```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from matplotlib import rcParams
```

Load the dataset into the tool.

```
from google.colab import files
files.upload()
```

```
<IPython.core.display.HTML object>
```

```
Saving Mall_Customers.csv to Mall_Customers (1).csv
```

```
{'Mall_Customers.csv': b'CustomerID,Gender,Age,Annual Income
(k$),Spending Score (1-100)\r\n1,Male,19,15,39\r\n2,Male,21,15,81\r\
n3,Female,20,16,6\r\n4,Female,23,16,77\r\n5,Female,31,17,40\r\
n6,Female,22,17,76\r\n7,Female,35,18,6\r\n8,Female,23,18,94\r\
n9,Male,64,19,3\r\n10,Female,30,19,72\r\n11,Male,67,19,14\r\
n12,Female,35,19,99\r\n13,Female,58,20,15\r\n14,Female,24,20,77\r\
n15,Male,37,20,13\r\n16,Male,22,20,79\r\n17,Female,35,21,35\r\
n18,Male,20,21,66\r\n19,Male,52,23,29\r\n20,Female,35,23,98\r\
n21,Male,35,24,35\r\n22,Male,25,24,73\r\n23,Female,46,25,5\r\
n24,Male,31,25,73\r\n25,Female,54,28,14\r\n26,Male,29,28,82\r\
n27,Female,45,28,32\r\n28,Male,35,28,61\r\n29,Female,40,29,31\r\
n30,Female,23,29,87\r\n31,Male,60,30,4\r\n32,Female,21,30,73\r\
n33,Male,53,33,4\r\n34,Male,18,33,92\r\n35,Female,49,33,14\r\
n36,Female,21,33,81\r\n37,Female,42,34,17\r\n38,Female,30,34,73\r\
n39,Female,36,37,26\r\n40,Female,20,37,75\r\n41,Female,65,38,35\r\
n42,Male,24,38,92\r\n43,Male,48,39,36\r\n44,Female,31,39,61\r\
n45,Female,49,39,28\r\n46,Female,24,39,65\r\n47,Female,50,40,55\r\
n48,Female,27,40,47\r\n49,Female,29,40,42\r\n50,Female,31,40,42\r\
n51,Female,49,42,52\r\n52,Male,33,42,60\r\n53,Female,31,43,54\r\
n54,Male,59,43,60\r\n55,Female,50,43,45\r\n56,Male,47,43,41\r\
n57,Female,51,44,50\r\n58,Male,69,44,46\r\n59,Female,27,46,51\r\
n60,Male,53,46,46\r\n61,Male,70,46,56\r\n62,Male,19,46,55\r\
n63,Female,67,47,52\r\n64,Female,54,47,59\r\n65,Male,63,48,51\r\
n66,Male,18,48,59\r\n67,Female,43,48,50\r\n68,Female,68,48,48\r\
n69,Male,19,48,59\r\n70,Female,32,48,47\r\n71,Male,70,49,55\r\
n72,Female,47,49,42\r\n73,Female,60,50,49\r\n74,Female,60,50,56\r\
n75,Male,59,54,47\r\n76,Male,26,54,54\r\n77,Female,45,54,53\r\
n78,Male,40,54,48\r\n79,Female,23,54,52\r\n80,Female,49,54,42\r\
n81,Male,57,54,51\r\n82,Male,38,54,55\r\n83,Male,67,54,41\r\
n84,Female,46,54,44\r\n85,Female,21,54,57\r\n86,Male,48,54,46\r\
n87,Female,55,57,58\r\n88,Female,22,57,55\r\n89,Female,34,58,60\r\
n90,Female,50,58,46\r\n91,Female,68,59,55\r\n92,Male,18,59,41\r\
n93,Male,48,60,49\r\n94,Female,40,60,40\r\n95,Female,32,60,42\r\
n96,Male,24,60,52\r\n97,Female,47,60,47\r\n98,Female,27,60,50\r\
n99,Male,48,61,42\r\n100,Male,20,61,49\r\n101,Female,23,62,41\r\
n102,Female,49,62,48\r\n103,Male,67,62,59\r\n104,Male,26,62,55\r\
```

```
n105,Male,49,62,56\r\n106,Female,21,62,42\r\n107,Female,66,63,50\r\
n108,Male,54,63,46\r\n109,Male,68,63,43\r\n110,Male,66,63,48\r\
n111,Male,65,63,52\r\n112,Female,19,63,54\r\n113,Female,38,64,42\r\
n114,Male,19,64,46\r\n115,Female,18,65,48\r\n116,Female,19,65,50\r\
n117,Female,63,65,43\r\n118,Female,49,65,59\r\n119,Female,51,67,43\r\
n120,Female,50,67,57\r\n121,Male,27,67,56\r\n122,Female,38,67,40\r\
n123,Female,40,69,58\r\n124,Male,39,69,91\r\n125,Female,23,70,29\r\
n126,Female,31,70,77\r\n127,Male,43,71,35\r\n128,Male,40,71,95\r\
n129,Male,59,71,11\r\n130,Male,38,71,75\r\n131,Male,47,71,9\r\
n132,Male,39,71,75\r\n133,Female,25,72,34\r\n134,Female,31,72,71\r\
n135,Male,20,73,5\r\n136,Female,29,73,88\r\n137,Female,44,73,7\r\
n138,Male,32,73,73\r\n139,Male,19,74,10\r\n140,Female,35,74,72\r\
n141,Female,57,75,5\r\n142,Male,32,75,93\r\n143,Female,28,76,40\r\
n144,Female,32,76,87\r\n145,Male,25,77,12\r\n146,Male,28,77,97\r\
n147,Male,48,77,36\r\n148,Female,32,77,74\r\n149,Female,34,78,22\r\
n150,Male,34,78,90\r\n151,Male,43,78,17\r\n152,Male,39,78,88\r\
n153,Female,44,78,20\r\n154,Female,38,78,76\r\n155,Female,47,78,16\r\
n156,Female,27,78,89\r\n157,Male,37,78,1\r\n158,Female,30,78,78\r\
n159,Male,34,78,1\r\n160,Female,30,78,73\r\n161,Female,56,79,35\r\
n162,Female,29,79,83\r\n163,Male,19,81,5\r\n164,Female,31,81,93\r\
n165,Male,50,85,26\r\n166,Female,36,85,75\r\n167,Male,42,86,20\r\
n168,Female,33,86,95\r\n169,Female,36,87,27\r\n170,Male,32,87,63\r\
n171,Male,40,87,13\r\n172,Male,28,87,75\r\n173,Male,36,87,10\r\
n174,Male,36,87,92\r\n175,Female,52,88,13\r\n176,Female,30,88,86\r\
n177,Male,58,88,15\r\n178,Male,27,88,69\r\n179,Male,59,93,14\r\
n180,Male,35,93,90\r\n181,Female,37,97,32\r\n182,Female,32,97,86\r\
n183,Male,46,98,15\r\n184,Female,29,98,88\r\n185,Female,41,99,39\r\
n186,Male,30,99,97\r\n187,Female,54,101,24\r\n188,Male,28,101,68\r\
n189,Female,41,103,17\r\n190,Female,36,103,85\r\n191,Female,34,103,23\
r\n192,Female,32,103,69\r\n193,Male,33,113,8\r\n194,Female,38,113,91\
r\n195,Female,47,120,16\r\n196,Female,35,120,79\r\
n197,Female,45,126,28\r\n198,Male,32,126,74\r\n199,Male,32,137,18\r\
n200,Male,30,137,83\r\n'}
```

```
df = pd.read_csv("Mall_Customers.csv")
df.head()
```

	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

```
dummy=pd.get_dummies(df['Gender'])
dummy.head()
```

	Female	Male
0	0	1
1	0	1
2	1	0

```
3       1     0
4       1     0
```

```
df2=pd.concat((df,dummy),axis=1)
df2.head()
```

```
   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


   Female  Male
0       0     1
1       0     1
2       1     0
3       1     0
4       1     0
```

```
df2.drop(['Gender'],axis=1)
```

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

```
199         200   30                  137                        83
0     1

[200 rows x 6 columns]
```

```
df2=df2.drop(['Gender'],axis=1)
df2.head()
```

```
   CustomerID  Age  Annual Income (k$)  Spending Score (1-100)  Female
Male
0           1   19                  15                      39       0
1
1           2   21                  15                      81       0
1
2           3   20                  16                       6       1
0
3           4   23                  16                      77       1
0
4           5   31                  17                      40       1
0
```

```
df2=df2.drop(['Male'],axis=1)
df2.head()
```

```
   CustomerID  Age  Annual Income (k$)  Spending Score (1-100)  Female
0           1   19                  15                      39       0
1           2   21                  15                      81       0
2           3   20                  16                       6       1
3           4   23                  16                      77       1
4           5   31                  17                      40       1
```

```
df2.rename(columns={"Female":"Gender"})
```

```
     CustomerID  Age  Annual Income (k$)  Spending Score (1-100)
Gender
0             1   19                  15                      39
0
1             2   21                  15                      81
0
2             3   20                  16                       6
1
3             4   23                  16                      77
1
4             5   31                  17                      40
1
..          ...  ...                 ...                     ...     .
..
195         196   35                  120                     79
1
196         197   45                  126                     28
1
```

```
197         198   32                  126                      74
0
198         199   32                  137                      18
0
199         200   30                  137                      83
0

[200 rows x 5 columns]
```

```
df.shape
```

```
(200, 5)
```

```
df.info
```

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

[200 rows x 5 columns]>
```

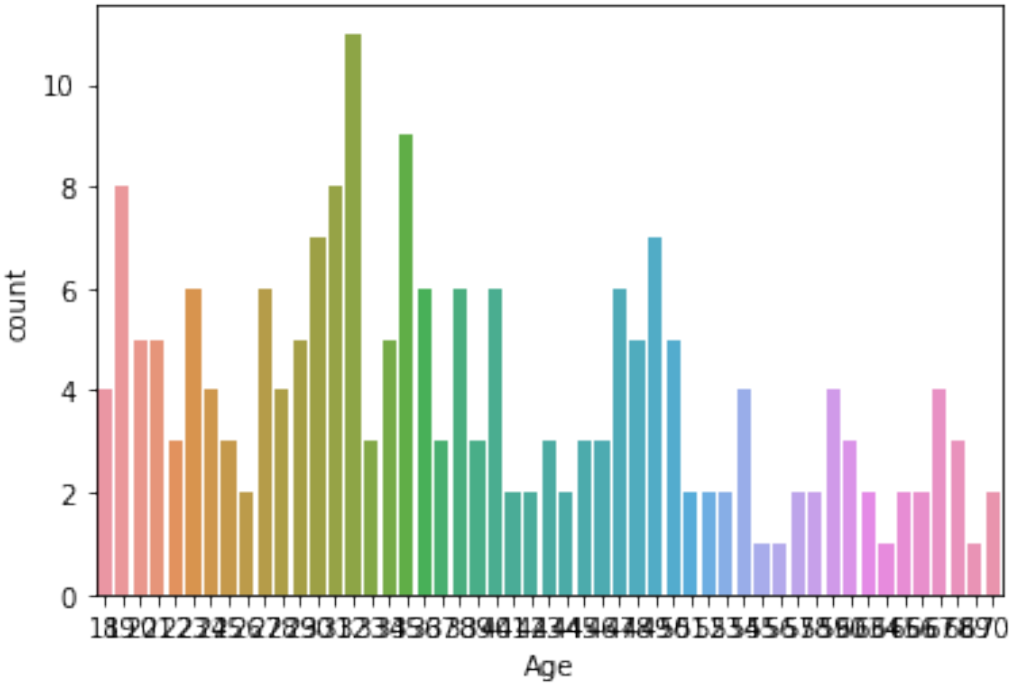
Perform Below Visualizations.

```
#Univariate Analysis
sns.countplot(df['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
```

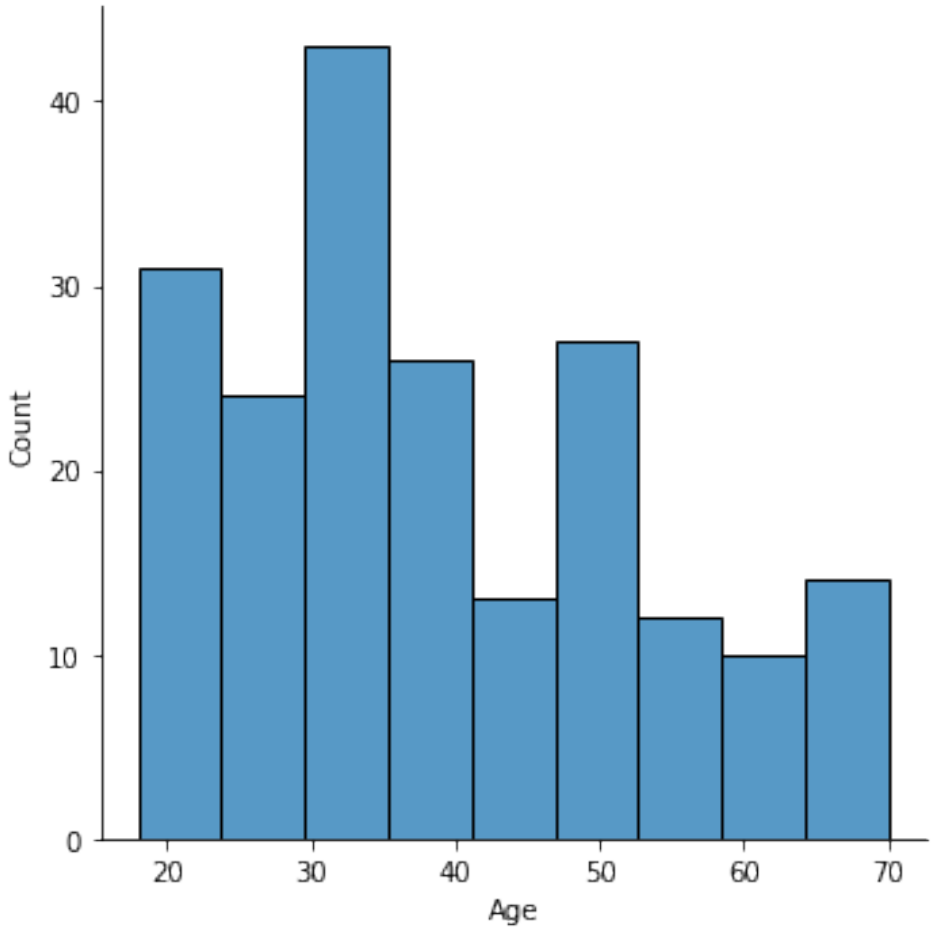
```
error or misinterpretation.
  FutureWarning
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f4c077418d0>
```



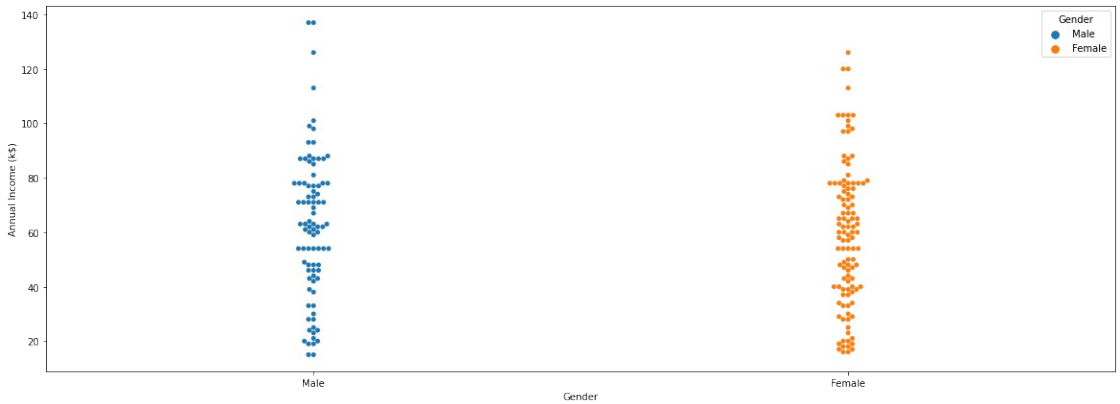

```
sns.displot(df.Age)
```

```
<seaborn.axisgrid.FacetGrid at 0x7f4c0789a090>
```



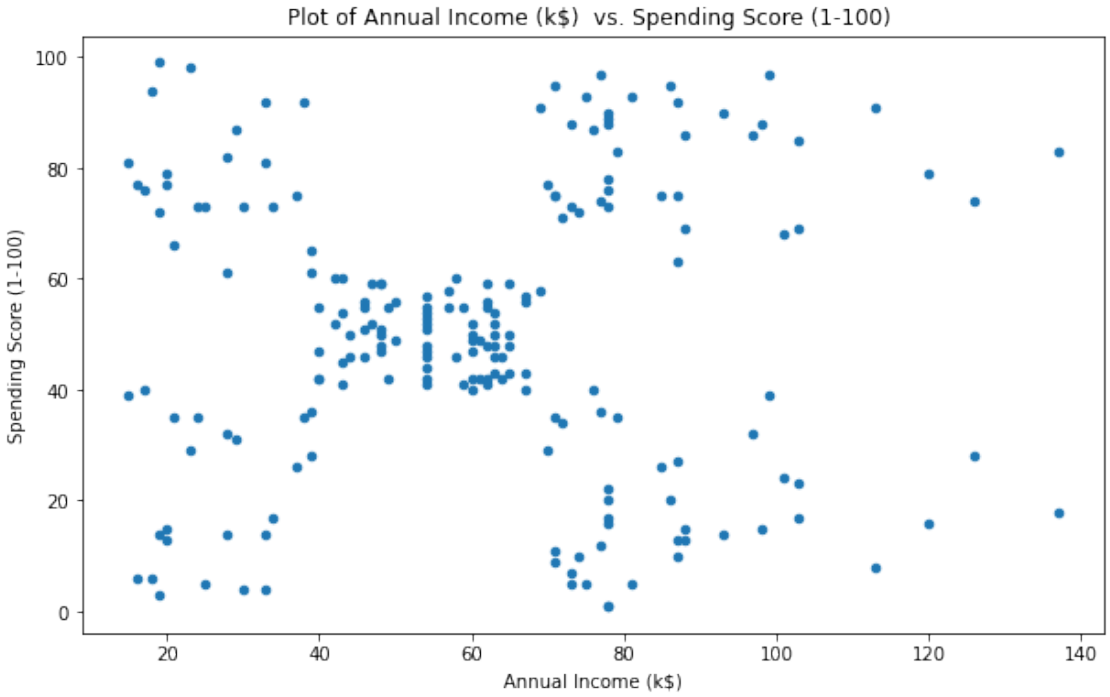

```
#Bivariate Analysis
plt.figure(figsize = (20,7))
sns.swarmplot(x = 'Gender', y = 'Annual Income (k$)', data = df, hue =
'Gender')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f4c075a0e10>
```




```
df.plot.scatter("Annual Income (k$)", "Spending Score (1-100)",
figsize=(10, 6),
```

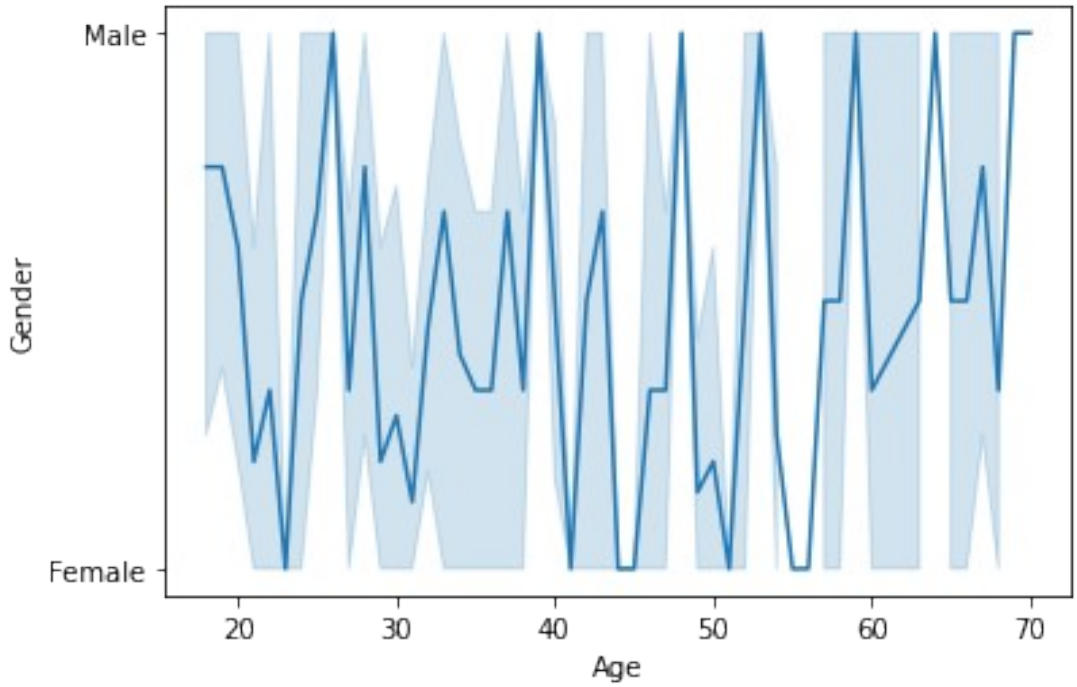
```
                title="Plot of Annual Income (k$)  vs. Spending Score
(1-100) ")
plt.show()
plt.close()
```




```
sns.lineplot(df.Age,df.Gender)
```

```
/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
```

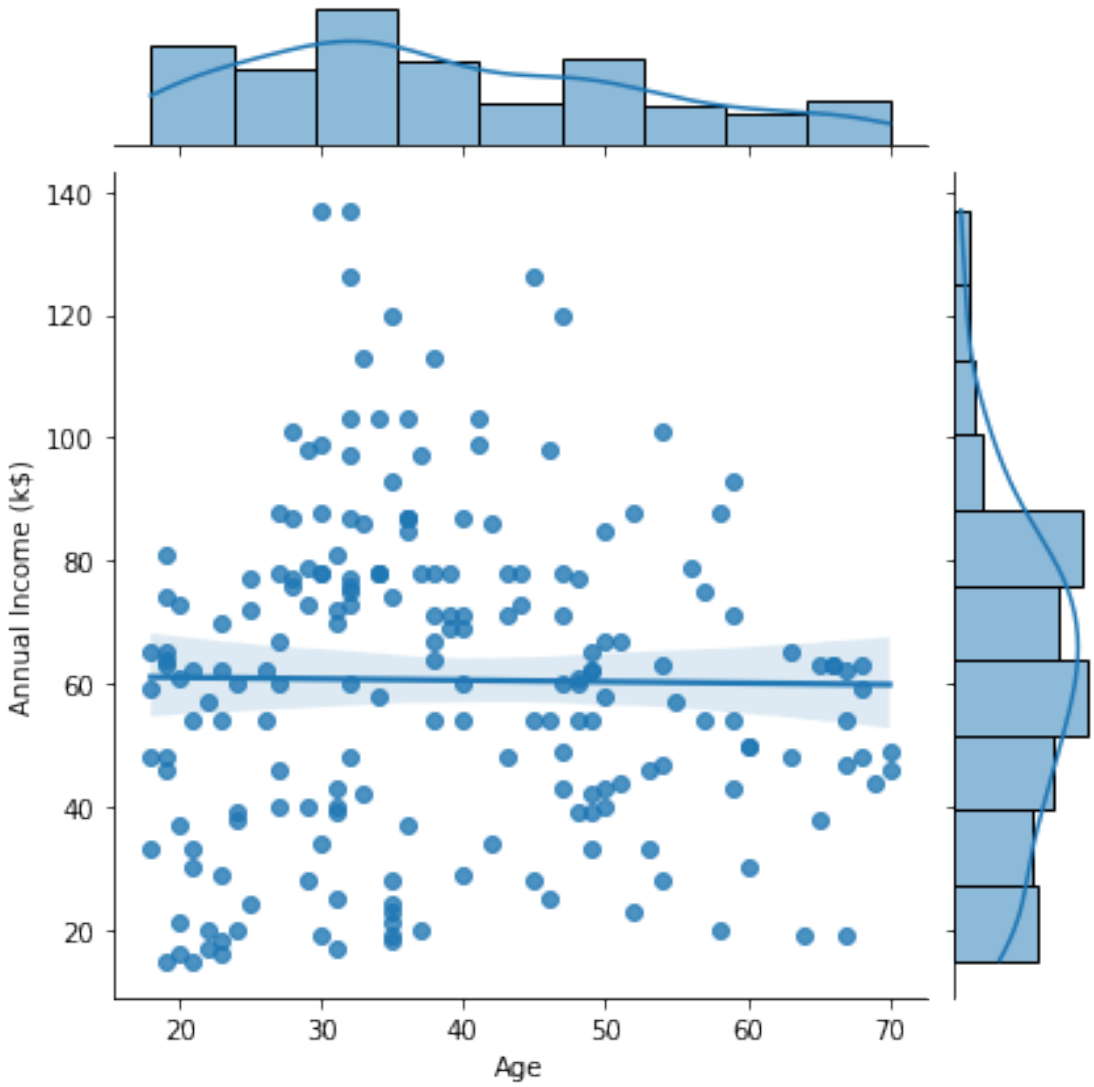
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f4c066fffd0>
```

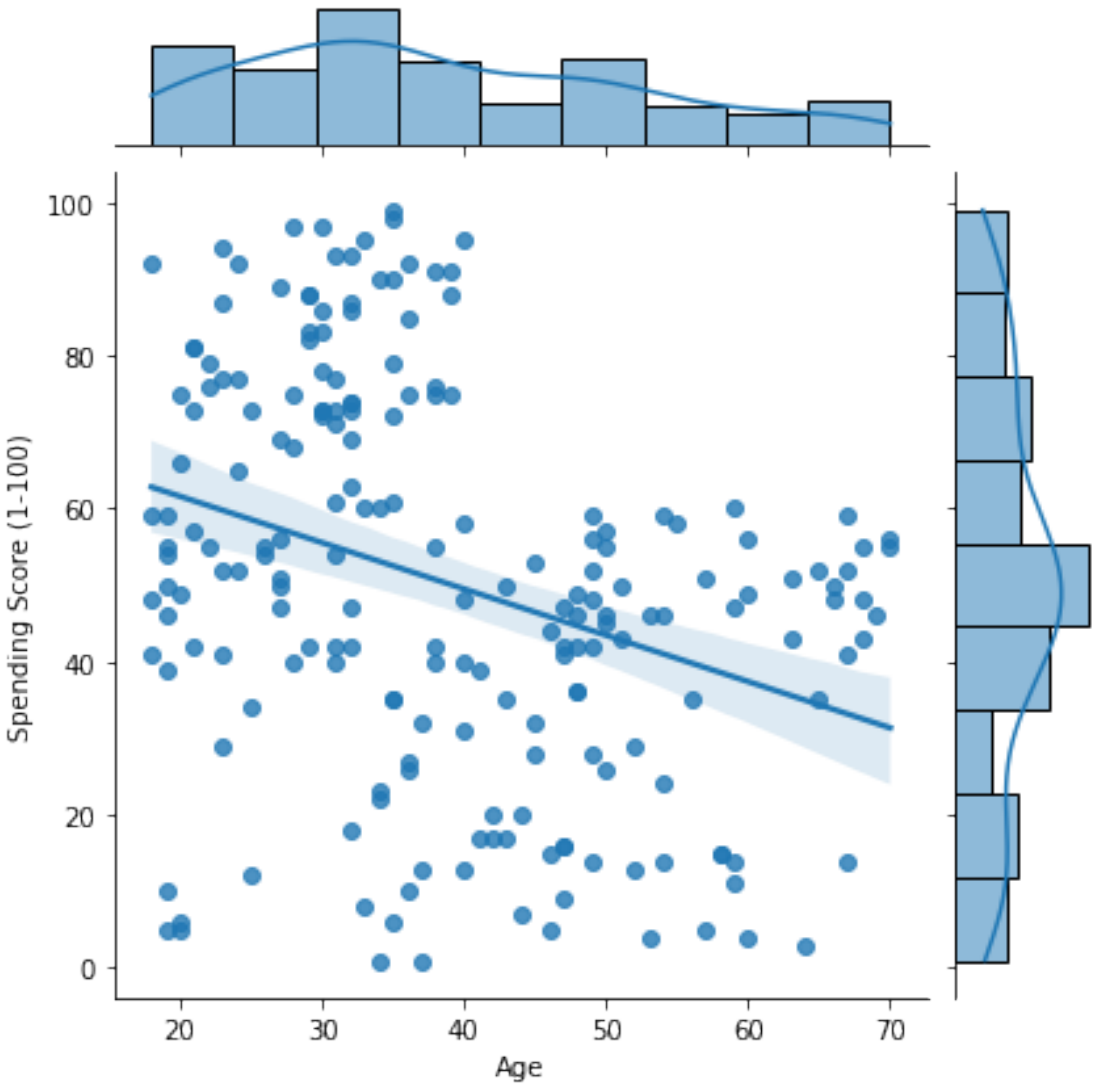



```
plt.figure(figsize=(20, 5))

_ = sns.jointplot(data=df, x='Age', y='Annual Income (k$)',
kind='reg')
_ = sns.jointplot(data=df, x='Age', y='Spending Score (1-100)',
kind='reg')
```

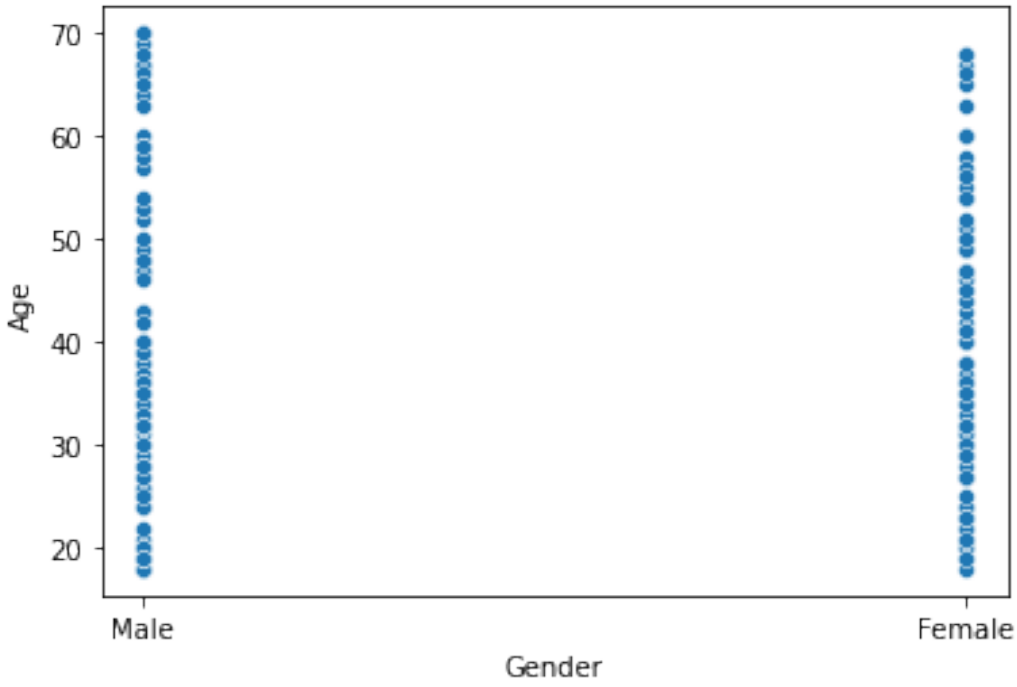
```
<Figure size 1440x360 with 0 Axes>
```


Annual Income (k$)
Age
140
120
100
80
60
40
20
20
30
40
50
60
70



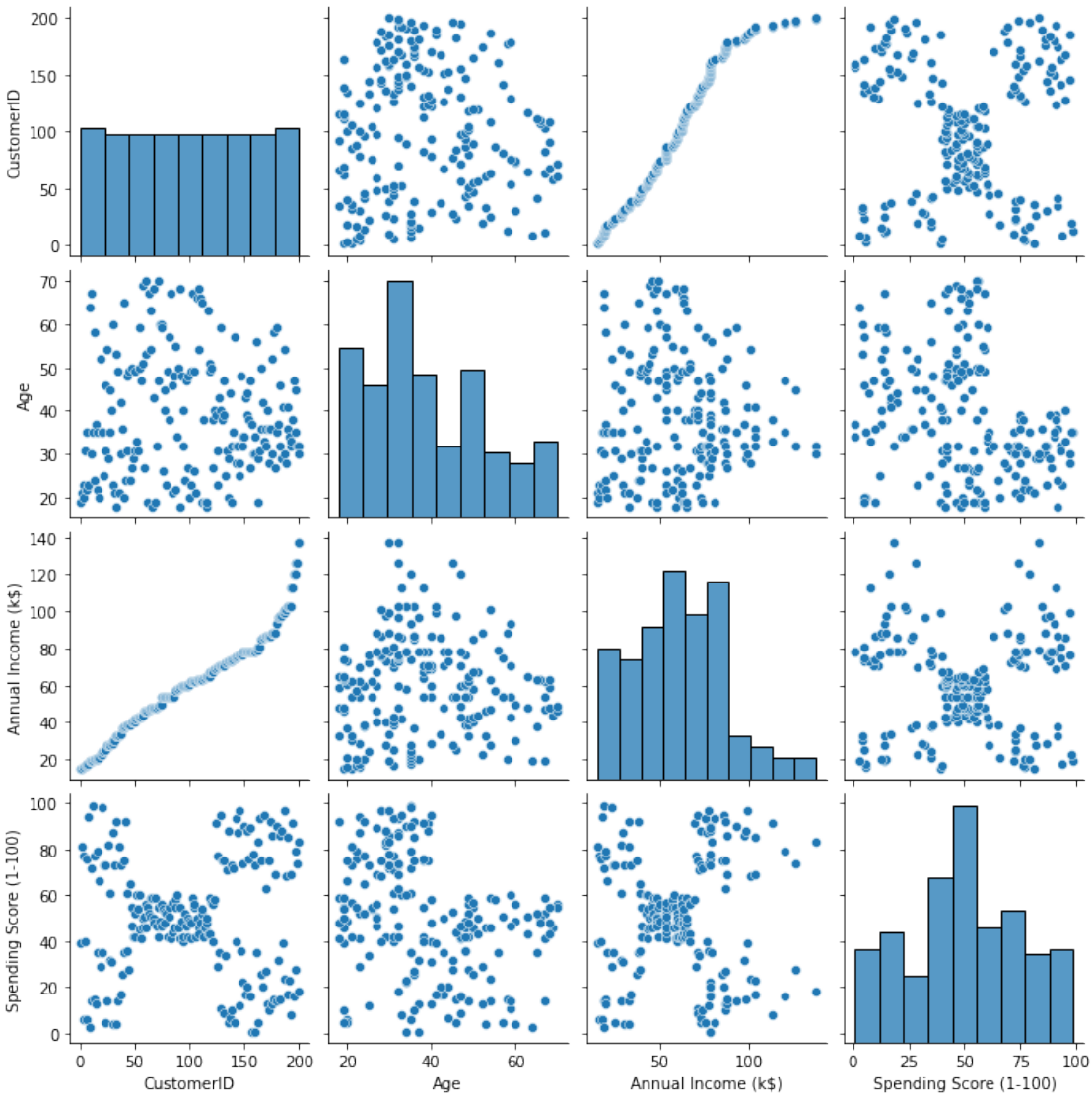

```
sns.scatterplot(x=df["Gender"],y=df["Age"])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f4c0cbe5090>
```




```
#Multivariate Analysis
sns.pairplot(df)
```

```
<seaborn.axisgrid.PairGrid at 0x7f4c0cd81710>
```



Perform descriptive statistics on the dataset.

```
df.describe()
```

```
       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
```

Check for Missing values and deal with them

```
df.isnull().sum().sort_values(ascending=False)
```

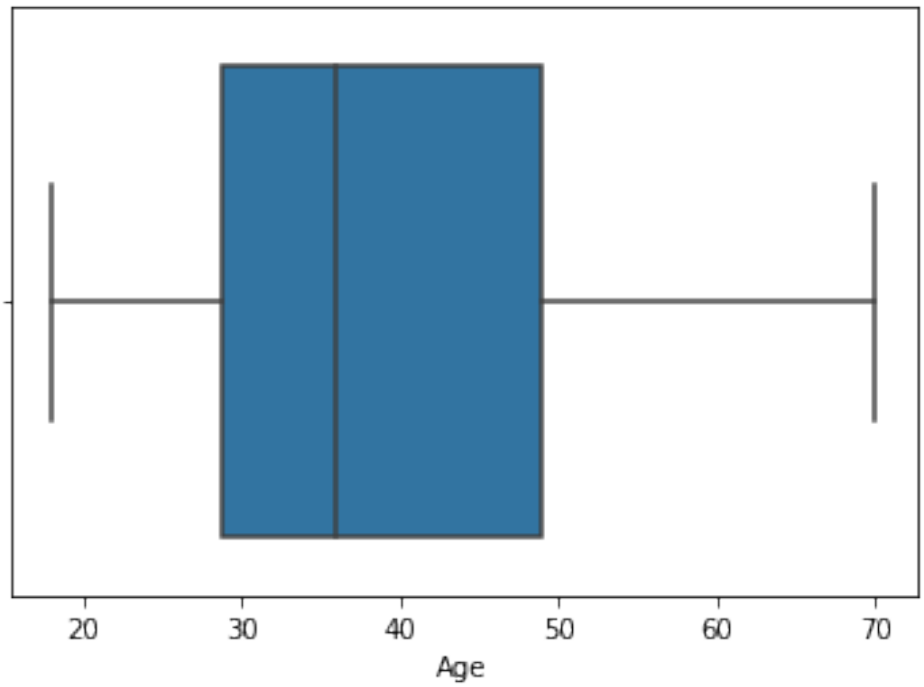
```
CustomerID                0
Gender                    0
Age                       0
Annual Income (k$)        0
Spending Score (1-100)    0
dtype: int64
```

Find the outliers and replace them outliers

```
sns.boxplot(df.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
```

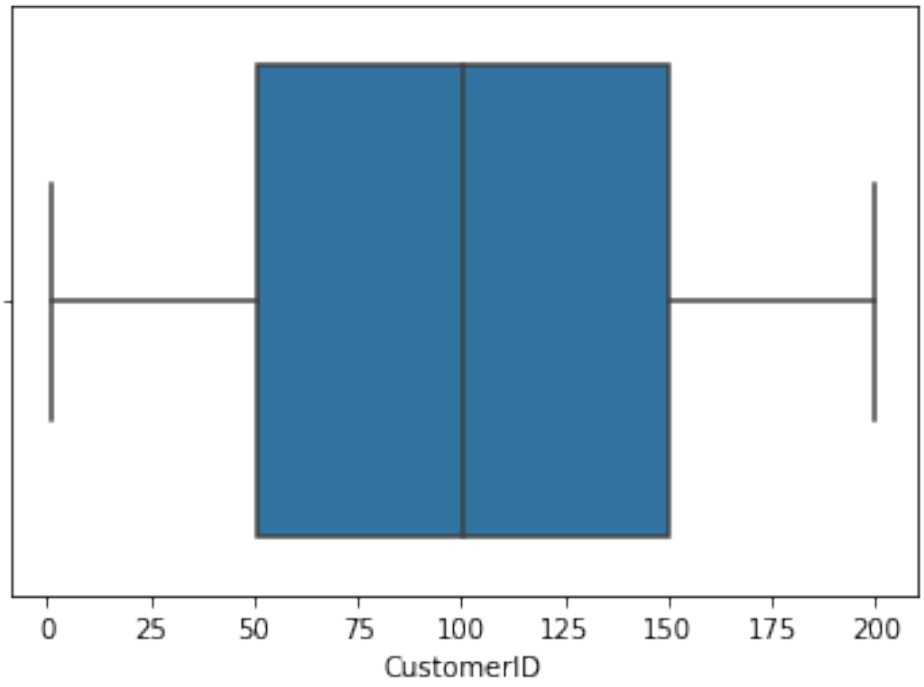
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f4c0adef750>
```



```
sns.boxplot(df.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
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f4c0ad3dc90>
```



Check for Categorical columns and perform encoding.

```
df=pd.get_dummies(df,columns=['CustomerID'])
df.head()
```

```
   Gender  Age  Annual Income (k$)  Spending Score (1-100)
CustomerID_1  \
0    Male   19                  15                      39
1
1    Male   21                  15                      81
0
2  Female   20                  16                       6
0
3  Female   23                  16                      77
0
4  Female   31                  17                      40
0

   CustomerID_2  CustomerID_3  CustomerID_4  CustomerID_5
```

```
CustomerID_6  ...  \
0             0             0             0             0
0  ...
1             1             0             0             0
0  ...
2             0             1             0             0
0  ...
3             0             0             1             0
0  ...
4             0             0             0             1
0  ...

   CustomerID_191  CustomerID_192  CustomerID_193  CustomerID_194  \
0               0               0               0               0
1               0               0               0               0
2               0               0               0               0
3               0               0               0               0
4               0               0               0               0

   CustomerID_195  CustomerID_196  CustomerID_197  CustomerID_198  \
0               0               0               0               0
1               0               0               0               0
2               0               0               0               0
3               0               0               0               0
4               0               0               0               0

   CustomerID_199  CustomerID_200
0               0               0
1               0               0
2               0               0
3               0               0
4               0               0

[5 rows x 204 columns]
```

Scaling the data

```
from sklearn.preprocessing import scale

x_scaled=pd.DataFrame(scale(X),columns=X.columns)
x_scaled.head()
```

```
        Age  Annual Income (k$)  Spending Score (1-100)  CustomerID_1
\
0 -1.424569           -1.738999               -0.434801     14.106736

1 -1.281035           -1.738999                1.195704     -0.070888

2 -1.352802           -1.700830               -1.715913     -0.070888
```

```
3 -1.137502           -1.700830                1.040418     -0.070888

4 -0.563369           -1.662660               -0.395980     -0.070888


   CustomerID_2  CustomerID_3  CustomerID_4  CustomerID_5
CustomerID_6  \
0     -0.070888     -0.070888     -0.070888     -0.070888     -
0.070888
1     14.106736     -0.070888     -0.070888     -0.070888     -
0.070888
2     -0.070888     14.106736     -0.070888     -0.070888     -
0.070888
3     -0.070888     -0.070888     14.106736     -0.070888     -
0.070888
4     -0.070888     -0.070888     -0.070888     14.106736     -
0.070888

   CustomerID_7  ...  CustomerID_191  CustomerID_192
CustomerID_193  \
0     -0.070888  ...       -0.070888       -0.070888       -0.070888

1     -0.070888  ...       -0.070888       -0.070888       -0.070888

2     -0.070888  ...       -0.070888       -0.070888       -0.070888

3     -0.070888  ...       -0.070888       -0.070888       -0.070888

4     -0.070888  ...       -0.070888       -0.070888       -0.070888


   CustomerID_194  CustomerID_195  CustomerID_196  CustomerID_197  \
0       -0.070888       -0.070888       -0.070888       -0.070888
1       -0.070888       -0.070888       -0.070888       -0.070888
2       -0.070888       -0.070888       -0.070888       -0.070888
3       -0.070888       -0.070888       -0.070888       -0.070888
4       -0.070888       -0.070888       -0.070888       -0.070888

   CustomerID_198  CustomerID_199  CustomerID_200
0       -0.070888       -0.070888       -0.070888
1       -0.070888       -0.070888       -0.070888
2       -0.070888       -0.070888       -0.070888
3       -0.070888       -0.070888       -0.070888
4       -0.070888       -0.070888       -0.070888

[5 rows x 203 columns]
```

Perform any of the clustering algorithms and Add the cluster data with the primary dataset

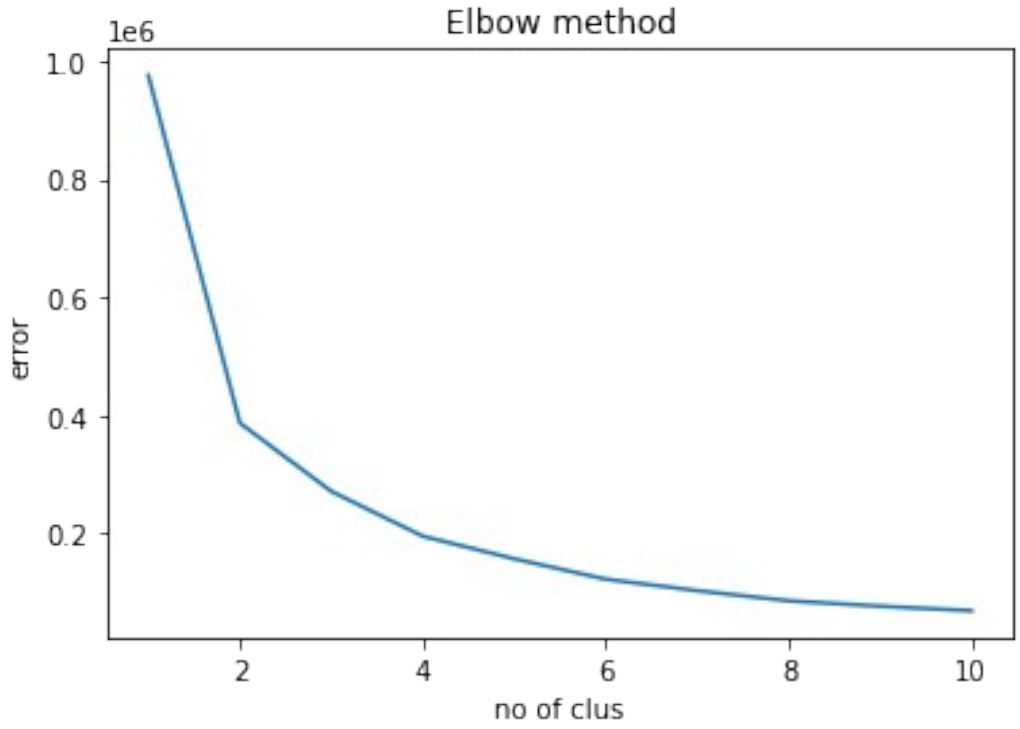
```
from sklearn import cluster

error=[]
for i in range(1,11):
  kmeans=cluster.KMeans(n_clusters=i,init='k-means++',random_state=0)
  kmeans.fit(df2)
  error.append(kmeans.inertia_)


error
```

```
[975512.0600000003,
 387065.71377137717,
 271384.508782868,
 195401.19855991466,
 157157.7579059829,
 122625.19813553878,
 103233.01724386725,
 86053.67444777445,
 76938.97565600359,
 69231.3360761156]
```

```
import matplotlib.pyplot as plt
plt.plot(range(1,11),error)
plt.title('Elbow method')
plt.xlabel('no of clus')
plt.ylabel('error')
plt.show()
```



```
km_model=cluster.KMeans(n_clusters=i,init='k-means++',random_state=0)

km_model.fit(df2)

KMeans(n_clusters=10, random_state=0)

km_model.predict(df2)

array([5, 9, 5, 9, 5, 9, 5, 9, 5, 9, 5, 9, 5, 9, 5, 9, 5, 9, 5,
9,
       5, 9, 5, 9, 5, 9, 5, 9, 5, 9, 5, 9, 5, 9, 5, 9, 5, 9, 5,
1,
       5, 1, 3, 1, 1, 1, 3, 1, 1, 3, 3, 3, 3, 3, 1, 3, 3, 1, 3, 3, 3,
1,
       3, 3, 1, 1, 3, 3, 3, 3, 3, 1, 3, 3, 4, 3, 3, 4, 3, 3, 4, 3, 7,
4,
       4, 7, 7, 4, 7, 4, 4, 4, 7, 4, 7, 4, 4, 7, 7, 4, 7, 4, 7, 7, 7,
7,
       7, 4, 7, 4, 4, 4, 7, 7, 7, 7, 4, 7, 7, 8, 0, 8, 0, 8, 0, 8, 0,
8,
       0, 8, 0, 8, 0, 8, 0, 8, 0, 8, 0, 8, 0, 8, 0, 8, 0, 8, 0, 8, 0,
8,
       0, 8, 0, 8, 0, 8, 0, 8, 0, 8, 6, 2, 6, 2, 6, 2, 6, 2, 6, 2, 6,
2,
       6, 2, 6, 2, 6, 2, 6, 2, 6, 2, 6, 2, 6, 2, 6, 2, 6, 2, 6, 2, 6,
2,
       6, 2], dtype=int32)

km_model.predict([[19,21,20,23,31]])

/usr/local/lib/python3.7/dist-packages/sklearn/base.py:451:
UserWarning: X does not have valid feature names, but KMeans was
fitted with feature names
  "X does not have valid feature names, but"

array([5], dtype=int32)
```

Split the data into dependent and independent variables.

```
y=df['Age']
y

0      19
1      21
2      20
3      23
4      31
       ..
195    35
196    45
197    32
198    32
```

```
199    30
Name: Age, Length: 200, dtype: int64
```

```
X=df.drop(columns=['Gender'],axis=1)
X.head()
```

```
   Age  Annual Income (k$)  Spending Score (1-100)  CustomerID_1  \
0   19                  15                      39             1
1   21                  15                      81             0
2   20                  16                       6             0
3   23                  16                      77             0
4   31                  17                      40             0

   CustomerID_2  CustomerID_3  CustomerID_4  CustomerID_5
CustomerID_6  \
0             0             0             0             0
0
1             1             0             0             0
0
2             0             1             0             0
0
3             0             0             1             0
0
4             0             0             0             1
0

   CustomerID_7  ...  CustomerID_191  CustomerID_192
CustomerID_193  \
0             0  ...               0               0               0

1             0  ...               0               0               0

2             0  ...               0               0               0

3             0  ...               0               0               0

4             0  ...               0               0               0


   CustomerID_194  CustomerID_195  CustomerID_196  CustomerID_197  \
0               0               0               0               0
1               0               0               0               0
2               0               0               0               0
3               0               0               0               0
4               0               0               0               0

   CustomerID_198  CustomerID_199  CustomerID_200
0               0               0               0
1               0               0               0
2               0               0               0
```

```
3               0               0               0
4               0               0               0

[5 rows x 203 columns]
```

Split the data into training and testing

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test =
train_test_split(x_scaled,y,test_size=0.2,random_state=0)
```

Build the Model

```
from sklearn.linear_model import Ridge
from sklearn.linear_model import Lasso

r=Ridge()
l=Lasso()

r.fit(X_train,y_train)

Ridge()

l.fit(X_train,y_train)

Lasso()
```

Train the model

```
pred1=r.predict(X_test)
pred1

array([45.40190903, 40.56733809, 45.56811187, 43.06755995,
32.70560084,
       43.12534643, 30.37011146, 43.26168608, 48.58966798,
42.09175307,
       29.77544482, 34.21018815, 38.50510145, 43.56977944,
46.72528056,
       30.13331981, 27.96638989, 43.84702599, 33.98940373,
47.76080314,
       32.85588314, 32.11625646, 31.62552153, 33.90237619,
52.30550922,
       37.22539382, 35.44553785, 32.95775026, 38.90964643,
44.08038004,
       37.65752793, 42.69828709, 41.76213554, 29.7794493 ,
43.64602056,
       36.05640641, 37.38099427, 35.70728062, 42.12114016,
50.7276423 ])

pred1_train=r.predict(X_train)

pred2=l.predict(X_test)
pred2
```

```
array([51.0750721 , 39.90569023, 52.93663574, 47.35194481,
27.80552655,
       45.49038116, 23.15161744, 47.35194481, 56.65976303,
43.62881752,
       20.35927197, 31.52865384, 37.11334477, 46.42116299,
55.7289812 ,
       24.08239926, 19.42849015, 46.42116299, 30.59787201,
57.59054485,
       29.66709019, 28.73630837, 25.01318108, 30.59787201,
67.82914489,
       38.97490841, 35.25178112, 30.59787201, 39.90569023,
48.28272663,
       35.25178112, 46.42116299, 42.6980357 , 20.35927197,
45.49038116,
       36.18256294, 38.04412659, 31.52865384, 45.49038116,
64.1060176 ])

pred2_train=l.predict(X_train)

profit=pd.DataFrame({'Actual':y_test,'ridge_pred':pred1,'lasso_pred':p
red2})
profit.head(11)
```

	Actual	ridge_pred	lasso_pred
18	52	45.401909	51.075072
170	40	40.567338	39.905690
107	54	45.568112	52.936636
98	48	43.067560	47.351945
177	27	32.705601	27.805527
182	46	43.125346	45.490381
5	22	30.370111	23.151617
146	48	43.261686	47.351945
12	58	48.589668	56.659763
152	44	42.091753	43.628818
61	19	29.775445	20.359272

Test the model

```
p=r.predict([[5, 9, 5, 9, 5, 9, 5, 9, 5, 9, 5, 9, 5, 9, 5, 9, 5, 9, 5,
9, 5, 9,
       5, 9, 5, 9, 5, 9, 5, 9, 5, 9, 5, 9, 5, 9, 5, 9, 5, 9, 5, 9, 5,
1,
       5, 1, 3, 1, 1, 1, 3, 1, 1, 3, 3, 3, 3, 3, 1, 3, 3, 1, 3, 3, 3,
1,
       3, 3, 1, 1, 3, 3, 3, 3, 3, 1, 3, 3, 4, 3, 3, 4, 3, 3, 4, 3, 7,
4,
       4, 7, 7, 4, 7, 4, 4, 4, 7, 4, 7, 4, 4, 7, 7, 4, 7, 4, 7, 7, 7,
7,
       7, 4, 7, 4, 4, 4, 7, 7, 7, 7, 4, 7, 7, 8, 0, 8, 0, 8, 0, 8, 0,
8,
       0, 8, 0, 8, 0, 8, 0, 8, 0, 8, 0, 8, 0, 8, 0, 8, 0, 8, 0, 8, 0,
```

```
8,
       0, 8, 0, 8, 0, 8, 0, 8, 0, 8, 6, 2, 6, 2, 6, 2, 6, 2, 6,
2,
       6, 2, 6, 2, 6, 2, 6, 2, 6, 2, 6, 2, 6, 2, 6, 2, 6, 2, 6,
2,
       6, 2,4,6,7]])
print(p)
p1=l.predict([[5, 9, 5, 9, 5, 9, 5, 9, 5, 9, 5, 9, 5, 9, 5, 9, 5, 9,
5, 9, 5, 9,
       5, 9, 5, 9, 5, 9, 5, 9, 5, 9, 5, 9, 5, 9, 5, 9, 5, 9, 5, 9, 5,
1,
       5, 1, 3, 1, 1, 1, 3, 1, 1, 3, 3, 3, 3, 3, 1, 3, 3, 1, 3, 3, 3,
1,
       3, 3, 1, 1, 3, 3, 3, 3, 3, 1, 3, 3, 4, 3, 3, 4, 3, 3, 4, 3, 7,
4,
       4, 7, 7, 4, 7, 4, 4, 4, 7, 4, 7, 4, 4, 7, 7, 4, 7, 4, 7, 7, 7,
7,
       7, 4, 7, 4, 4, 4, 7, 7, 7, 7, 4, 7, 7, 8, 0, 8, 0, 8, 0, 8, 0,
8,
       0, 8, 0, 8, 0, 8, 0, 8, 0, 8, 0, 8, 0, 8, 0, 8, 0, 8, 0, 8, 0,
8,
       0, 8, 0, 8, 0, 8, 0, 8, 0, 8, 6, 2, 6, 2, 6, 2, 6, 2, 6, 2, 6,
2,
       6, 2, 6, 2, 6, 2, 6, 2, 6, 2, 6, 2, 6, 2, 6, 2, 6, 2, 6, 2, 6,
2,
       6, 2,4,6,7]])
print(p1)
```

```
[34.6929234]
[103.68305171]
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/base.py:451:
UserWarning: X does not have valid feature names, but Ridge was fitted
with feature names
  "X does not have valid feature names, but"
/usr/local/lib/python3.7/dist-packages/sklearn/base.py:451:
UserWarning: X does not have valid feature names, but Lasso was fitted
with feature names
  "X does not have valid feature names, but"
```

Measure the performance using Evaluation Metrics

```
from sklearn import metrics

# R-Square
# testing accuracy for both model
print(metrics.r2_score(y_test,pred1))
print(metrics.r2_score(y_test,pred2))
```

```
0.728431906852201
0.995175995732047
```

```
#Training accuracy for both model
```

```
print(metrics.r2_score(y_train,pred1_train))
print(metrics.r2_score(y_train,pred2_train))
```

```
0.9999927831781418
0.9952088438199637
```

```
## MSE(Mean square error)
```

```
print(metrics.mean_squared_error(y_test,pred1))
print(metrics.mean_squared_error(y_test,pred2))
```

```
44.71640221771658
0.7943205427611527
```

```
## RMSE
```

```
print(np.sqrt(metrics.mean_squared_error(y_test,pred1)))
print(np.sqrt(metrics.mean_squared_error(y_test,pred2)))
```

```
6.687032392453066
0.8912466228610085
```