```
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
```

```
data= pd.read_csv("Mall_Customers.csv")
```

```
data.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

```
data.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
```

```
data.isna().sum()
```

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

```
from sklearn.preprocessing import LabelEncoder
le= LabelEncoder()
```

```
data['Gender']= le.fit_transform(data['Gender'])
```

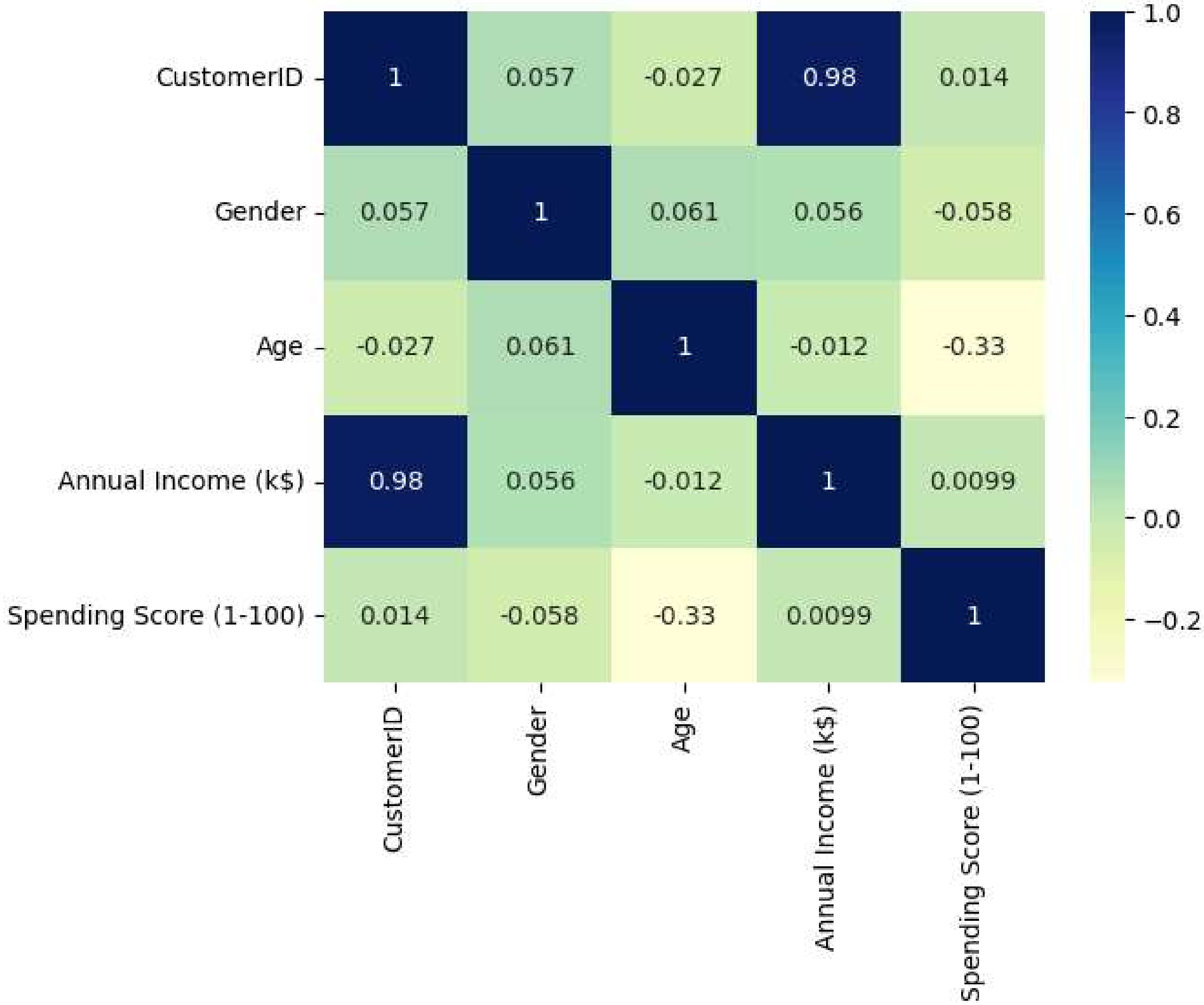
```
data.head()
```

	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

```
import seaborn as sns
```

```
sns.heatmap(data.corr(), cmap= 'YlGnBu', annot= True)
```

```
<Axes: >
```



Age and spending score is found to be moderately negatively correlated. Which means young customers are found to be spending less as compare to older customers.

```
copy_data= data.copy()
customer= copy_data.pop('CustomerID')
copy_data
```

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

200 rows × 4 columns

```
from sklearn.preprocessing import StandardScaler
scaler= StandardScaler()
```

```
scaled_data= scaler.fit_transform(copy_data)
scaled_data
```

```
array([[ 1.12815215, -1.42456879, -1.73899919, -0.43480148],
       [ 1.12815215, -1.28103541, -1.73899919,  1.19570407],
       [-0.88640526, -1.3528021 , -1.70082976, -1.71591298],
       [-0.88640526, -1.13750203, -1.70082976,  1.04041783],
       [-0.88640526, -0.56336851, -1.66266033, -0.39597992],
       [-0.88640526, -1.20926872, -1.66266033,  1.00159627],
       [-0.88640526, -0.27630176, -1.62449091, -1.71591298],
       [-0.88640526, -1.13750203, -1.62449091,  1.70038436],
       [ 1.12815215,  1.80493225, -1.58632148, -1.83237767],
       [-0.88640526, -0.6351352 , -1.58632148,  0.84631002],
       [ 1.12815215,  2.02023231, -1.58632148, -1.4053405 ],
       [-0.88640526, -0.27630176, -1.58632148,  1.89449216],
       [-0.88640526,  1.37433211, -1.54815205, -1.36651894],
       [-0.88640526, -1.06573534, -1.54815205,  1.04041783],
       [ 1.12815215, -0.13276838, -1.54815205, -1.44416206],
       [ 1.12815215, -1.20926872, -1.54815205,  1.11806095],
       [-0.88640526, -0.27630176, -1.50998262, -0.59008772],
       [ 1.12815215, -1.3528021 , -1.50998262,  0.61338066],
       [ 1.12815215,  0.94373197, -1.43364376, -0.82301709],
       [-0.88640526, -0.27630176, -1.43364376,  1.8556706 ],
       [ 1.12815215, -0.27630176, -1.39547433, -0.59008772],
       [ 1.12815215, -0.99396865, -1.39547433,  0.88513158],
       [-0.88640526,  0.51313183, -1.3573049 , -1.75473454],
```

```
       [ 1.12815215, -0.56336851, -1.3573049 ,  0.88513158],
       [-0.88640526,  1.08726535, -1.24279661, -1.4053405 ],
       [ 1.12815215, -0.70690189, -1.24279661,  1.23452563],
       [-0.88640526,  0.44136514, -1.24279661, -0.7065524 ],
       [ 1.12815215, -0.27630176, -1.24279661,  0.41927286],
       [-0.88640526,  0.08253169, -1.20462718, -0.74537397],
       [-0.88640526, -1.13750203, -1.20462718,  1.42863343],
       [ 1.12815215,  1.51786549, -1.16645776, -1.7935561 ],
       [-0.88640526, -1.28103541, -1.16645776,  0.88513158],
       [ 1.12815215,  1.01549866, -1.05194947, -1.7935561 ],
       [ 1.12815215, -1.49633548, -1.05194947,  1.62274124],
       [-0.88640526,  0.7284319 , -1.05194947, -1.4053405 ],
       [-0.88640526, -1.28103541, -1.05194947,  1.19570407],
       [-0.88640526,  0.22606507, -1.01378004, -1.28887582],
       [-0.88640526, -0.6351352 , -1.01378004,  0.88513158],
       [-0.88640526, -0.20453507, -0.89927175, -0.93948177],
       [-0.88640526, -1.3528021 , -0.89927175,  0.96277471],
       [-0.88640526,  1.87669894, -0.86110232, -0.59008772],
       [ 1.12815215, -1.06573534, -0.86110232,  1.62274124],
       [ 1.12815215,  0.65666521, -0.82293289, -0.55126616],
       [-0.88640526, -0.56336851, -0.82293289,  0.41927286],
       [-0.88640526,  0.7284319 , -0.82293289, -0.86183865],
       [-0.88640526, -1.06573534, -0.82293289,  0.5745591 ],
       [-0.88640526,  0.80019859, -0.78476346,  0.18634349],
       [-0.88640526, -0.85043527, -0.78476346, -0.12422899],
       [-0.88640526, -0.70690189, -0.78476346, -0.3183368 ],
       [-0.88640526, -0.56336851, -0.78476346, -0.3183368 ],
       [-0.88640526,  0.7284319 , -0.70842461,  0.06987881],
       [ 1.12815215, -0.41983513, -0.70842461,  0.38045129],
       [-0.88640526, -0.56336851, -0.67025518,  0.14752193],
       [ 1.12815215,  1.4460988 , -0.67025518,  0.38045129],
       [-0.88640526,  0.80019859, -0.67025518, -0.20187212],
       [ 1.12815215,  0.58489852, -0.67025518, -0.35715836],
       [-0.88640526,  0.87196528, -0.63208575, -0.00776431],
       [ 1.12815215,  2.16376569, -0.63208575, -0.16305055],
```

```
data2= pd.DataFrame(scaled_data, columns= ['Gender', 'Age', 'Annual Income (k$)', 'Spendi
data2
```

	Gender	Age	Annual Income (k$)	Spending Score (1-100)
0	1.128152	-1.424569	-1.738999	-0.434801
1	1.128152	-1.281035	-1.738999	1.195704
2	-0.886405	-1.352802	-1.700830	-1.715913
3	-0.886405	-1.137502	-1.700830	1.040418
4	-0.886405	-0.563369	-1.662660	-0.395980
...	...	...	...	...
195	-0.886405	-0.276302	2.268791	1.118061
196	-0.886405	0.441365	2.497807	-0.861839
197	1.128152	-0.491602	2.497807	0.923953
198	1.128152	-0.491602	2.917671	-1.250054
199	1.128152	-0.635135	2.917671	1.273347

200 rows × 4 columns

```
from sklearn.cluster import KMeans
```

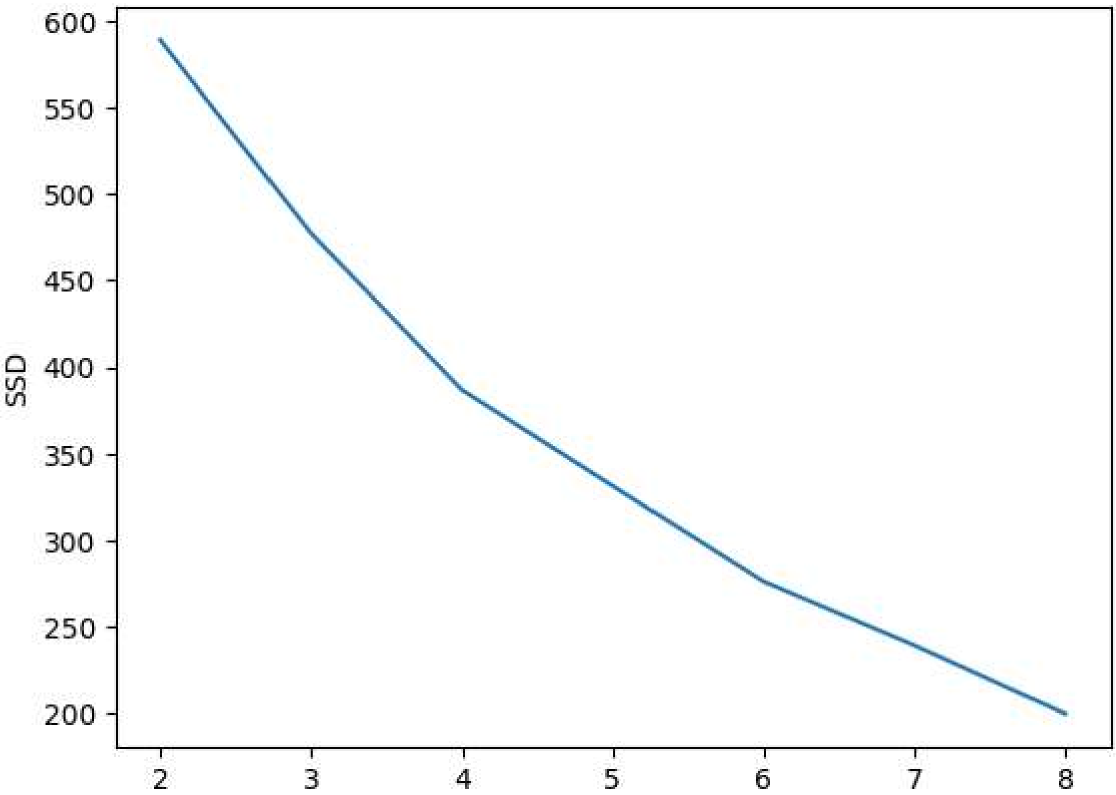
```
clusters= list(range(2,9))
ssd= []

for NumClusters in clusters:
    test= KMeans(n_clusters= NumClusters, max_iter= 150)
    test.fit(data2)
    ssd.append(test.inertia_)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning
  warnings.warn(
```

```
plt.plot(clusters, ssd)
plt.ylabel('SSD')
```

```
Text(0, 0.5, 'SSD')
```




```
model= KMeans(n_clusters= 4, max_iter= 150, random_state=15)
model.fit(data2)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning
  warnings.warn(
```

```
KMeans
KMeans(max_iter=150, n_clusters=4, random_state=15)
```

```
model.labels_
```

```
array([2, 2, 0, 3, 0, 3, 0, 3, 1, 3, 1, 3, 0, 3, 1, 2, 0, 2, 1, 3, 1, 2,
       0, 2, 0, 2, 0, 2, 0, 3, 1, 3, 1, 2, 0, 3, 0, 3, 0, 3, 0, 2, 1, 3,
       0, 3, 0, 3, 3, 3, 0, 2, 3, 1, 0, 1, 0, 1, 3, 1, 1, 2, 0, 0, 1, 2,
       0, 0, 2, 3, 1, 0, 0, 0, 1, 2, 0, 1, 3, 0, 1, 2, 1, 0, 3, 1, 0, 3,
       3, 0, 0, 2, 1, 0, 3, 2, 0, 3, 1, 2, 3, 0, 1, 2, 1, 3, 0, 1, 1, 1,
       1, 3, 0, 2, 3, 3, 0, 0, 0, 0, 2, 0, 3, 2, 3, 3, 1, 2, 1, 2, 1, 2,
       3, 3, 1, 3, 0, 2, 1, 3, 0, 2, 3, 3, 1, 2, 1, 3, 0, 2, 1, 2, 0, 3,
       0, 3, 1, 3, 1, 3, 0, 3, 1, 3, 1, 3, 1, 3, 0, 2, 1, 2, 1, 2, 0, 3,
       1, 2, 1, 2, 0, 3, 1, 3, 0, 2, 0, 2, 0, 3, 0, 3, 1, 3, 0, 3, 0, 2,
       1, 2], dtype=int32)
```

```
data['Labels']= model.labels_
```

```
data.head()
```

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

```
data['Labels'].value_counts(ascending= True)
```

```
Labels
2    40
1    48
0    56
3    56
Name: count, dtype: int64
```