Importing required libraries

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

1.Dataset was Downloaded

2.Loading the Dataset

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

```
     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 x 5 columns]
```

```
df.shape
```

```
(200, 5)
```

```
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
```

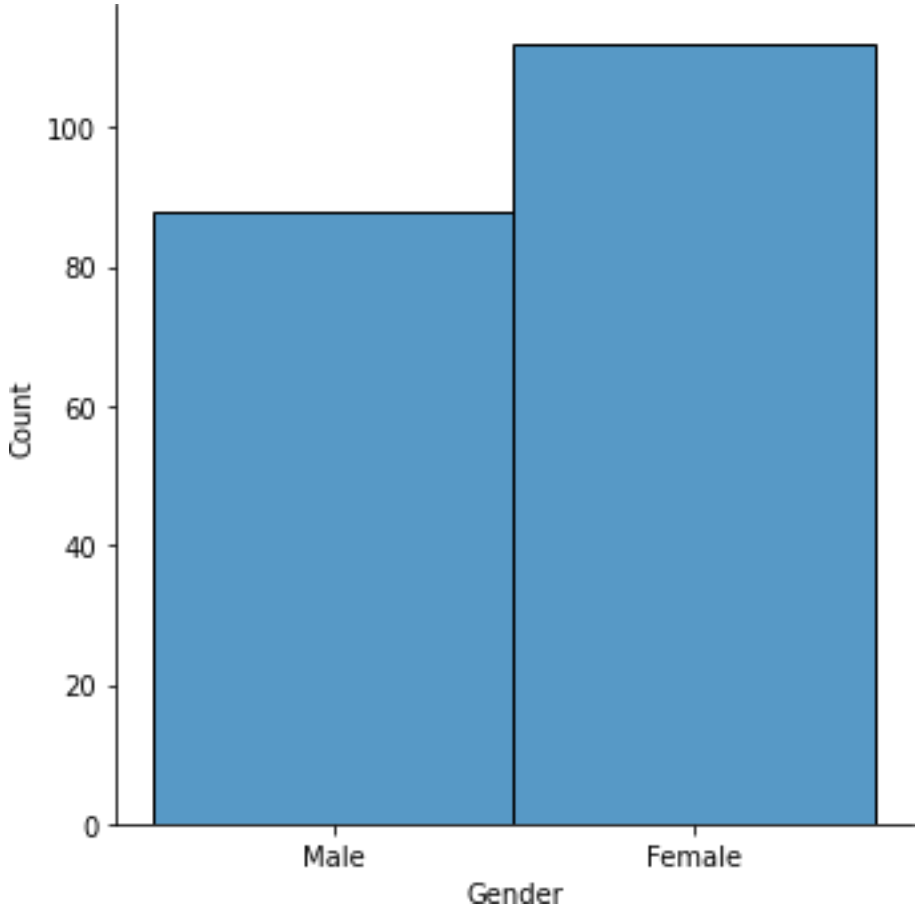
```
3           4  Female   23                  16                  77
4           5  Female   31                  17                  40
```

3. Performing Visualizations

Univariate Analysis

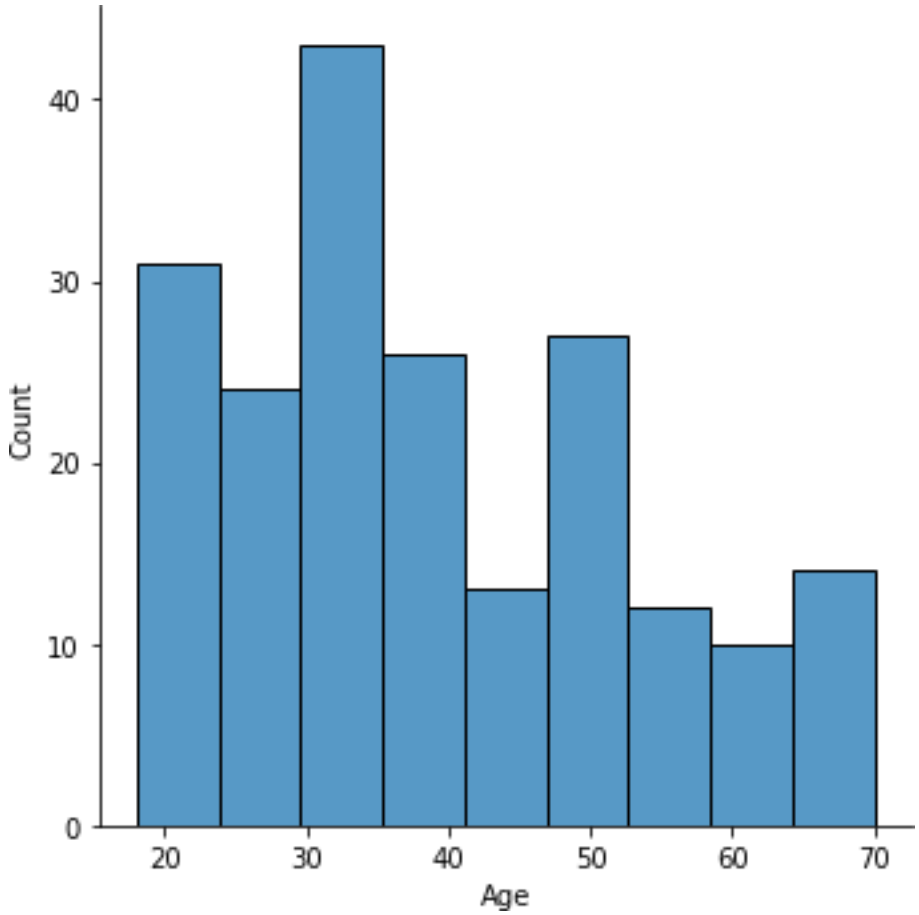
```
sns.displot(df.Gender)
```

```
<seaborn.axisgrid.FacetGrid at 0x2a0d31ee3a0>
```




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

```
<seaborn.axisgrid.FacetGrid at 0x2a0d534e7c0>
```

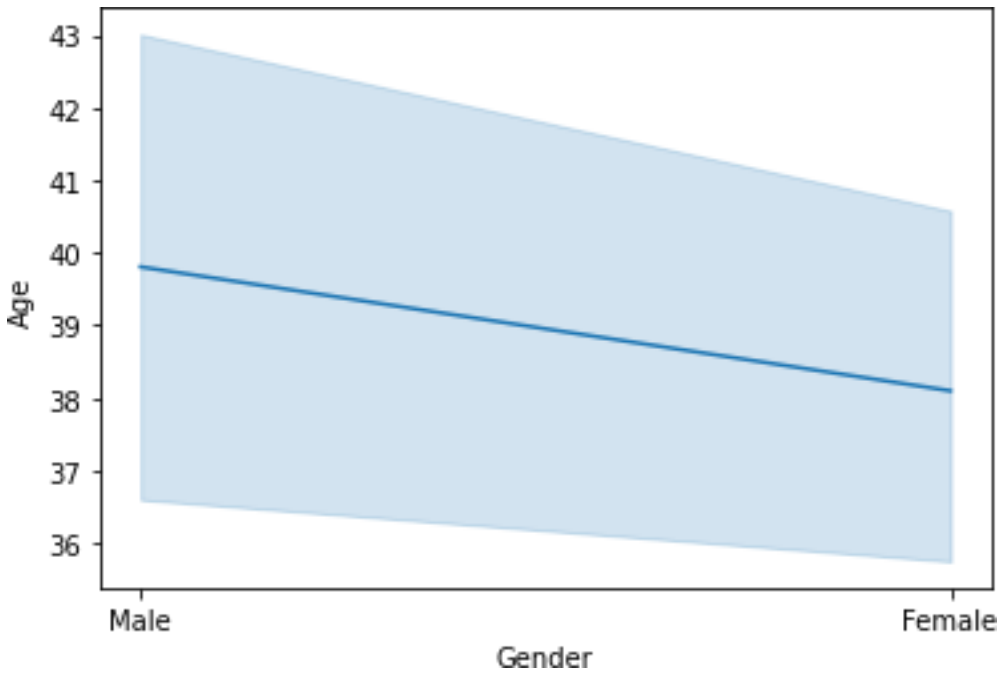


Bi-Variate Analysis

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

```
C:\Users\AMMU\anaconda3\New folder\lib\site-packages\seaborn\
_decorators.py:36: 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.
  warnings.warn(
```

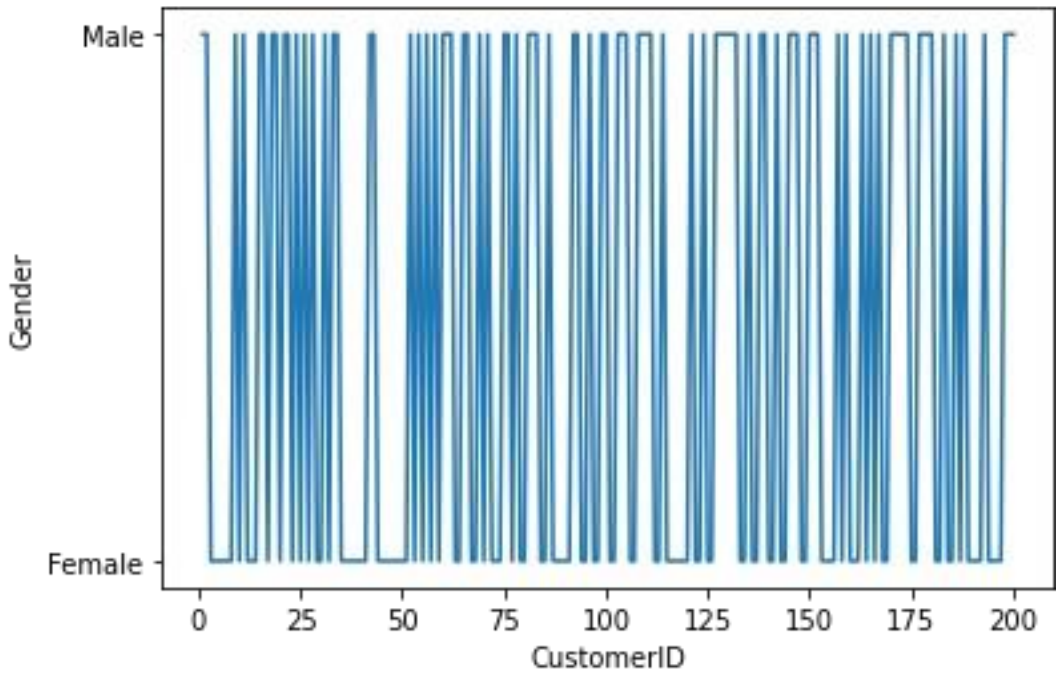
```
<AxesSubplot:xlabel='Gender', ylabel='Age'>
```




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

```
C:\Users\AMMU\anaconda3\New folder\lib\site-packages\seaborn\
_decorators.py:36: 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.
  warnings.warn(
```

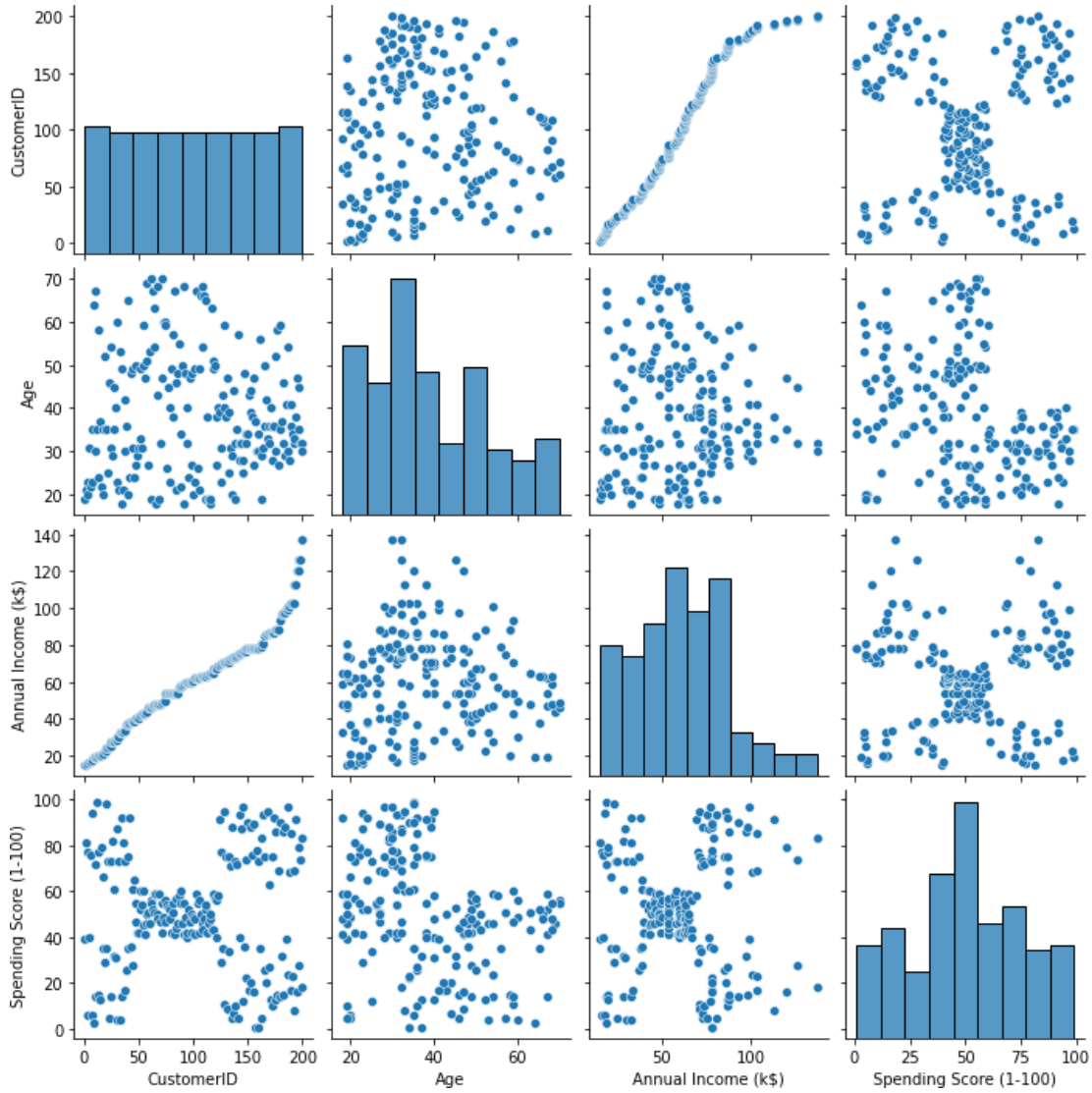
```
<AxesSubplot:xlabel='CustomerID', ylabel='Gender'>
```



Multi-Variate Analysis

```
sns.pairplot(df)
```

```
<seaborn.axisgrid.PairGrid at 0x2a0d55326a0>
```



4. Descriptive Statistics

```
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
```

5. Finding Missing Values And Replacing It.

```
df.isnull().any()
```

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

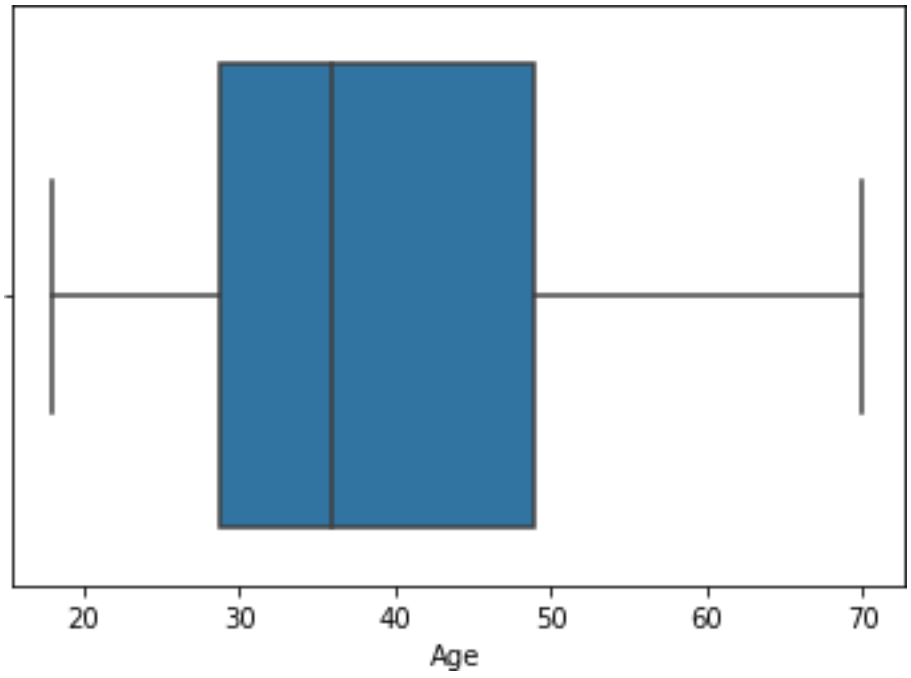
There is no any null values in the dataset

6. Finding Outliers And Replacing them.

```
sns.boxplot(df.Age)
```

```
C:\Users\AMMU\anaconda3\New folder\lib\site-packages\seaborn\
_decorators.py:36: 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.
  warnings.warn(
```

```
<AxesSubplot:xlabel='Age'>
```



```
## There is no outliers present
```

7.Checking Categorical columns and performing encoding.

```
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
```

Label Encoding

```
from sklearn.preprocessing import LabelEncoder

le=LabelEncoder()

df.Gender=le.fit_transform(df.Gender)

df.head()
```

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

8.Scaling the data

```
from sklearn.preprocessing import scale

X=df

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

```
   CustomerID    Gender       Age  Annual Income (k$)  Spending Score
(1-100)
0   -1.723412  1.128152 -1.424569           -1.738999               -
0.434801
1   -1.706091  1.128152 -1.281035           -1.738999
1.195704
2   -1.688771 -0.886405 -1.352802           -1.700830               -
1.715913
3   -1.671450 -0.886405 -1.137502           -1.700830
1.040418
4   -1.654129 -0.886405 -0.563369           -1.662660               -
0.395980
```

9.Performing Clustering Algorithm.

```
from sklearn import cluster
```

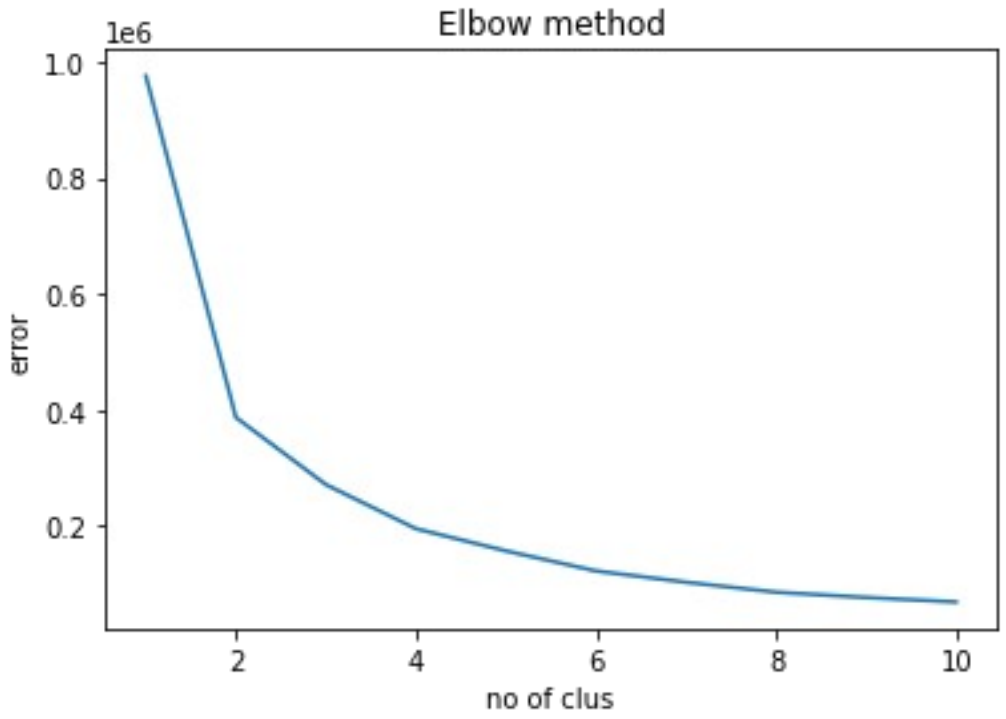
```
error =[]
for i in range(1,11):
    kmeans=cluster.KMeans(n_clusters=i,init='k-means+
+',random_state=0)
    kmeans.fit(df)
    error.append(kmeans.inertia_)
```

```
C:\Users\AMMU\anaconda3\New folder\lib\site-packages\sklearn\cluster\
_kmeans.py:1036: UserWarning: KMeans is known to have a memory leak on
Windows with MKL, when there are less chunks than available threads.
You can avoid it by setting the environment variable
OMP_NUM_THREADS=1.
  warnings.warn(
```

```
error
```

```
[975512.0599999999,
 387065.71377137717,
 271384.508782868,
 195401.19855991466,
 157157.7579059829,
 122625.1981355388,
 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=3,init='k-means++',random_state=0)

km_model.fit(df)

KMeans(n_clusters=3, random_state=0)

ykmeans =km_model.predict(df)
ykmeans

array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1,
1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
2,
       2, 2])
```

```
km_model.predict([[1,1,19,15,39]])
```

```
C:\Users\AMMU\anaconda3\New folder\lib\site-packages\sklearn\
base.py:450: UserWarning: X does not have valid feature names, but
KMeans was fitted with feature names
  warnings.warn(

array([0])
```

10. Adding Cluster Data with Primary Dataset.

```
df['kclus'] = pd.Series(ykmeans)

df.head()
```

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

11. Splitting Data into Dependent And independent Variables.

```
X= df.iloc[:,:-1]
X
```

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

```
198         199       1    32                  137
18
199         200       1    30                  137
83

[200 rows x 5 columns]
```

```
y=df.kclus
```

12. Splitting Data into Training And Testing Data.

```
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.
3,random_state=0)
```

```
X_train.shape
```

```
(140, 5)
```

```
X_test.shape
```

```
(60, 5)
```

```
y_train.shape
```

```
(140,)
```

```
y_test.shape
```

```
(60,)
```

13. Building the Model.

```
from sklearn.neighbors import KNeighborsClassifier
model =KNeighborsClassifier()
```

14. Training the model.

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

```
KNeighborsClassifier()
```

15. Testing the model.

```
pred_test=model.predict(X_test)
```

```
pred_train=model.predict(X_train)
```

16. Evaluating the model using evaluation metrics.

```
from sklearn.metrics import
accuracy_score,classification_report,confusion_matrix
```

```
print('Test accuracy score: ',accuracy_score(y_test,pred_test))
print('Training accuracy score: ',accuracy_score(y_train,pred_train))
```

```
Test accuracy score: 0.8333333333333334
Training accuracy score:  0.9571428571428572

pd.crosstab(y_test,pred_test)

col_0   0   1   2
kclus
0      16   4   0
1       1  15   4
2       0   1  19

print(classification_report(y_test,pred_test))

              precision    recall  f1-score   support

           0       0.94      0.80      0.86        20
           1       0.75      0.75      0.75        20
           2       0.83      0.95      0.88        20

    accuracy                           0.83        60
   macro avg       0.84      0.83      0.83        60
weighted avg       0.84      0.83      0.83        60
```