.44733223, -1.52180518],
[ 0.51096138,  1.12815215, -0.06100169,  0.44733223,  0.96277471],
[ 0.5282821 ,  1.12815215,  0.58489852,  0.44733223, -1.5994483 ],
[ 0.54560282,  1.12815215,  0.010765  ,  0.44733223,  0.96277471],
[ 0.56292355, -0.88640526, -0.99396865,  0.48725657, -0.62890928],
[ 0.58024427, -0.88640526, -0.56336851,  0.48725657,  0.80748846],
[ 0.597565  ,  1.12815215, -1.3528021 ,  0.5271809 , -1.75473454],
[ 0.61488572, -0.88640526, -0.70690189,  0.5271809 ,  1.46745499],
[ 0.63220645, -0.88640526,  0.36959845,  0.5271809 , -1.67709142],
[ 0.64952717,  1.12815215, -0.49160182,  0.5271809 ,  0.88513158],
[ 0.6668479 ,  1.12815215, -1.42456879,  0.56710524, -1.56062674],
[ 0.68416862, -0.88640526, -0.27630176,  0.56710524,  0.84631002],
[ 0.70148935, -0.88640526,  1.30256542,  0.60702958, -1.75473454],
[ 0.71881007,  1.12815215, -0.49160182,  0.60702958,  1.6615628 ],
[ 0.73613079, -0.88640526, -0.77866858,  0.64695391, -0.39597992],
[ 0.75345152, -0.88640526, -0.49160182,  0.64695391,  1.42863343],
[ 0.77077224,  1.12815215, -0.99396865,  0.68687825, -1.48298362],
[ 0.78809297,  1.12815215, -0.77866858,  0.68687825,  1.81684904],
[ 0.80541369,  1.12815215,  0.65666521,  0.68687825, -0.55126616],
[ 0.82273442, -0.88640526, -0.49160182,  0.68687825,  0.92395314],
[ 0.84005514, -0.88640526, -0.34806844,  0.72680259, -1.09476801],
```

```
[ 0.85737587,  1.12815215, -0.34806844,  0.72680259,  1.54509812],
[ 0.87469659,  1.12815215,  0.29783176,  0.72680259, -1.28887582],
[ 0.89201732,  1.12815215,  0.010765  ,  0.72680259,  1.46745499],
[ 0.90933804, -0.88640526,  0.36959845,  0.72680259, -1.17241113],
[ 0.92665877, -0.88640526, -0.06100169,  0.72680259,  1.00159627],
[ 0.94397949, -0.88640526,  0.58489852,  0.72680259, -1.32769738],
[ 0.96130021, -0.88640526, -0.85043527,  0.72680259,  1.50627656],
[ 0.97862094,  1.12815215, -0.13276838,  0.72680259, -1.91002079],
[ 0.99594166, -0.88640526, -0.6351352 ,  0.72680259,  1.07923939],
[ 1.01326239,  1.12815215, -0.34806844,  0.72680259, -1.91002079],
[ 1.03058311, -0.88640526, -0.6351352 ,  0.72680259,  0.88513158],
[ 1.04790384, -0.88640526,  1.23079873,  0.76672692, -0.59008772],
[ 1.06522456, -0.88640526, -0.70690189,  0.76672692,  1.27334719],
[ 1.08254529,  1.12815215, -1.42456879,  0.8465756 , -1.75473454],
[ 1.09986601, -0.88640526, -0.56336851,  0.8465756 ,  1.6615628 ],
[ 1.11718674,  1.12815215,  0.80019859,  1.00627294, -0.93948177],
[ 1.13450746, -0.88640526, -0.20453507,  1.00627294,  0.96277471],
[ 1.15182818,  1.12815215,  0.22606507,  1.04619728, -1.17241113],
[ 1.16914891, -0.88640526, -0.41983513,  1.04619728,  1.73920592],
[ 1.18646963, -0.88640526, -0.20453507,  1.08612162, -0.90066021],
[ 1.20379036,  1.12815215, -0.49160182,  1.08612162,  0.49691598],
[ 1.22111108,  1.12815215,  0.08253169,  1.08612162, -1.44416206],
[ 1.23843181,  1.12815215, -0.77866858,  1.08612162,  0.96277471],
[ 1.25575253,  1.12815215, -0.20453507,  1.08612162, -1.56062674],
[ 1.27307326,  1.12815215, -0.20453507,  1.08612162,  1.62274124],
[ 1.29039398, -0.88640526,  0.94373197,  1.12604595, -1.44416206],
[ 1.30771471, -0.88640526, -0.6351352 ,  1.12604595,  1.38981187],
[ 1.32503543,  1.12815215,  1.37433211,  1.12604595, -1.36651894],
[ 1.34235616,  1.12815215, -0.85043527,  1.12604595,  0.72984534],
[ 1.35967688,  1.12815215,  1.4460988 ,  1.32566764, -1.4053405 ],
[ 1.3769976 ,  1.12815215, -0.27630176,  1.32566764,  1.54509812],
[ 1.39431833, -0.88640526, -0.13276838,  1.48536498, -0.7065524 ],
[ 1.41163905, -0.88640526, -0.49160182,  1.48536498,  1.38981187],
[ 1.42895978,  1.12815215,  0.51313183,  1.52528932, -1.36651894],
[ 1.4462805 , -0.88640526, -0.70690189,  1.52528932,  1.46745499],
[ 1.46360123, -0.88640526,  0.15429838,  1.56521366, -0.43480148],
[ 1.48092195,  1.12815215, -0.6351352 ,  1.56521366,  1.81684904],
[ 1.49824268, -0.88640526,  1.08726535,  1.64506233, -1.01712489],
[ 1.5155634 ,  1.12815215, -0.77866858,  1.64506233,  0.69102378],
[ 1.53288413, -0.88640526,  0.15429838,  1.724911  , -1.28887582],
[ 1.55020485, -0.88640526, -0.20453507,  1.724911  ,  1.35099031],
[ 1.56752558, -0.88640526, -0.34806844,  1.724911  , -1.05594645],
[ 1.5848463 , -0.88640526, -0.49160182,  1.724911  ,  0.72984534],
[ 1.60216702,  1.12815215, -0.41983513,  2.12415437, -1.63826986],
[ 1.61948775, -0.88640526, -0.06100169,  2.12415437,  1.58391968],
[ 1.63680847, -0.88640526,  0.58489852,  2.40362473, -1.32769738],
[ 1.6541292 , -0.88640526, -0.27630176,  2.40362473,  1.11806095],
[ 1.67144992, -0.88640526,  0.44136514,  2.64317075, -0.86183865],
[ 1.68877065,  1.12815215, -0.49160182,  2.64317075,  0.92395314],
[ 1.70609137,  1.12815215, -0.49160182,  0.03012291, -1.25005425],
[ 1.7234121 ,  1.12815215, -0.6351352 ,  0.03012291,  1.27334719]])
```

9. Perform any of the clustering algorithms

In [52]:

```
from sklearn.cluster import KMeans
TWSS = []
k = list(range(2,9))

for i in k:
  kmeans = KMeans(n_clusters = i , init = 'k-means++')
  kmeans.fit(db)
  TWSS.append(kmeans.inertia_)
TWSS
```

Out[52]:

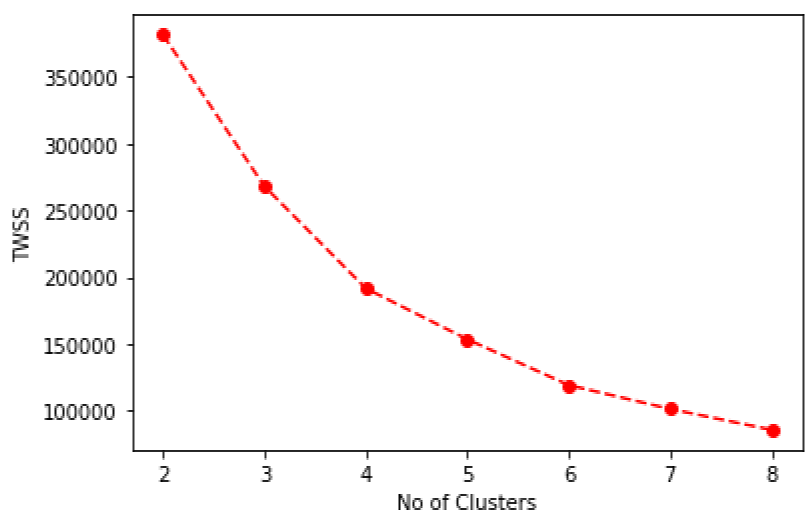
```
[381507.64738523855,
 268062.55433747417,
 191575.26912927354,
 153463.37722463682,
 119166.15727643928,
 101320.9360018038,
 85717.76124902876]
```

In [54]:

```
plt.plot(k,TWSS, 'ro--')
plt.xlabel('No of Clusters')
plt.ylabel('TWSS')
```

Out[54]:

```
Text(0, 0.5, 'TWSS')
```



In [55]:

```
model = KMeans(n_clusters = 4)
model.fit(db)
```

Out[55]:

```
KMeans(n_clusters=4)
```

In [56]:

```
mb = pd.Series(model.labels_)
db['Cluster'] = mb
db
```

Out[56]:

	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)	Cluster
0	1	1	19	15.00	39	1
1	2	1	21	15.00	81	1
2	3	0	20	16.00	6	1
3	4	0	23	16.00	77	1
4	5	0	31	17.00	40	1
...	...	...	...	...	...	...
195	196	0	35	120.00	79	0
196	197	0	45	126.00	28	3
197	198	1	32	126.00	74	0
198	199	1	32	60.55	18	3
199	200	1	30	60.55	83	0

200 rows × 6 columns

In [57]:

```
mb=pd.Series(model.labels_)
db.head(3)
```

Out[57]:

	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)	Cluster
0	1	1	19	15.0	39	1
1	2	1	21	15.0	81	1
2	3	0	20	16.0	6	1

10. Add the cluster data with the primary dataset

In [58]:

```
db['Cluster']=kmeans.labels_
db.head()
```

Out[58]:

	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)	Cluster
0	1	1	19	15.0	39	5

1	2	1	21	15.0	81	3
2	3	0	20	16.0	6	5
3	4	0	23	16.0	77	3
4	5	0	31	17.0	40	5

In [59]:

```
db.tail()
```

Out[59]:

	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)	Cluster
195	196	0	35	120.00	79	4
196	197	0	45	126.00	28	7
197	198	1	32	126.00	74	4
198	199	1	32	60.55	18	7
199	200	1	30	60.55	83	4

11. Split the data into dependent and independent variables.

In [62]:

```
X=db.drop('Cluster',axis=1)
Y=db['Cluster']
```

In [61]:

```
y=db['Cluster']
y
```

Out[61]:

```
0      5
1      3
2      5
3      3
4      5
      ..
195    4
196    7
197    4
198    7
199    4
Name: Cluster, Length: 200, dtype: int32
```

In [63]:

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,Y,test_size=0.2,random_state=42)
```

In [ ]:

```
print("Number transactions X_train dataset: ", X_train.shape)
print("Number transactions y_train dataset: ", y_train.shape)
```

```
print("Number transactions X_test dataset: ", X_test.shape)
print("Number transactions y_test dataset: ", y_test.shape)
```

```
Number transactions X_train dataset:  (160, 5)
Number transactions y_train dataset:  (160,)
Number transactions X_test dataset:  (40, 5)
Number transactions y_test dataset:  (40,)
```

12. Split the data into training and testing

In [64]:

```
X_train
```

Out[64]:

	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)
79	80	0	49	54.0	42
197	198	1	32	126.0	74
38	39	0	36	37.0	26
24	25	0	54	28.0	14
122	123	0	40	69.0	58
...	...	...	...	...	...
106	107	0	66	63.0	50
14	15	1	37	20.0	13
92	93	1	48	60.0	49
179	180	1	35	93.0	90
102	103	1	67	62.0	59

160 rows × 5 columns

In [65]:

```
X_test
```

Out[65]:

	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)
95	96	1	24	60.0	52
15	16	1	22	20.0	79
30	31	1	60	30.0	4
158	159	1	34	78.0	1
128	129	1	59	71.0	11
115	116	0	19	65.0	50
69	70	0	32	48.0	47
170	171	1	40	87.0	13
174	175	0	52	88.0	13
45	46	0	24	39.0	65
66	67	0	43	48.0	50
182	183	1	46	98.0	15
165	166	0	36	85.0	75
78	79	0	23	54.0	52
186	187	0	54	101.0	24
177	178	1	27	88.0	69

56	57	0	51	44.0	50
152	153	0	44	78.0	20
82	83	1	67	54.0	41
68	69	1	19	48.0	59
124	125	0	23	70.0	29
16	17	0	35	21.0	35
148	149	0	34	78.0	22
93	94	0	40	60.0	40
65	66	1	18	48.0	59
60	61	1	70	46.0	56
84	85	0	21	54.0	57
67	68	0	68	48.0	48
125	126	0	31	70.0	77
132	133	0	25	72.0	34
9	10	0	30	19.0	72
18	19	1	52	23.0	29
55	56	1	47	43.0	41
75	76	1	26	54.0	54
150	151	1	43	78.0	17
104	105	1	49	62.0	56
135	136	0	29	73.0	88
137	138	1	32	73.0	73
164	165	1	50	85.0	26
76	77	0	45	54.0	53

In [66]:

```
y_train
```

Out[66]:

```
79     1
197    4
38     5
24     5
122    6
      ..
106    6
14     5
92     6
179    4
102    6
Name: Cluster, Length: 160, dtype: int32
```

In [67]:

```
y_test
```

Out[67]:

```
95     6
15     3
30     5
158    0
128    0
115    6
```

```
69     1
170    7
174    7
45     3
66     1
182    7
165    4
78     1
186    7
177    4
56     1
152    0
82     1
68     1
124    0
16     5
148    0
93     6
65     1
60     1
84     6
67     1
125    2
132    0
9      3
18     5
55     1
75     1
150    0
104    6
135    2
137    2
164    7
76     1
Name: Cluster, dtype: int32
```

13. Build the Model

In [68]:

```python
from sklearn.linear_model import LogisticRegression
model=LogisticRegression()
model.fit(X_train,y_train)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/linear_model/_logistic.py:818: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
```

extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG,

Out[68]:

LogisticRegression()

14. Train the Model

In [69]:

```
model.score(X_train,y_train)
```

Out[69]:

0.85625

15. Test the Model

In [70]:

```
model.score(X_test,y_test)
```

Out[70]:

0.725

16. Measure the performance using Evaluation Metrics.

In [71]:

```
from sklearn.metrics import confusion_matrix,classification_report
y_pred=model.predict(X_test)
confusion_matrix(y_test,y_pred)
```

Out[71]:

```
array([[ 3,  0,  0,  0,  0,  0,  0,  4],
       [ 0, 11,  0,  0,  0,  0,  1,  0],
       [ 0,  0,  3,  0,  0,  0,  0,  0],
       [ 0,  0,  0,  3,  0,  0,  0,  0],
       [ 0,  0,  1,  0,  1,  0,  0,  0],
       [ 0,  0,  0,  0,  0,  3,  0,  0],
       [ 0,  0,  1,  0,  1,  0,  3,  0],
       [ 3,  0,  0,  0,  0,  0,  0,  2]])
```

In [72]:

```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.50      0.43      0.46         7
           1       1.00      0.92      0.96        12
           2       0.60      1.00      0.75         3
```

3	1.00	1.00	1.00	3
4	0.50	0.50	0.50	2
5	1.00	1.00	1.00	3
6	0.75	0.60	0.67	5
7	0.33	0.40	0.36	5
accuracy			0.73	40
macro avg	0.71	0.73	0.71	40
weighted avg	0.74	0.72	0.73	40