CUSTOMER SEGMENTATION USING DATA SCIENCE

TEAM MEMBERS

Vandna KUMARI D:-310521104123

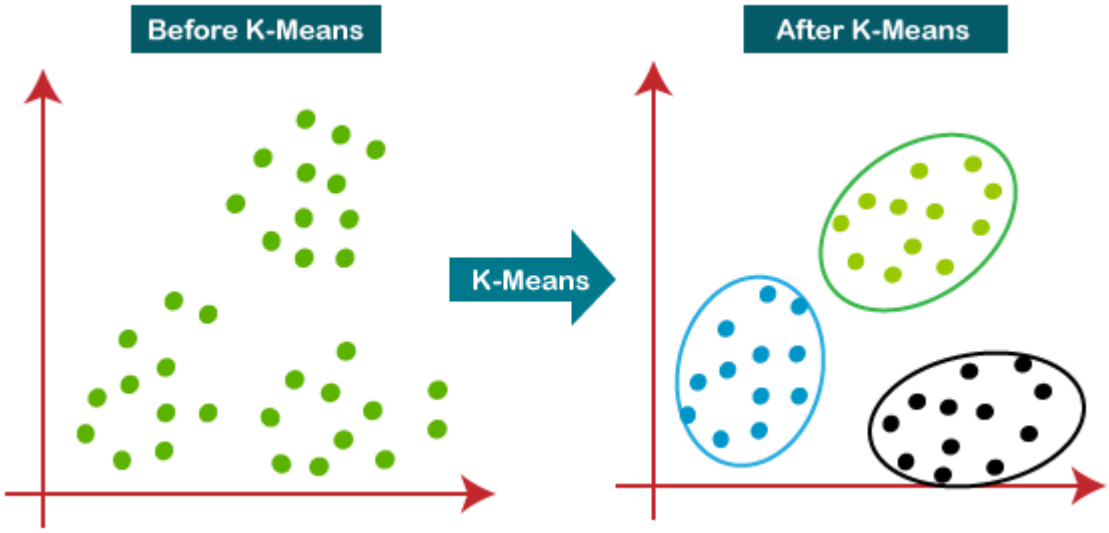
INTRODUCTION:

In comparison to customer segmentation, engineering segmentation is more general and looks at the entire marketplace. Whereas engineering segmentation relates to the whole market, customer segmentation is your part of the market.

For example, if you're in the business of selling vehicles and you typically sell directly to businesses, then your customer segment is B2B and you might compare customers that are likely to buy large commercial trucks, versus small business-owned vans. These two customers have different needs and, depending on the correlation you find, might then become two different customer segments for you to focus on.

```
import numpy as np # linear algebra

import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

import os

for dirname, _, filenames in os.walk('/kaggle/input'):

    for filename in filenames:

        print(os.path.join(dirname, filename))
```

K Means Clustering for Customer Data



Clustering

Clustering is the task of dividing the population or data points into a number of groups such that data points in the same groups are more similar to other data points in the same group than those in other groups. In simple words, the aim is to segregate groups with similar traits and assign them into clusters.

KMeans Clustering

K-means clustering is one of the simplest and popular unsupervised machine learning algorithms. You'll define a target number k, which refers to the number of centroids you need in the dataset. A centroid is the imaginary or real location representing the center of the cluster. Every data point is allocated to each of the clusters through reducing the in-cluster sum of squares. In other words, the K-means algorithm identifies k number of centroids, and then allocates every data point to the nearest cluster, while keeping the centroids as small as possible. The 'means' in the K-means refers to averaging of the data; that is, finding the centroid.

About the dataset

This input file contains the basic information (ID, age, gender, income, spending score) about the customers of a mall. Spending Score is something you assign to the customer based on your defined parameters like customer behavior and purchasing data.

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

```
In [3]:df=pd.read_csv('/kaggle/input/mall-customers/Mall_Customers.
df.rename(columns={'Genre':'Gender'},inplace=True)
df.head()
```

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

	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

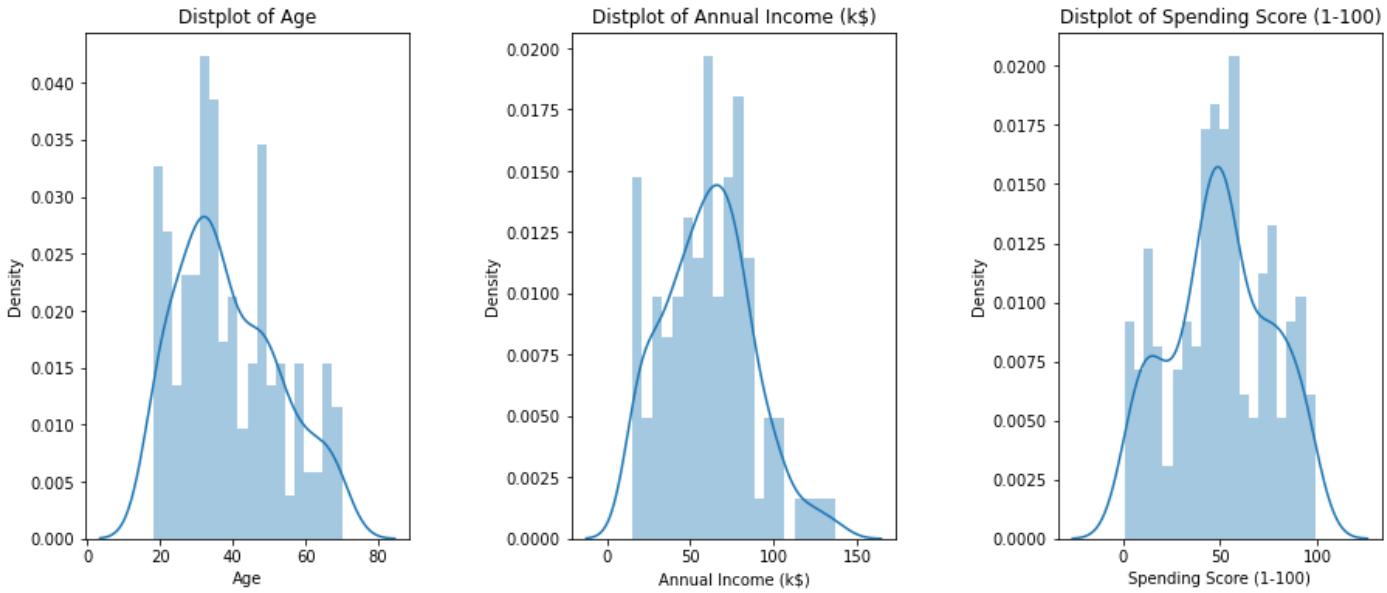
	CustomerID	Age	Annual Income (k$)	Spending Score (1-100)
50%	100.500000	36.000000	61.500000	50.000000
75%	150.250000	49.000000	78.000000	73.000000
max	200.000000	70.000000	137.000000	99.000000

df.isnull().sum()

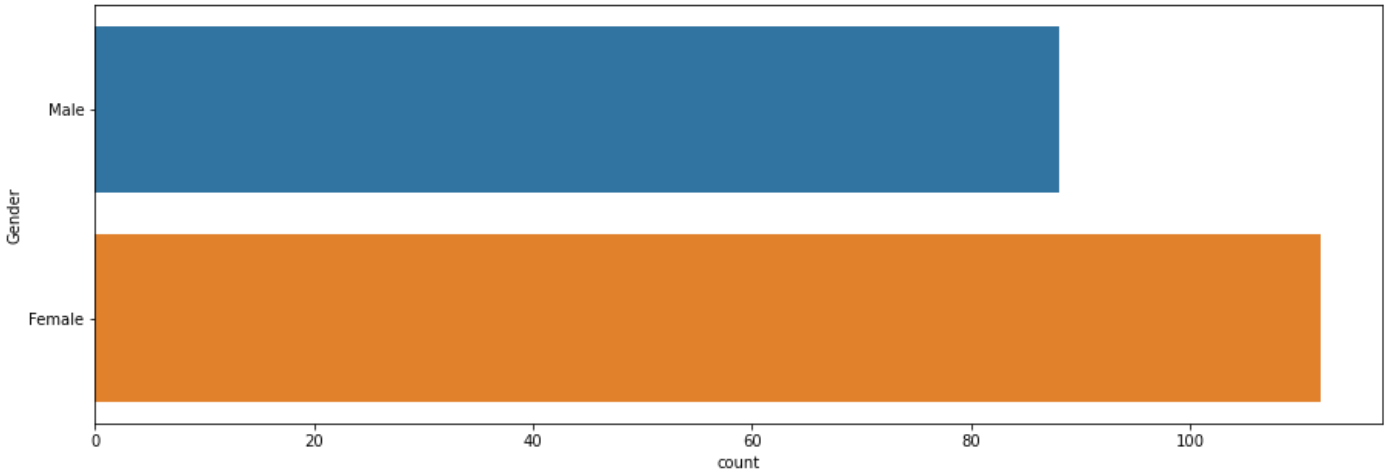
Out[5]:
CustomerID 0
Gender 0
Age 0
Annual Income (k$) 0
Spending Score (1-100) 0
dtype: int64

In [6]:
df.drop(['CustomerID'],axis=1,inplace=True)

In [7]:
linkcode
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))

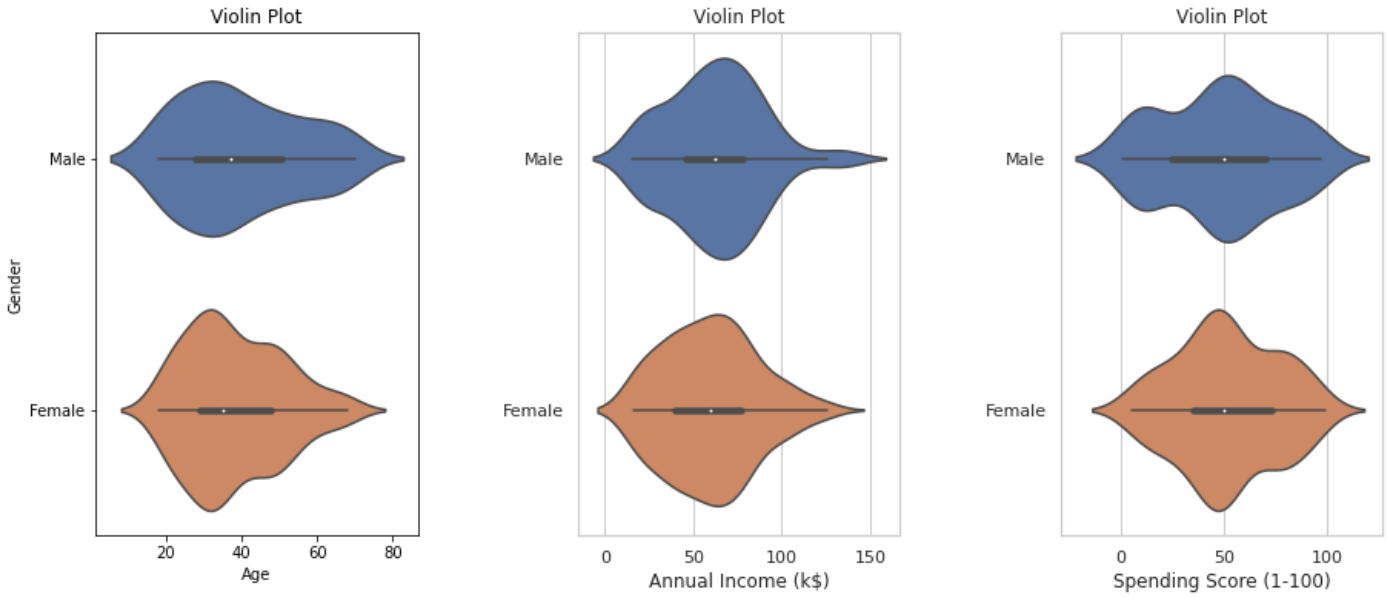



```
plt.figure(figsize=(15,5))
sns.countplot(y='Gender',data=df)
plt.show()
```

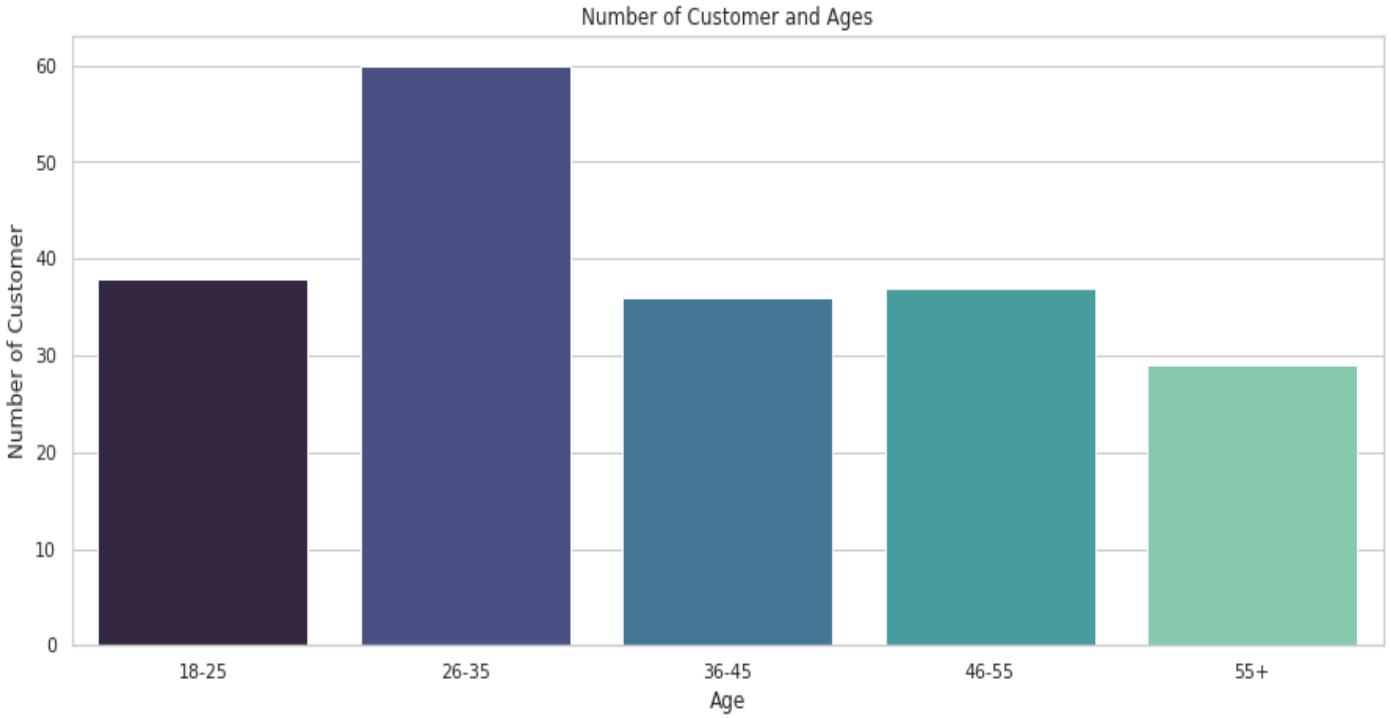



```
plt.figure(1,figsize=(15,6))
n = 0
for cols in ['Age','Annual Income (k$)','Spending Score (1-100)']:
    n +=1
    plt.subplot(1,3,n)
    sns.set(style="whitegrid")
    plt.subplots_adjust(hspace=0.5,wspace=0.5)
    sns.violinplot(x = cols,y = 'Gender',data=df)
    plt.ylabel('Gender' if n== 1 else '')
    plt.title('Violin Plot')
```

```
plt.show()
```

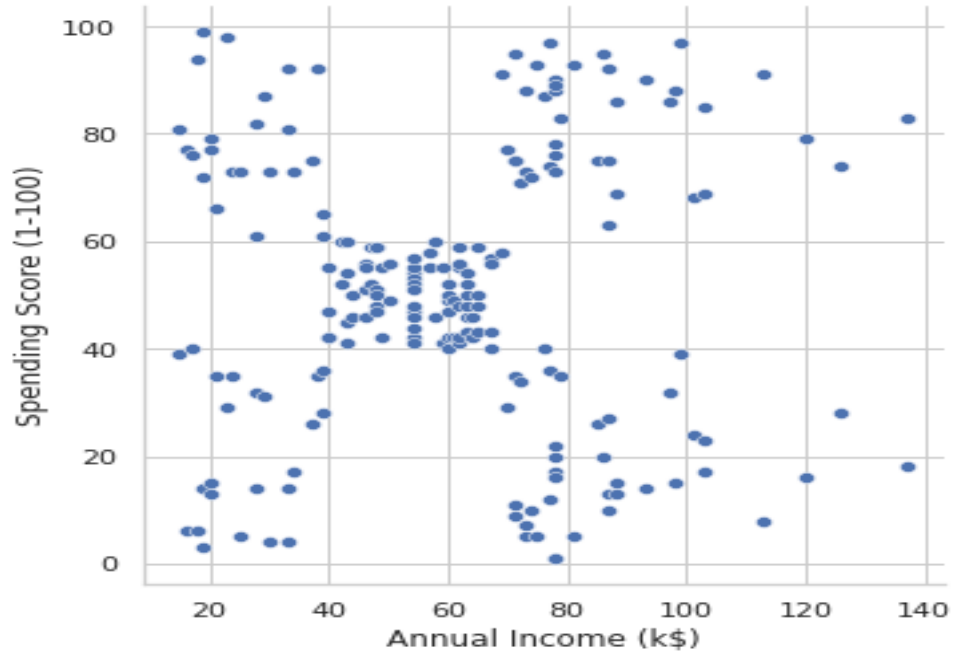



```
age_18_25 = df.Age[(df.Age >=18) & (df.Age <= 25)]
age_26_35 = df.Age[(df.Age >=26) & (df.Age <= 35)]
age_36_45 = df.Age[(df.Age >=36) & (df.Age <= 45)]
age_46_55 = df.Age[(df.Age >=46) & (df.Age <= 55)]
age_55_above = df.Age[(df.Age >= 56)]

age_x =["18-25","26-35","36-45","46-55","55+"]
age_y = [len(age_18_25.values),len(age_26_35.values),len(age_36_45),len(age
_46_55),len(age_55_above)]

plt.figure(figsize = (15,6))
sns.barplot(x=age_x, y=age_y,palette = "mako")
plt.title("Number of Customer and Ages")
plt.xlabel("Age")
plt.ylabel("Number of Customer")
plt.show()
```



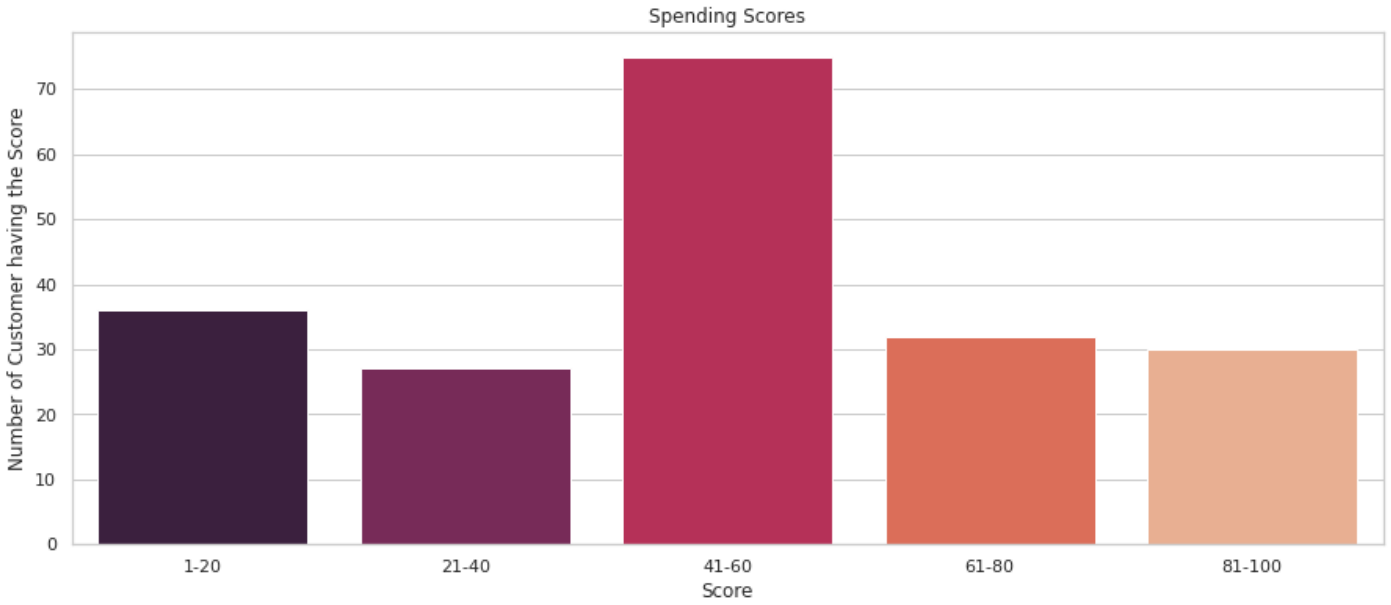
sns.relplot(x="Annual Income (k$)",y = "Spending Score (1-100)",data=df)

<seaborn.axisgrid.FacetGrid at 0x7fcef0536fd0>



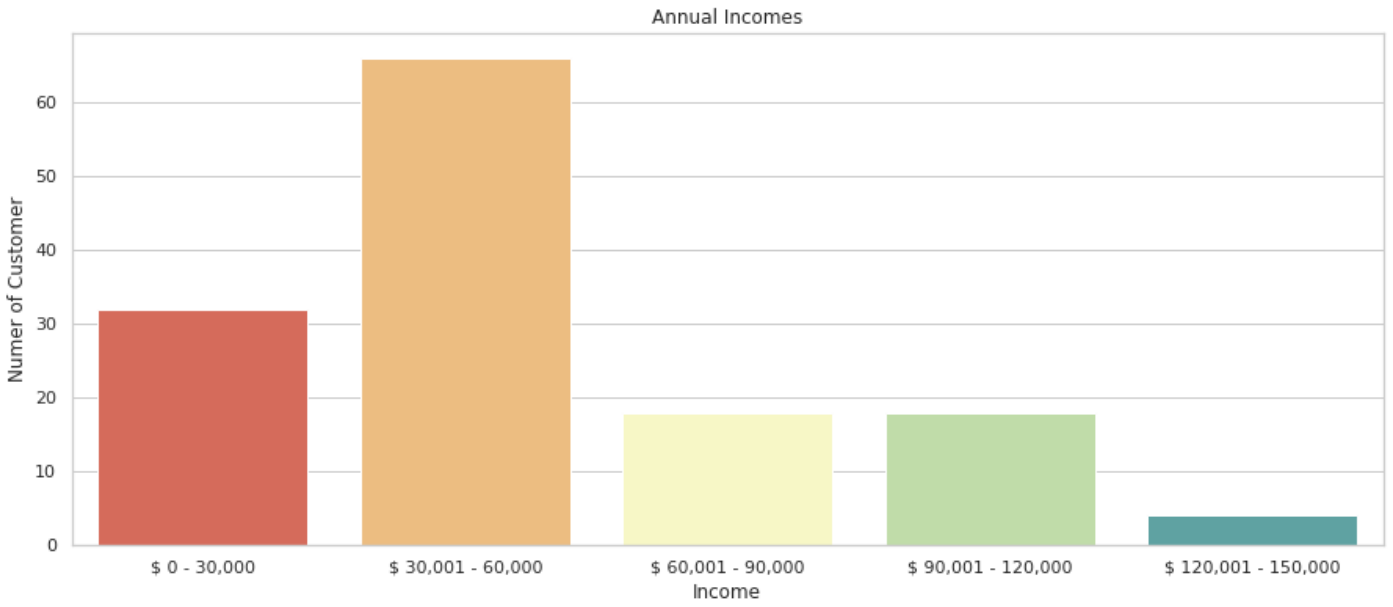
ss_1_20 = df["Spending Score (1-100)"][(df["Spending Score (1-100)"] >= 1) & (df["Spending Score (1-100)"] <= 20)]

```
ss_21_40 = df["Spending Score (1-100)"][(df["Spending Score (1-100)"] >= 21
) & (df["Spending Score (1-100)"] <= 40)]
ss_41_60 = df["Spending Score (1-100)"][(df["Spending Score (1-100)"] >= 41
) & (df["Spending Score (1-100)"] <= 60)]
ss_61_80 = df["Spending Score (1-100)"][(df["Spending Score (1-100)"] >= 61
) & (df["Spending Score (1-100)"] <= 80)]
ss_81_100 = df["Spending Score (1-100)"][(df["Spending Score (1-100)"] >= 8
1) & (df["Spending Score (1-100)"] <= 100)]

ssx= ["1-20","21-40","41-60","61-80","81-100"]
ssy=[len(ss_1_20.values),len(ss_21_40.values),len(ss_41_60.values),len(ss_6
1_80.values),len(ss_81_100.values)]

plt.figure(figsize=(15,6))
sns.barplot(x=ssx,y=ssy, palette="rocket")
plt.title("Spending Scores")
plt.xlabel("Score")
plt.ylabel("Number of Customer having the Score")
plt.show()
```

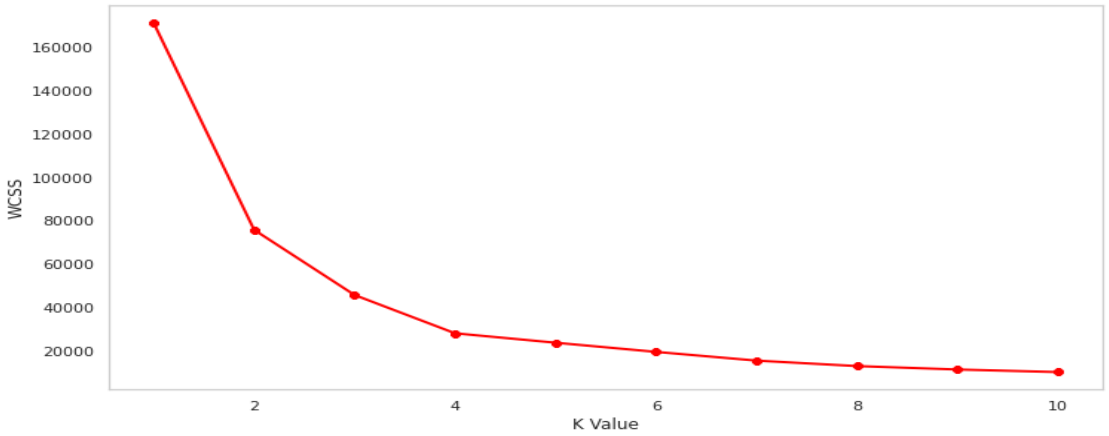



```
ai_0_30 = df["Annual Income (k$)"][(df["Annual Income (k$)"] >= 0) & (df["A
nnual Income (k$)"] <= 30)]
ai_31_60= df["Annual Income (k$)"][(df["Annual Income (k$)"] >=31)& (df["An
nual Income (k$)"] <=60)]
ai_61_90= df["Annual Income (k$)"][(df["Annual Income (k$)"] >=61)& (df["An
nual Income (k$)"] <=90)]
ai_61_90=df["Annual Income (k$)"][(df["Annual Income (k$)"] >=91)& (df["Ann
ual Income (k$)"] <=120)]
ai_121_150 = df["Annual Income (k$)"][(df["Annual Income (k$)"]>=121) & (df
["Annual Income (k$)"] <=150)]

aix = ["$ 0 - 30,000","$ 30,001 - 60,000","$ 60,001 - 90,000","$ 90,001 - 1
20,000","$ 120,001 - 150,000"]
```

```
aiy = [len(ai_0_30.values),len(ai_31_60.values),len(ai_61_90.values),len(ai
_61_90.values),len(ai_121_150.values)]

plt.figure(figsize=(15,6))
sns.barplot(x=aix,y=aiy,palette="Spectral")
plt.title("Annual Incomes")
plt.xlabel("Income")
plt.ylabel("Numer of Customer")
plt.show()
```




```
X1 = df.loc[:,["Age","Spending Score (1-100)"]].values

from sklearn.cluster import KMeans
wcss=[]
for k in range(1,11):
    kmeans = KMeans(n_clusters = k, init = "k-means++")
    kmeans.fit(X1)
    wcss.append(kmeans.inertia_)
plt.figure(figsize =( 12,6))
plt.grid()
plt.plot(range(1,11),wcss,linewidth=2,color="red",marker="8")
plt.xlabel("K Value")
plt.ylabel("WCSS")
plt.show()
```




```
kmeans = KMeans(n_clusters=4)

label = kmeans.fit_predict(X1)

print(label)
```

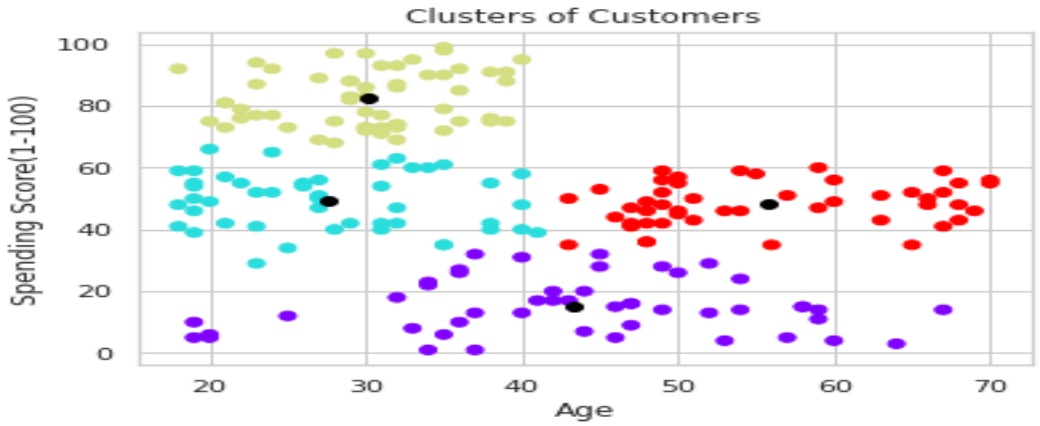
```
[1 2 0 2 1 2 0 2 0 2 0 2 0 2 0 2 1 1 0 2 1 2 0 2 0 2 0 1 0 2 0 2 0 2 0 2 0
 2 0 2 3 2 3 1 0 1 3 1 1 1 3 1 1 3 3 3 3 3 1 3 3 1 3 3 3 1 3 3 1 1 3 3 3 3
 3 1 3 1 1 3 3 1 3 3 1 3 3 1 1 3 3 1 3 1 1 1 3 1 3 1 1 3 3 1 3 1 3 3 3 3 3
 1 1 1 1 1 3 3 3 3 1 1 1 2 1 2 3 2 0 2 0 2 1 2 0 2 0 2 0 2 0 2 1 2 0 2 3 2
 0 2 0 2 0 2 0 2 0 2 0 2 3 2 0 2 0 2 0 2 0 1 0 2 0 2 0 2 0 2 0 2 0 2 0 2 1
 2 0 2 0 2 0 2 0 2 0 2 0 2 0 2 0 2]
```

```
print(kmeans.cluster_centers_)
```

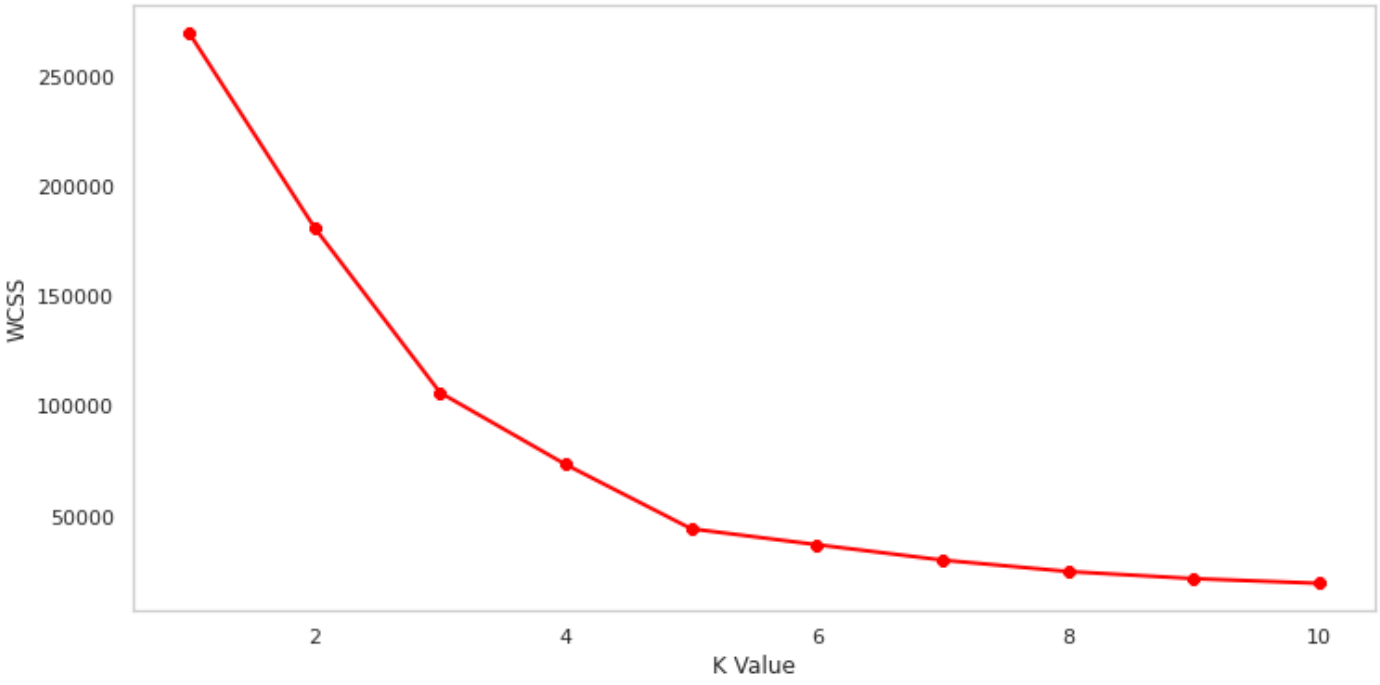
```
[[43.29166667 15.02083333]
 [27.61702128 49.14893617]
 [30.1754386  82.35087719]
 [55.70833333 48.22916667]]
```

```
plt.scatter(X1[:,0],X1[:,1],c=kmeans.labels_,cmap='rainbow')
plt.scatter(kmeans.cluster_centers_[:,0],kmeans.cluster_centers_[:,1],color='black')
plt.title('Clusters of Customers')
plt.xlabel('Age')
plt.ylabel('Spending Score(1-100)')
plt.show
```

```
<function matplotlib.pyplot.show(close=None, block=None)>
```




```
X2 = df.loc[:,["Annual Income (k$)","Spending Score (1-100)"]].values

from sklearn.cluster import KMeans
wcss=[]
for k in range(1,11):
    kmeans = KMeans(n_clusters = k, init = "k-means++")
    kmeans.fit(X2)
    wcss.append(kmeans.inertia_)
plt.figure(figsize =( 12,6))
plt.grid()
plt.plot(range(1,11),wcss,linewidth=2,color="red",marker="8")
plt.xlabel("K Value")
plt.ylabel("WCSS")
plt.show()
```



kmeans = KMeans(n_clusters=5)

label = kmeans.fit_predict(X3)

print(label)

```
[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 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 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 4 2 4 0 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 0 4 2 4 2 4
 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2
 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4]
```

```
plt.scatter(X2[:,0],X1[:,1],c=kmeans.labels_,cmap='rainbow')
plt.scatter(kmeans.cluster_centers_[:,0],kmeans.cluster_centers_[:,1],color
='black')
plt.title('Clusters of Customers')
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score(1-100)')
plt.show
```

```
<function matplotlib.pyplot.show(close=None, block=None)
```


```
X3 = df.iloc[:,1:]
```

```
wcss=[]
for k in range(1,11):
    kmeans = KMeans(n_clusters = k, init = "k-means++")
    kmeans.fit(X3)
    wcss.append(kmeans.inertia_)
plt.figure(figsize =( 12,6))
plt.grid()
plt.plot(range(1,11),wcss,linewidth=2,color="red",marker="8")
plt.xlabel("K Value")
plt.ylabel("WCSS")
plt.show()
```

Customer Segmentation Model

This notebook contains a clustering exercise used to practice model interpretation of grouped data. The hardest part about building a clustering model is generally understanding the results it outputs, as you don't have a target variable that you can compare your results with.

Therefore, interpretation can become quite complex and so we explore a couple of alternatives for understanding your clusters once you have trained your model. Below are the steps taken to clean up the data, exploring it and finally fitting it to a K-Means clustering model and interpreting the results.

I belive the most important part of this exercise is the clustering interpretation, as every time we perform this type of unsupervised task we should be aware that the end goal is to provide business insights through the results.

If you like the notebook, don't forget to upvote!!

Imports and information about the variables:

Variable description including type, range and full description:

Variable	Data Type	Range	Description
ID	numerical	Integer	Shows a unique identificator of a customer.
Sex	categorical	{0,1}	Biological sex (gender) of a customer. 0 = male / 1 = female
Marital status	categorical	{0,1}	Marital status of a customer. 0 = single / 1 = non-single

Variable	Data Type	Range	Description
Age	numerical	Integer	The age of the customer in years, calculated as current year minus the year of birth of the customer at the time of creation of the dataset (Min. age = 18 / Max. age = 78)
Education	categorical	{0,1,2,3}	Level of education of the customer. 0=no education / 1=high-school / 2=university / 3=graduate
Income	numerical	Real	Self-reported annual income in US dollars of the customer.
Occupation	categorical	{0,1,2}	Category of occupation of the customer. 0=unemployed / 1=employee/oficial / 2=management or self-employed
Settlement size	categorical	{0,1,2}	The size of the city that the customer lives in. 0=small / 1=mid-size / 2=big

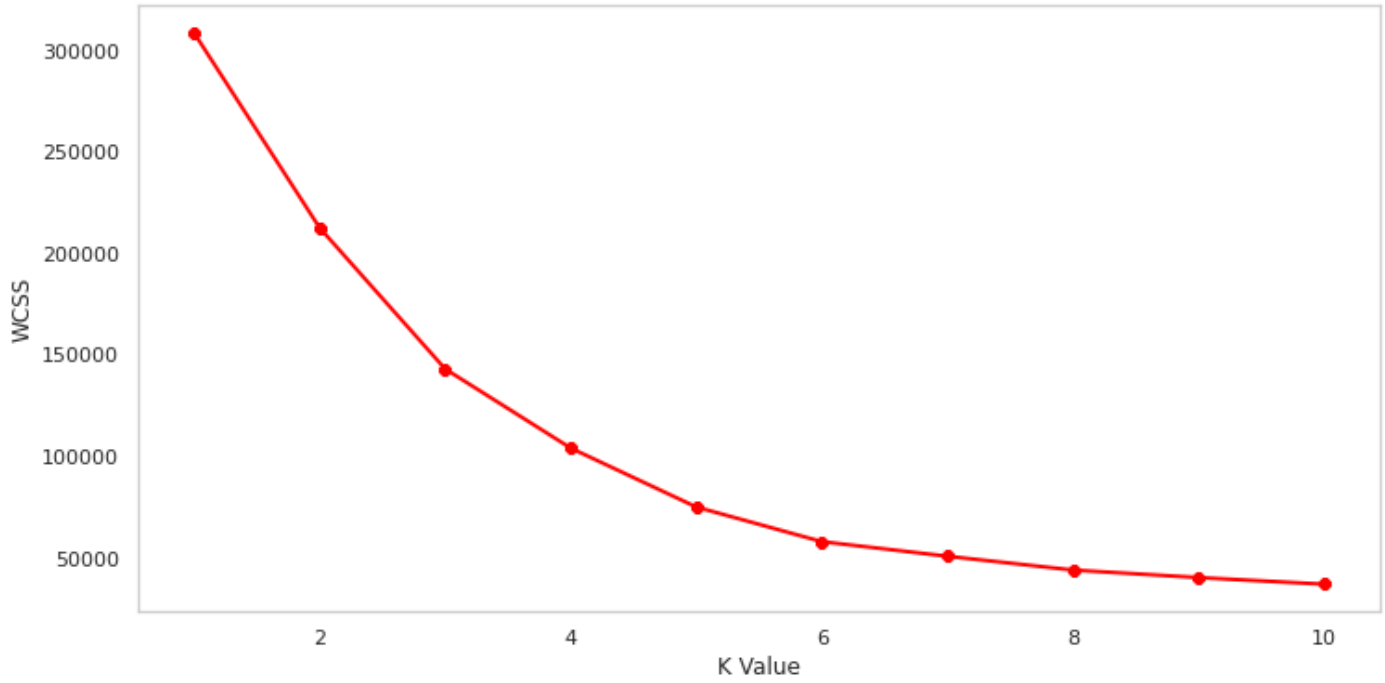
Imports:

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

Loading the dataset

```
customer_info = pd.read_csv('../input/customer-clustering/segmentation data
.csv')
customer_info.sample(5)
```

	ID	Sex	Marital status	Age	Education	Income	Occupation	Settlement size
1851	100001852	1	1	26	1	57165	0	0
1425	100001426	1	1	27	1	131542	2	1
1030	100001031	0	0	60	2	166298	2	1
1784	100001785	0	0	26	0	124783	1	0
1563	100001564	1	1	34	1	127255	1	0



kmeans = KMeans(n_clusters=5)

```
label = kmeans.fit_predict(X3)
```

```
print(label)
```

```
[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 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 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 4 2 4 0 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 0 4 2 4 2 4
 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2
 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4]
```

```
print(kmeans.cluster_centers_)
```

```
[[43.08860759 55.29113924 49.56962025]
 [25.52173913 26.30434783 78.56521739]
 [40.66666667 87.75       17.58333333]
 [45.2173913  26.30434783 20.91304348]
 [32.69230769 86.53846154 82.12820513]]
```

linkcode

```
cluster = kmeans.fit_predict(X3)
df["label"] = cluster


from mpl_toolkits.mplot3d import Axes3D


fig = plt.figure(figsize=(20,10))
ax = fig.add_subplot(111,projection = '3d')

ax.scatter(df.Age[df.label == 0 ],df["Annual Income (k$)"][df.label == 0],df["Spending Score (1-100)"][
df.label == 0], c = 'blue',s=60)
ax.scatter(df.Age[df.label == 1 ],df["Annual Income (k$)"][df.label == 1],df["Spending Score (1-100)"][
df.label == 1], c = 'red',s=60)
ax.scatter(df.Age[df.label == 2 ],df["Annual Income (k$)"][df.label == 2],df["Spending Score (1-100)"][
df.label == 2], c = 'green',s=60)
ax.scatter(df.Age[df.label == 3 ],df["Annual Income (k$)"][df.label == 3],df["Spending Score (1-100)"][
df.label == 3], c = 'orange',s=60)
ax.scatter(df.Age[df.label == 4 ],df["Annual Income (k$)"][df.label == 4],df["Spending Score (1-100)"][
df.label == 4], c = 'purple',s=60)


ax.view_init(30,185)


plt.xlabel("Age")
```

```
plt.ylabel("Annual Income (K$)")
ax.set_zlabel('Spending Score(1-100)')


plt.show()
```