CUSTOMER SEGMENTATION USING MACHINE LEARNING

(adopted from kaggle)

Appendix-1

Importing necessary libraries

In [125]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
import plotly.express as px
import plotly.graph_objects as go
```

Reading the data

In [126]:
```
df = pd.read_csv(r"E:\Fynn Lab\Note Books/Mall_Customers.csv")
```

Exploring the data

In [127]:
```
df
```

Out[127]:

	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

In [128]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 5 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   CustomerID              200 non-null    int64
 1   Gender                  200 non-null    object
 2   Age                     200 non-null    int64
 3   Annual Income (k$)      200 non-null    int64
 4   Spending Score (1-100)  200 non-null    int64
dtypes: int64(4), object(1)
memory usage: 7.9+ KB
```

Performing Exploratory Data Analysis

In [129]:
```
df.drop("CustomerID",axis=1)
```

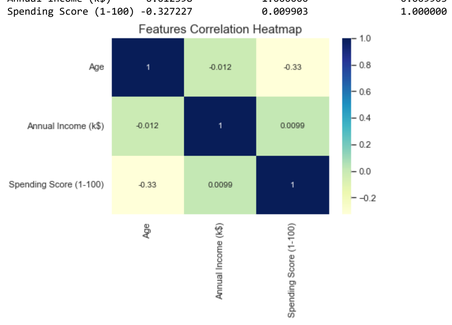
Out[129]:

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

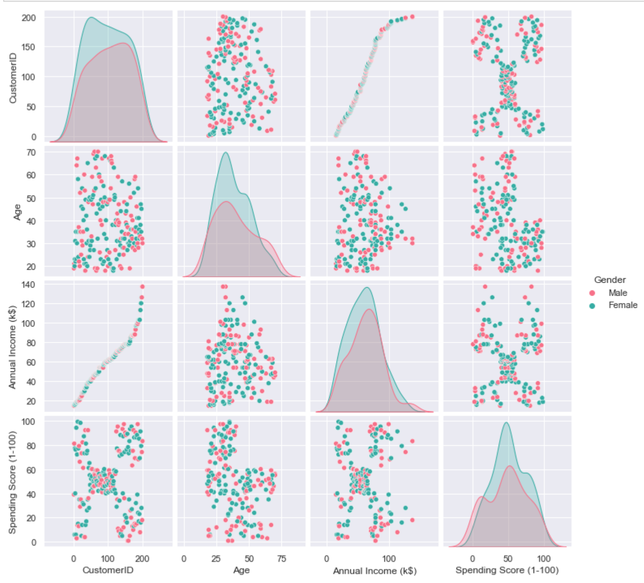
200 rows × 4 columns

In [130]:
```
print(df.drop("CustomerID", axis=1).corr())
sns.heatmap(df.drop("CustomerID",axis=1).corr(), cmap="YlGnBu", annot=True)
plt.title('Features Correlation Heatmap', fontsize = 15);
```

```
                        Age  Annual Income (k$)  Spending Score (1-100)
Age                1.000000           -0.012398               -0.327227
Annual Income (k$) -0.012398            1.000000                0.009903
Spending Score (1-100) -0.327227        0.009903                1.000000
```



In [131]:
```
sns.pairplot(df, hue = 'Gender' ,palette="husl");
```



In [132]:
```
sns.countplot(data = df, x = 'Gender')
plt.title('Gender Countplot', fontsize = 16, y = 1.1);
```


In [133]:
```
#Generating box plot to examine if any outlier is there
fig, axes = plt.subplots(1,3, figsize = (15,5))
fig.suptitle('Useful Boxplots', fontsize = 18)

sns.boxplot(data = df, x = 'Gender', y = 'Age', ax = axes[0])
sns.boxplot(data = df, x = 'Gender', y = 'Annual Income (k$)', ax = axes[1])
sns.boxplot(data = df, x = 'Gender', y = 'Spending Score (1-100)', ax = axes[2]);
```

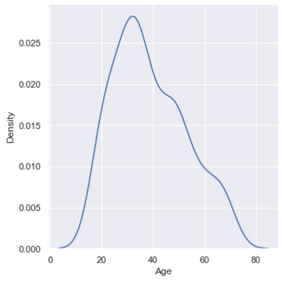

We can observe that there is mostly no outliers.

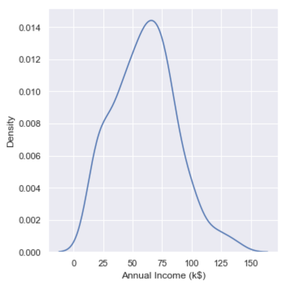
In [134]:
```
#Droping "Custoemer ID column"
df1=df.drop("CustomerID", axis=1)
df1.describe()
```

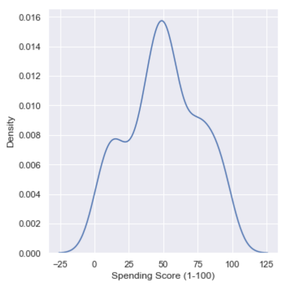
Out[134]:

	Age	Annual Income (k$)	Spending Score (1-100)
count	200.000000	200.000000	200.000000
mean	38.850000	60.560000	50.200000
std	13.969007	26.264721	25.823522
min	18.000000	15.000000	1.000000
25%	28.750000	41.500000	34.750000
50%	36.000000	61.500000	50.000000
75%	49.000000	78.000000	73.000000
max	70.000000	137.000000	99.000000

In [135]:
```
# Generating distribution curve to observe if any skweness is there
sns.set_theme()
sns.displot(df1, x="Age", kind="kde")
sns.displot(df1, x="Annual Income (k$)",kind="kde")
sns.displot(df1, x="Spending Score (1-100)",kind="kde");
```







It can be observed that data of Age coulumn and Annual Income is sckewed but of low mangnitute

hence no processing for removal of skewness is done

In [136]:
```
df1.head(1)
```

Out[136]:

	Gender	Age	Annual Income (k$)	Spending Score (1-100)
0	Male	19	15	39

Data Cleaning

In [137]:
```
df1.isnull().sum()
```

Out[137]:
```
Gender                    0
Age                       0
Annual Income (k$)        0
Spending Score (1-100)    0
dtype: int64
```

No null value in the data so no need for cleaning

In [138]:
```
df1.drop(['Gender'],axis=1, inplace=True)
```

In [139]:
```
df1
```

Out[139]:

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

200 rows × 3 columns

K-Means Clustering

In [140]:
```
#Importing libraries
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
```

Since K-Means is a distance based algorithim, performing Scalling

In [141]:
```
scaler=StandardScaler()
scaled_df = scaler.fit_transform(df1)
```

As a per part of hyperparameter tunng and to dicide the optimum Nos of clusters Elbow method is adopted

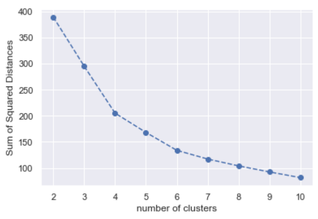
In [142]:
```
intertia = []

for i in range(2, 11):

    model=KMeans(n_clusters=i, random_state=101)
    model.fit(scaled_df)

    intertia.append(model.inertia_)
```

In [143]:
```
plt.plot(range(2,11), intertia, 'o--')
plt.xlabel('number of clusters')
plt.ylabel('Sum of Squared Distances');
```



The best cluser may be 6 or 5 as per the elbow method. The best way to generate the model for cluster No 5 and Cluster No 6 and

`Points to remember while calculating silhouette coefficient:`
`The value of the silhouette coefficient is between [-1, 1].`
`A score of 1 denotes the best meaning that the data point i is very compact within the cluster to which it belongs and far away from the other clusters.`
`The worst value is -1. Values near 0 denote overlapping clusters.`

Generating model for Cluster No 6

In [144]:
```
df_model = KMeans(n_clusters=6, random_state=101,init='k-means++',  max_iter=400)
```

In [145]:
```
df_model.fit(scaled_df)
```

Out[145]:
```
KMeans(max_iter=400, n_clusters=6, random_state=101)
```

In [146]:
```
df_labels = df_model.labels_
df_centroids = df_model.cluster_centers_
```

In [147]:
```
#Calculating the silhouette score for Cluster no 6
from sklearn.metrics import silhouette_score
silhouette_score(scaled_df,df_model.labels_)
```

Out[147]:
```
0.4284167762892593
```

Generating model for Cluster No 5

In [148]:
```
df_model1 = KMeans(n_clusters=5, random_state=101,init='k-means++',  max_iter=400)
df_model1.fit(scaled_df)
df_labels1 = df_model1.labels_
df_centroids1 = df_model1.cluster_centers_
```

In [149]:
```
#Calculating the silhouette score for Cluster no 6
from sklearn.metrics import silhouette_score
silhouette_score(scaled_df,df_model1.labels_)
```

Out[149]:
```
0.41664341513732767
```

silhouette_score for cluster No 6 is .42 where as the same is .41 for cluster No 5 so Cluster No 6 is taken for further analysis

In [150]:
```
df['clusters'] = df_labels
```

In [151]:
```
df.head()
```

Out[151]:

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

In [152]:
```
df_centroids
```

Out[152]:
```
array([[-0.86515664, -0.13196835, -0.08043031],
       [ 1.25472096, -0.24021294, -0.04399777],
       [ 0.47895722, -1.30822992, -1.19644353],
       [-0.44191719,  0.99158305,  1.23950275],
       [ 0.22171558,  1.08322527, -1.29005223],
       [-0.99396865, -1.34734766,  1.06404834]])
```

In [153]:
```
df.head(1)
```

Out[153]:

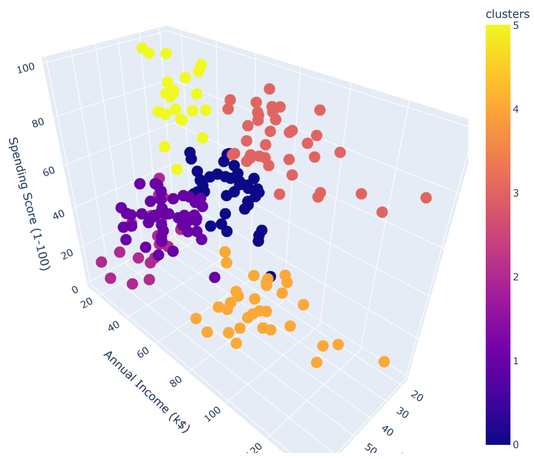
	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)	clusters
0	1	Male	19	15	39	5

In [154]:
```
import plotly.io as pio
pio.renderers.default = "notebook"
```

In [155]:
```
fig = plt.figure(figsize=(30,8))
# visualize clusters
figure = px.scatter_3d(df,
                       color='clusters',
                       x="Age",
                       y="Annual Income (k$)",
                       z="Spending Score (1-100)",
                       category_orders = {"clusters": ["0", "1", "2", "3", "4","5"]}
                       )
```

```
figure.update_layout(margin=dict(l=20, r=20, t=20, b=20),
                     #paper_bgcolor="LightSteelBlue",
    autosize=False,
    width=700,
    height=600
)

figure.show();
```




```
<Figure size 2160x576 with 0 Axes>
```

In [156]:
```
df.head(1)
```

Out[156]:

	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)	clusters
0	1	Male	19	15	39	5

Check for Cluster Magnitude

In [157]:
```
CustomerMagnitue_df = pd.DataFrame(
    df.clusters.value_counts().reset_index())

CustomerMagnitue_df.rename(columns={"index": "Customer Groups",
                                    "clusters": "Customer Group Magnitude"},
                           inplace=True)
```

In [158]:
```
CustomerMagnitue_df
```

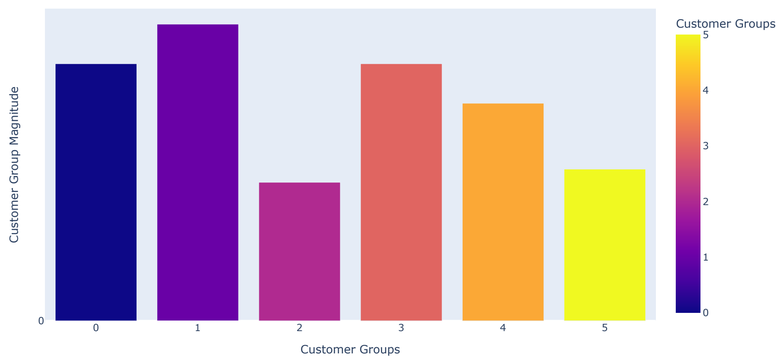
Out[158]:

	Customer Groups	Customer Group Magnitude
0	1	45
1	0	39
2	3	39
3	4	33
4	5	23
5	2	21

In [159]:
```
fig = px.bar(CustomerMagnitue_df, x="Customer Groups",
             y="Customer Group Magnitude",
             color = "Customer Groups",
             category_orders = {"Customer Groups": ["0", "1", "2", "3"]})

fig.update_layout(xaxis = dict(
        tickmode = 'linear',
        tick0 = 1,
        dtick = 1),
                  yaxis = dict(
        tickmode = 'linear',
        tick0 = 1000,
        dtick = 1000))

fig.show()
```



In [ ]:
```

```