Customer Segmentation and Analysis

Steps to solve the problem :

1. Importing Libraries.
2. Exploration of data.
3. Data Visualization.
4. Clustering using K-Means.
5. Selection of Clusters.
6. Ploting the Cluster Boundry and Clusters.
7. 3D Plot of Clusters.

Importing Libraries.

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import matplotlib.pyplot as plt
import seaborn as sns
import plotly as py
import plotly.graph_objs as go
from sklearn.cluster import KMeans
import warnings
import os
warnings.filterwarnings("ignore")
py.offline.init_notebook_mode(connected = True)
#print(os.listdir("../input"))
```

Data Exploration

```
df = pd.read_csv(r'../input/Mall_Customers.csv')
df.head()
```

```
   CustomerID           ...            Spending Score (1-100)
0           1           ...                                39
1           2           ...                                81
2           3           ...                                 6
3           4           ...                                77
4           5           ...                                40

[5 rows x 5 columns]
```

```
df.shape
```

```
(200, 5)
```

```
df.describe()
```

```
       CustomerID           ...            Spending Score (1-100)
count  200.000000           ...                        200.000000
mean   100.500000           ...                         50.200000
std     57.879185           ...                         25.823522
min      1.000000           ...                          1.000000
25%     50.750000           ...                         34.750000
50%    100.500000           ...                         50.000000
75%    150.250000           ...                         73.000000
max    200.000000           ...                         99.000000

[8 rows x 4 columns]
```

```
df.dtypes
```

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

```
df.isnull().sum()
```

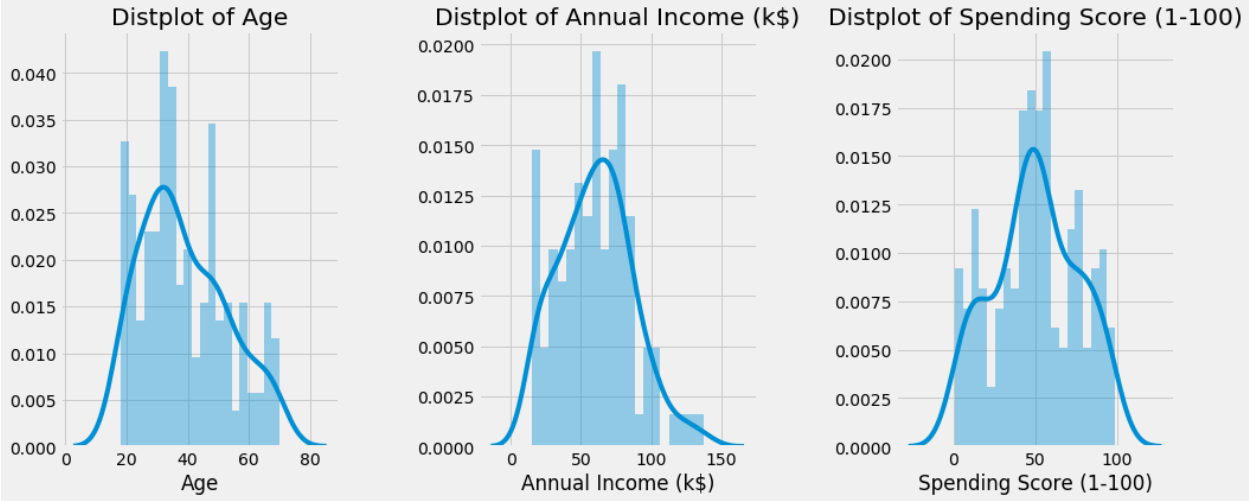
```
CustomerID                  0
Gender                      0
Age                         0
Annual Income (k$)          0
Spending Score (1-100)      0
dtype: int64
```

Data Visualization

```
plt.style.use('fivethirtyeight')
```

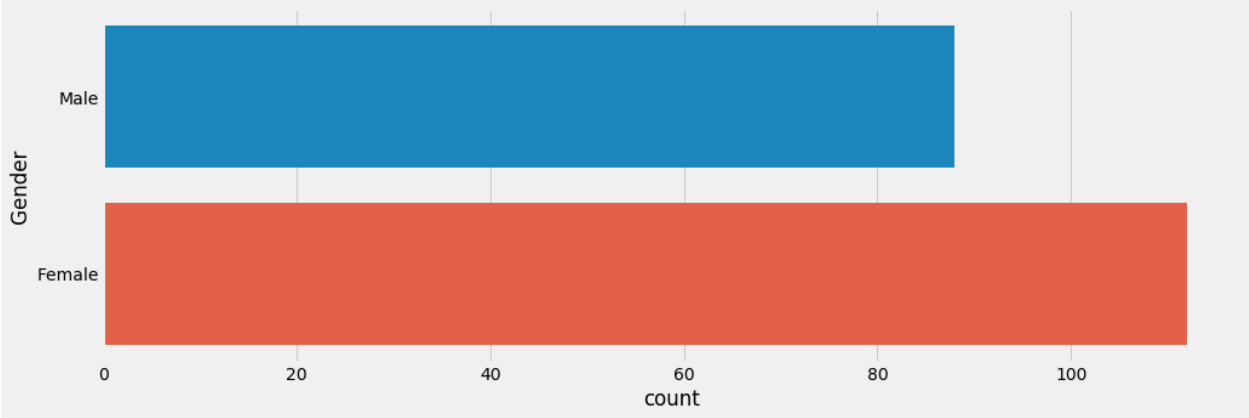
Histograms

```
plt.figure(1 , figsize = (15 , 6))
n = 0
for x in ['Age' , 'Annual Income (k$)' , 'Spending Score (1-100)']:
    n += 1
    plt.subplot(1 , 3 , n)
    plt.subplots_adjust(hspace =0.5 , wspace = 0.5)
    sns.distplot(df[x] , bins = 20)
    plt.title('Distplot of {}'.format(x))
plt.show()
```



Count Plot of Gender

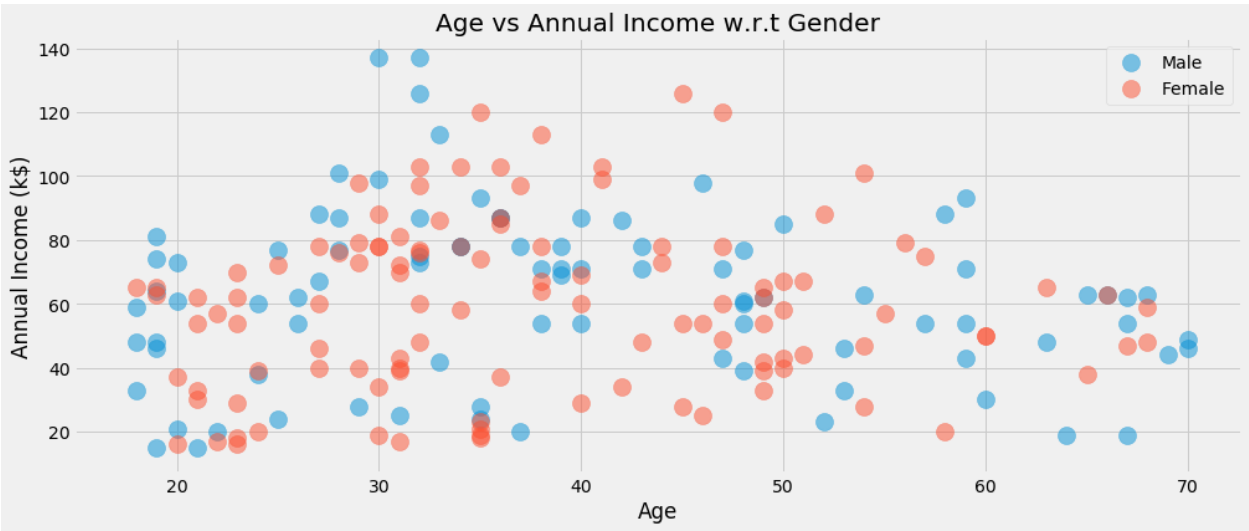
```
plt.figure(1 , figsize = (15 , 5))
sns.countplot(y = 'Gender' , data = df)
plt.show()
```



Ploting the Relation between Age , Annual Income and Spending Score

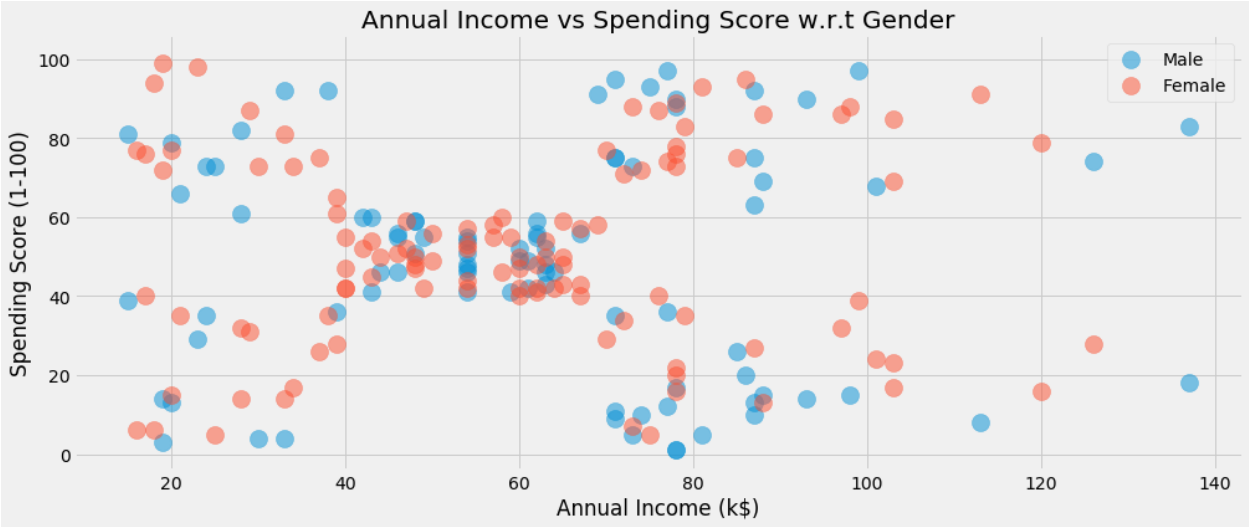
```
plt.figure(1 , figsize = (15 , 7))
n = 0
for x in ['Age' , 'Annual Income (k$)' , 'Spending Score (1-100)']:
    for y in ['Age' , 'Annual Income (k$)' , 'Spending Score (1-
100)']:
        n += 1
        plt.subplot(3 , 3 , n)
        plt.subplots_adjust(hspace = 0.5 , wspace = 0.5)
        sns.regplot(x = x , y = y , data = df)
        plt.ylabel(y.split()[0]+' '+y.split()[1] if len(y.split()) > 1
else y )
plt.show()
```


```
plt.figure(1 , figsize = (15 , 6))
for gender in ['Male' , 'Female']:
    plt.scatter(x = 'Age' , y = 'Annual Income (k$)' , data =
df[df['Gender'] == gender] ,
                s = 200 , alpha = 0.5 , label = gender)
plt.xlabel('Age'), plt.ylabel('Annual Income (k$)')
plt.title('Age vs Annual Income w.r.t Gender')
plt.legend()
plt.show()
```



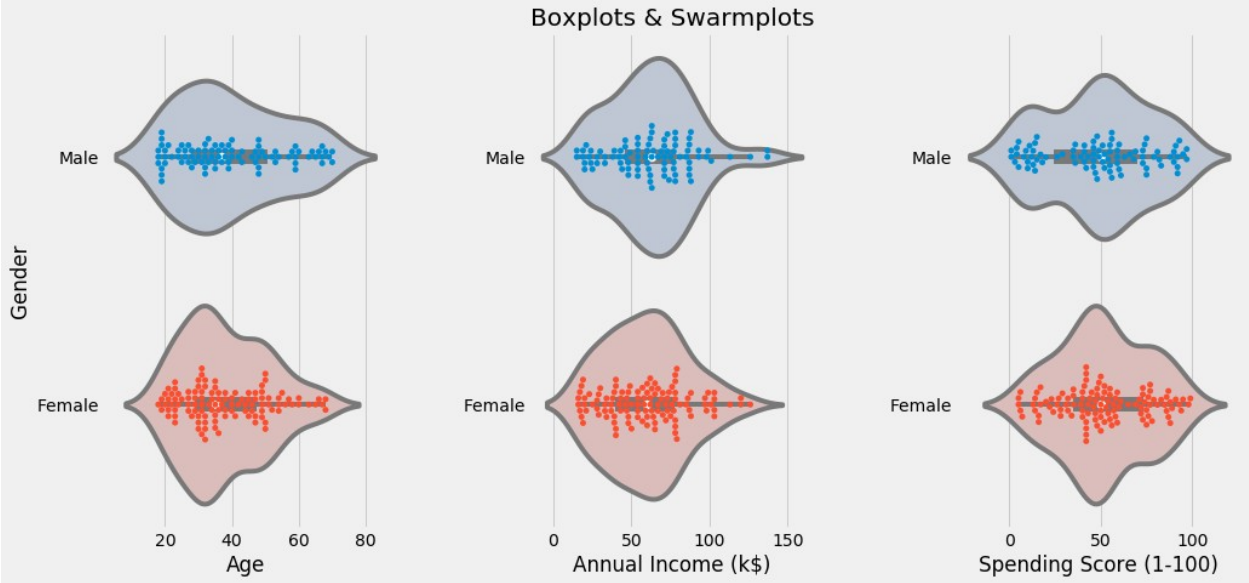

```
plt.figure(1 , figsize = (15 , 6))
for gender in ['Male' , 'Female']:
    plt.scatter(x = 'Annual Income (k$)',y = 'Spending Score (1-
100)' ,
                data = df[df['Gender'] == gender] ,s = 200 , alpha =
```

```
0.5 , label = gender)
plt.xlabel('Annual Income (k$)'), plt.ylabel('Spending Score (1-100)')

plt.title('Annual Income vs Spending Score w.r.t Gender')
plt.legend()
plt.show()
```



Distribution of values in Age , Annual Income and Spending Score according to Gender

```
plt.figure(1 , figsize = (15 , 7))
n = 0
for cols in ['Age' , 'Annual Income (k$)' , 'Spending Score (1-100)']:
    n += 1
    plt.subplot(1 , 3 , n)
    plt.subplots_adjust(hspace = 0.5 , wspace = 0.5)
    sns.violinplot(x = cols , y = 'Gender' , data = df , palette =
'vlag')
    sns.swarmplot(x = cols , y = 'Gender' , data = df)
    plt.ylabel('Gender' if n == 1 else '')
    plt.title('Boxplots & Swarmplots' if n == 2 else '')
plt.show()
```



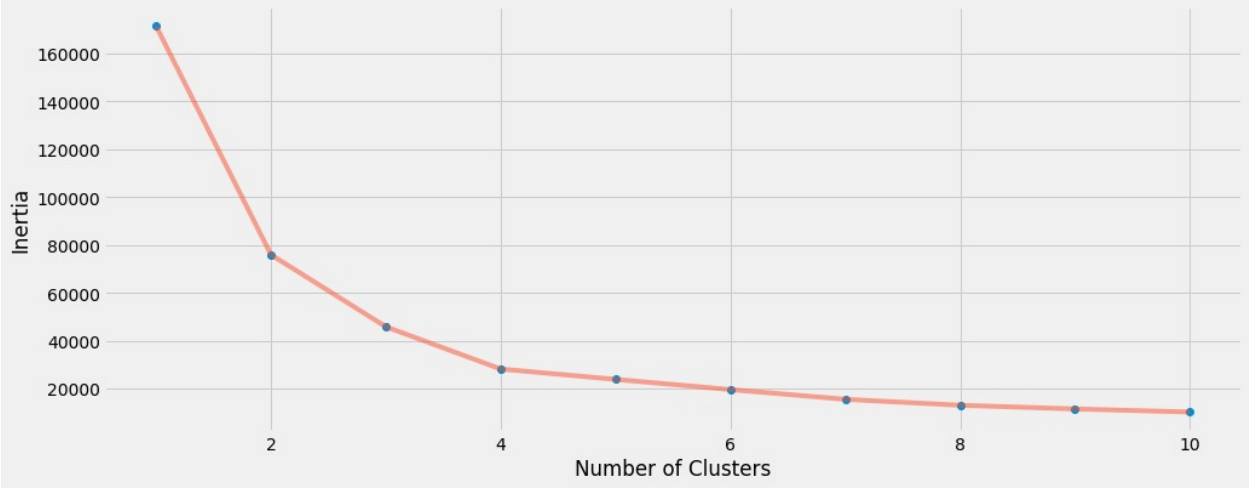
Clustering using K- means

1.Segmentation using Age and Spending Score

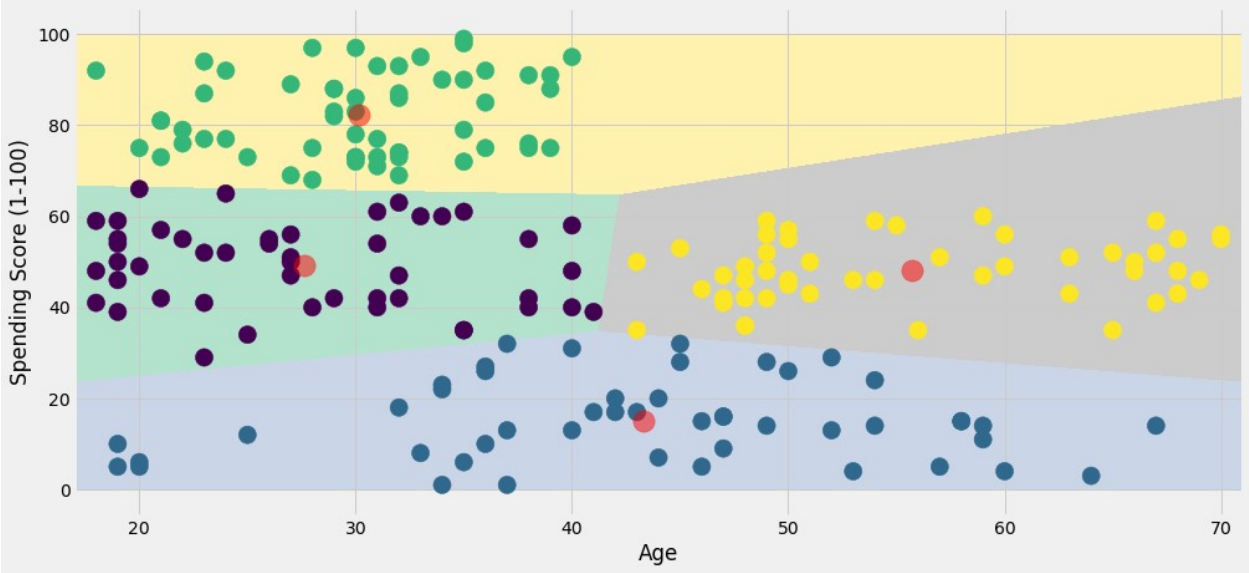
```
'''Age and spending Score'''
X1 = df[['Age' , 'Spending Score (1-100)']].iloc[: , :].values
inertia = []
for n in range(1 , 11):
    algorithm = (KMeans(n_clusters = n ,init='k-means++', n_init =
10 ,max_iter=300,
                        tol=0.0001,  random_state= 111  ,
algorithm='elkan') )
    algorithm.fit(X1)
    inertia.append(algorithm.inertia_)
```

Selecting N Clusters based in Inertia (Squared Distance between Centroids and data points, should be less)

```
plt.figure(1 , figsize = (15 ,6))
plt.plot(np.arange(1 , 11) , inertia , 'o')
plt.plot(np.arange(1 , 11) , inertia , '-' , alpha = 0.5)
plt.xlabel('Number of Clusters') , plt.ylabel('Inertia')
plt.show()
```

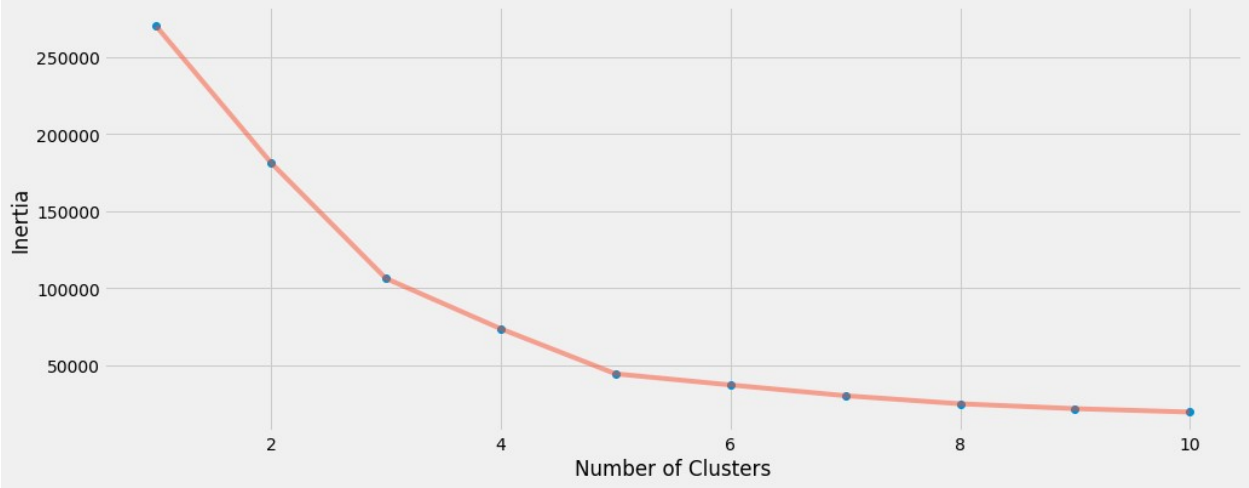



```
algorithm = (KMeans(n_clusters = 4 ,init='k-means++', n_init =
10 ,max_iter=300,
                        tol=0.0001,  random_state= 111  ,
algorithm='elkan') )
algorithm.fit(X1)
labels1 = algorithm.labels_
centroids1 = algorithm.cluster_centers_

h = 0.02
x_min, x_max = X1[:, 0].min() - 1, X1[:, 0].max() + 1
y_min, y_max = X1[:, 1].min() - 1, X1[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min,
y_max, h))
Z = algorithm.predict(np.c_[xx.ravel(), yy.ravel()])

plt.figure(1 , figsize = (15 , 7) )
plt.clf()
Z = Z.reshape(xx.shape)
plt.imshow(Z , interpolation='nearest',
           extent=(xx.min(), xx.max(), yy.min(), yy.max()),
           cmap = plt.cm.Pastel2, aspect = 'auto', origin='lower')

plt.scatter( x = 'Age' ,y = 'Spending Score (1-100)' , data = df , c =
labels1 ,
            s = 200 )
plt.scatter(x = centroids1[: , 0] , y =  centroids1[: , 1] , s = 300 ,
c = 'red' , alpha = 0.5)
plt.ylabel('Spending Score (1-100)') , plt.xlabel('Age')
plt.show()
```

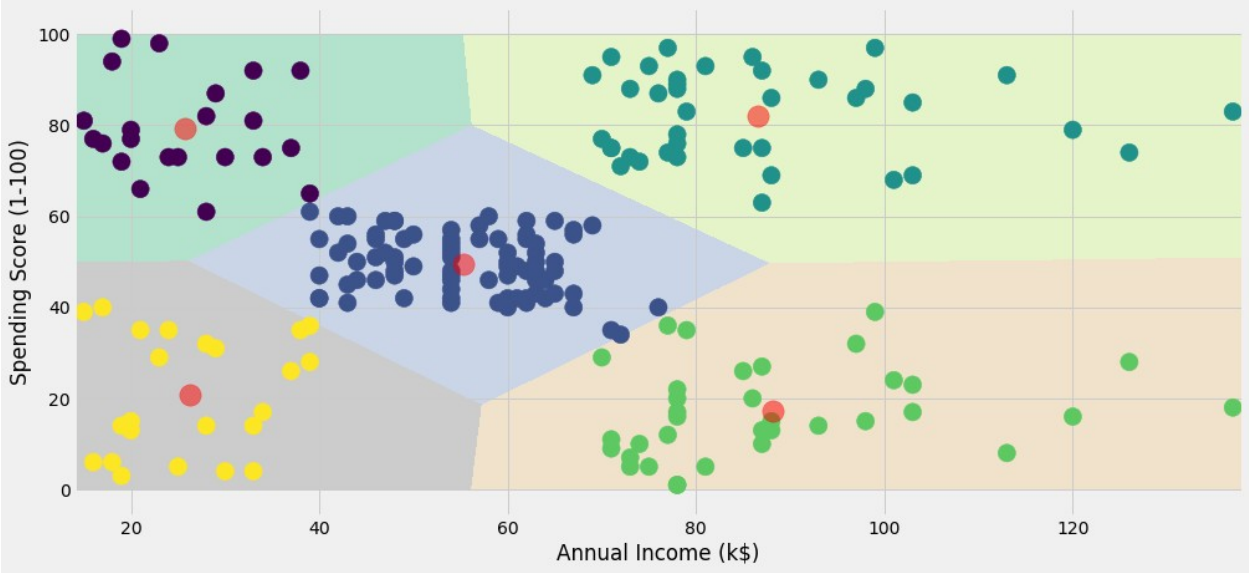


2. Segmentation using Annual Income and Spending Score

```
'''Annual Income and spending Score'''
X2 = df[['Annual Income (k$)' , 'Spending Score (1-
100)']].iloc[: , :].values
inertia = []
for n in range(1 , 11):
    algorithm = (KMeans(n_clusters = n ,init='k-means++', n_init =
10 ,max_iter=300,
                        tol=0.0001,  random_state= 111  ,
algorithm='elkan') )
    algorithm.fit(X2)
    inertia.append(algorithm.inertia_)

plt.figure(1 , figsize = (15 ,6))
plt.plot(np.arange(1 , 11) , inertia , 'o')
plt.plot(np.arange(1 , 11) , inertia , '-' , alpha = 0.5)
plt.xlabel('Number of Clusters') , plt.ylabel('Inertia')
plt.show()
```

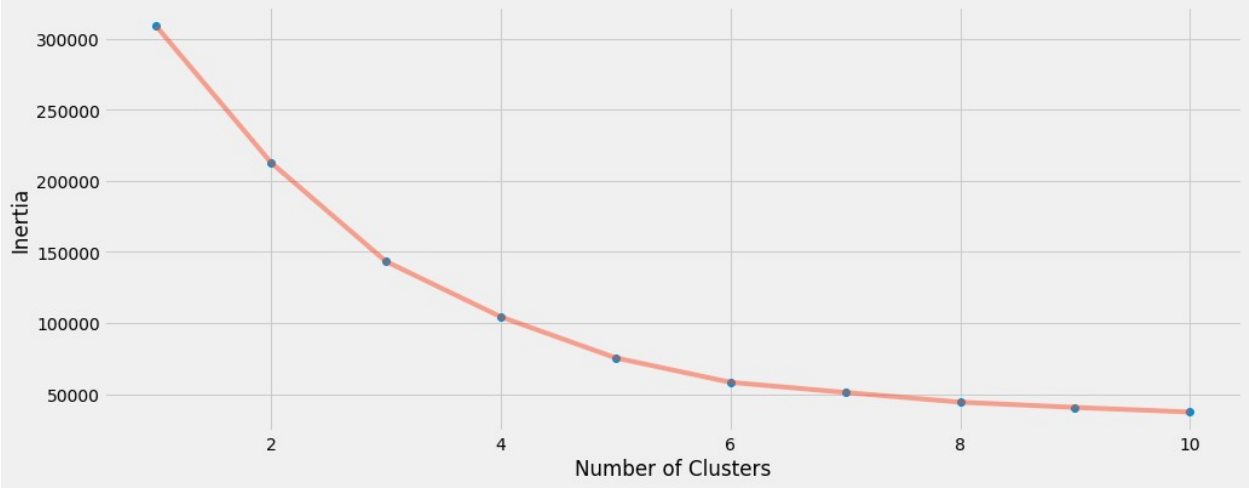



```
algorithm = (KMeans(n_clusters = 5 ,init='k-means++', n_init =
10 ,max_iter=300,
                        tol=0.0001,  random_state= 111  ,
algorithm='elkan') )
algorithm.fit(X2)
labels2 = algorithm.labels_
centroids2 = algorithm.cluster_centers_

h = 0.02
x_min, x_max = X2[:, 0].min() - 1, X2[:, 0].max() + 1
y_min, y_max = X2[:, 1].min() - 1, X2[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min,
y_max, h))
Z2 = algorithm.predict(np.c_[xx.ravel(), yy.ravel()])

plt.figure(1 , figsize = (15 , 7) )
plt.clf()
Z2 = Z2.reshape(xx.shape)
plt.imshow(Z2 , interpolation='nearest',
           extent=(xx.min(), xx.max(), yy.min(), yy.max()),
           cmap = plt.cm.Pastel2, aspect = 'auto', origin='lower')

plt.scatter( x = 'Annual Income (k$)' ,y = 'Spending Score (1-100)' ,
data = df , c = labels2 ,
            s = 200 )
plt.scatter(x = centroids2[: , 0] , y =  centroids2[: , 1] , s = 300 ,
c = 'red' , alpha = 0.5)
plt.ylabel('Spending Score (1-100)') , plt.xlabel('Annual Income
(k$)')
plt.show()
```



3.Segmentation using Age , Annual Income and Spending Score

```
X3 = df[['Age' , 'Annual Income (k$)' ,'Spending Score (1-
100)']].iloc[: , :].values
inertia = []
for n in range(1 , 11):
    algorithm = (KMeans(n_clusters = n ,init='k-means++', n_init =
10 ,max_iter=300,
                        tol=0.0001,  random_state= 111  ,
algorithm='elkan') )
    algorithm.fit(X3)
    inertia.append(algorithm.inertia_)

plt.figure(1 , figsize = (15 ,6))
plt.plot(np.arange(1 , 11) , inertia , 'o')
plt.plot(np.arange(1 , 11) , inertia , '-' , alpha = 0.5)
plt.xlabel('Number of Clusters') , plt.ylabel('Inertia')
plt.show()
```




```
algorithm = (KMeans(n_clusters = 6 ,init='k-means++', n_init =
10 ,max_iter=300,
                        tol=0.0001,  random_state= 111  ,
algorithm='elkan') )
algorithm.fit(X3)
labels3 = algorithm.labels_
centroids3 = algorithm.cluster_centers_

df['label3'] =  labels3
trace1 = go.Scatter3d(
    x= df['Age'],
    y= df['Spending Score (1-100)'],
    z= df['Annual Income (k$)'],
    mode='markers',
     marker=dict(
        color = df['label3'],
        size= 20,
        line=dict(
            color= df['label3'],
            width= 12
        ),
        opacity=0.8
     )
)
data = [trace1]
layout = go.Layout(
#     margin=dict(
#         l=0,
#         r=0,
#         b=0,
#         t=0
#     )
    title= 'Clusters',
    scene = dict(
            xaxis = dict(title  = 'Age'),
```

```
            yaxis = dict(title  = 'Spending Score'),
            zaxis = dict(title  = 'Annual Income')
        )
)
fig = go.Figure(data=data, layout=layout)
py.offline.iplot(fig)
```


```
{"config":{"linkText":"Export to
plot.ly","plotlyServerURL":"https://plot.ly","showLink":false},"data":
[{"marker":{"color":
[4,5,4,5,4,5,4,5,4,5,4,5,4,5,4,5,4,5,4,5,4,5,4,5,4,5,4,5,4,5,4,5,4
,5,4,5,4,5,2,5,2,0,4,5,2,0,0,0,2,0,0,2,2,2,2,2,0,2,2,0,2,2,2,0,2,2,0,0
,2,2,2,2,2,0,2,0,0,2,2,0,2,2,0,2,2,0,0,2,2,0,2,0,0,0,2,0,2,0,0,2,2,0,2
,0,2,2,2,2,2,0,0,0,0,0,2,2,2,2,2,0,0,0,3,0,3,1,3,1,3,1,3,0,3,1,3,1,3,1,3
,1,3,0,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1
,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3]],"line":{"color":
[4,5,4,5,4,5,4,5,4,5,4,5,4,5,4,5,4,5,4,5,4,5,4,5,4,5,4,5,4,5,4,5,4
,5,4,5,4,5,2,5,2,0,4,5,2,0,0,0,2,0,0,2,2,2,2,2,0,2,2,0,2,2,2,0,2,2,0,0
,2,2,2,2,2,0,2,0,0,2,2,0,2,2,0,2,2,0,0,2,2,0,2,0,0,0,2,0,2,0,0,2,2,0,2
,0,2,2,2,2,2,0,0,0,0,0,2,2,2,2,2,0,0,0,3,0,3,1,3,1,3,1,3,0,3,1,3,1,3,1,3
,1,3,0,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1
,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3,1,3],"width":12},"opaci
ty":0.8,"size":20},"mode":"markers","type":"scatter3d","uid":"d3d3d700
-6774-4cfd-80ab-c642007359d8","x":
[19,21,20,23,31,22,35,23,64,30,67,35,58,24,37,22,35,20,52,35,35,25,46,
31,54,29,45,35,40,23,60,21,53,18,49,21,42,30,36,20,65,24,48,31,49,24,5
0,27,29,31,49,33,31,59,50,47,51,69,27,53,70,19,67,54,63,18,43,68,19,32
,70,47,60,60,59,26,45,40,23,49,57,38,67,46,21,48,55,22,34,50,68,18,48,
40,32,24,47,27,48,20,23,49,67,26,49,21,66,54,68,66,65,19,38,19,18,19,6
3,49,51,50,27,38,40,39,23,31,43,40,59,38,47,39,25,31,20,29,44,32,19,35
,57,32,28,32,25,28,48,32,34,34,43,39,44,38,47,27,37,30,34,30,56,29,19,
31,50,36,42,33,36,32,40,28,36,36,52,30,58,27,59,35,37,32,46,29,41,30,5
4,28,41,36,34,32,33,38,47,35,45,32,32,30],"y":
[39,81,6,77,40,76,6,94,3,72,14,99,15,77,13,79,35,66,29,98,35,73,5,73,1
4,82,32,61,31,87,4,73,4,92,14,81,17,73,26,75,35,92,36,61,28,65,55,47,4
2,42,52,60,54,60,45,41,50,46,51,46,56,55,52,59,51,59,50,48,59,47,55,42
,49,56,47,54,53,48,52,42,51,55,41,44,57,46,58,55,60,46,55,41,49,40,42,
52,47,50,42,49,41,48,59,55,56,42,50,46,43,48,52,54,42,46,48,50,43,59,4
3,57,56,40,58,91,29,77,35,95,11,75,9,75,34,71,5,88,7,73,10,72,5,93,40,
87,12,97,36,74,22,90,17,88,20,76,16,89,1,78,1,73,35,83,5,93,26,75,20,9
5,27,63,13,75,10,92,13,86,15,69,14,90,32,86,15,88,39,97,24,68,17,85,23
,69,8,91,16,79,28,74,18,83],"z":
[15,15,16,16,17,17,18,18,19,19,19,19,20,20,20,20,21,21,23,23,24,24,25,
25,28,28,28,28,29,29,30,30,33,33,33,33,34,34,37,37,38,38,39,39,39,39,4
0,40,40,40,42,42,43,43,43,43,44,44,46,46,46,46,47,47,48,48,48,48,48,48
,49,49,50,50,54,54,54,54,54,54,54,54,54,54,54,57,57,58,58,59,59,60,
60,60,60,60,60,61,61,62,62,62,62,62,62,63,63,63,63,63,63,64,64,65,65,6
5,65,67,67,67,67,69,69,70,70,71,71,71,71,71,71,72,72,73,73,73,73,74,74
,75,75,76,76,77,77,77,77,78,78,78,78,78,78,78,78,78,78,78,78,79,79,81,
81,85,85,86,86,87,87,87,87,87,87,88,88,88,88,93,93,97,97,98,98,99,99,1
```

```
01,101,103,103,103,103,113,113,120,120,126,126,137,137]}],"layout":
{"scene":{"xaxis":{"title":{"text":"Age"}},"yaxis":{"title":
{"text":"Spending Score"}},"zaxis":{"title":{"text":"Annual
Income"}}},"title":{"text":"Clusters"}}}
```


If you liked my Work Please Upvote , Thank you.