In [1]:

```python
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib as plt
```

In [2]:

```python
df  = pd.read_csv(r"C:\Users\amits\Downloads\archive (26)\Mall_Customers.csv")
```

In [3]:

```python
# Data Preprocessing
```

In [4]:

```python
df.head()
```

Out[4]:

	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

In [5]:

```python
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
```

In [6]:

```python
df.describe()
```

Out[6]:

	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

In [7]:

```python
# Checking if there is null values present
```

In [8]:

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

Out[8]:

```
CustomerID                0
Gender                    0
Age                       0
```

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

In [9]:
```
df.shape
```

Out[9]:
```
(200, 5)
```

In [10]:
```
df.head()
```

Out[10]:

	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

In [11]:
```
df = df.drop('CustomerID', axis=1)
```

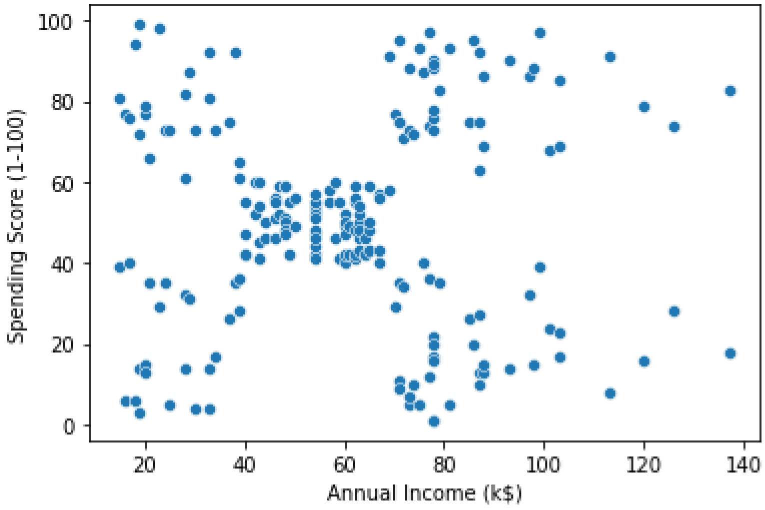
In [12]:
```
# Exploratory Data Analysis
```

In [13]:
```
df['Gender'].value_counts()
```

Out[13]:
```
Gender
Female    112
Male       88
Name: count, dtype: int64
```

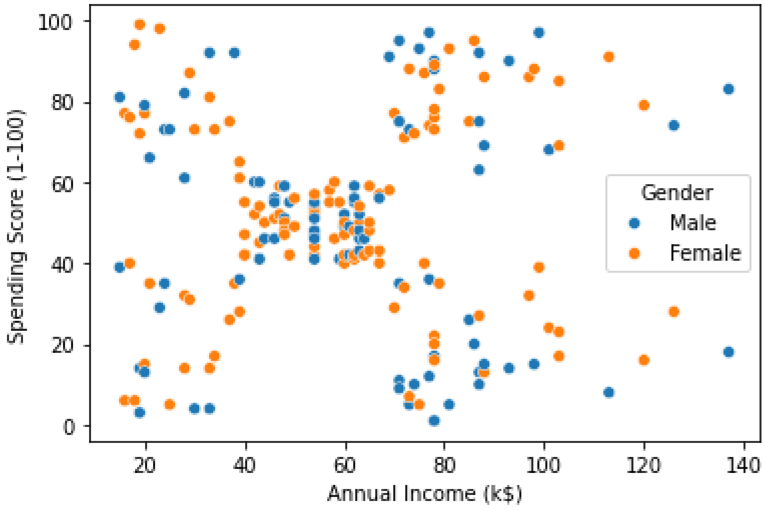
In [14]:
```
sns.scatterplot(x = df['Annual Income (k$)'], y= df['Spending Score (1-100)'])
```

Out[14]:
```
<Axes: xlabel='Annual Income (k$)', ylabel='Spending Score (1-100)'>
```



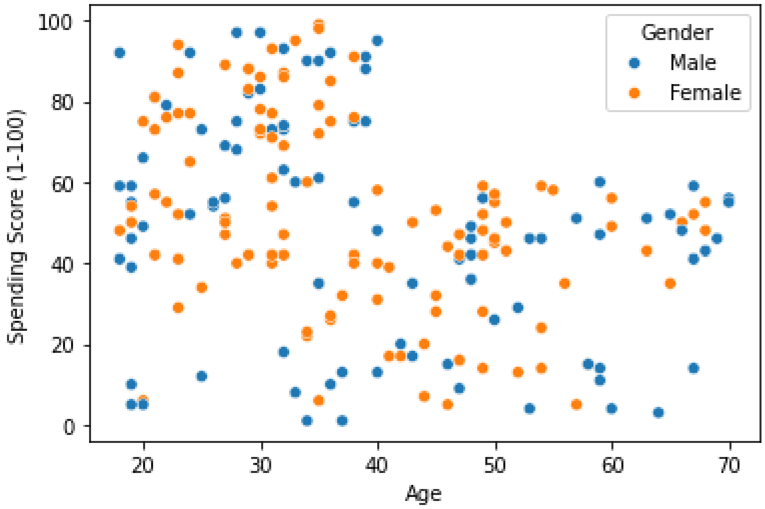
In [15]:
```
sns.scatterplot(x = df['Annual Income (k$)'], y= df['Spending Score (1-100)'], hue = df[
```

Out[15]:
```
<Axes: xlabel='Annual Income (k$)', ylabel='Spending Score (1-100)'>
```



```
In [16]: sns.scatterplot(x = df['Age'], y =df['Spending Score (1-100)'], hue = df['Gender'])
```

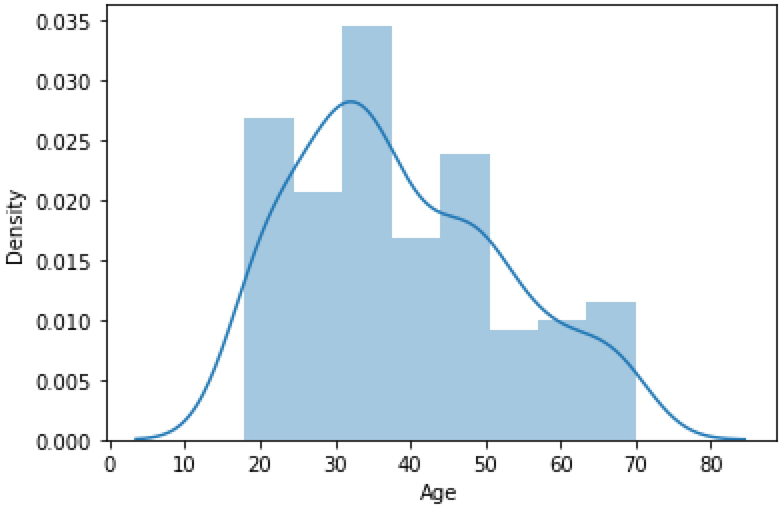
```
Out[16]: <Axes: xlabel='Age', ylabel='Spending Score (1-100)'>
```



```
In [17]: sns.distplot(df['Age'], kde =True)
```

```
C:\Users\amits\AppData\Local\Temp/ipykernel_20348/2658259778.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df['Age'], kde =True)
```

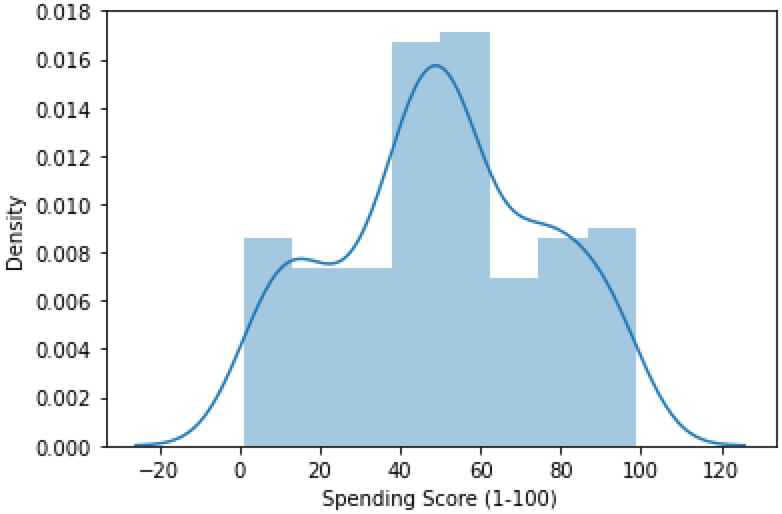
```
Out[17]: <Axes: xlabel='Age', ylabel='Density'>
```



```
In [18]: sns.distplot(df['Spending Score (1-100)'], kde =True)
```

```
C:\Users\amits\AppData\Local\Temp/ipykernel_20348/76464220.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df['Spending Score (1-100)'], kde =True)
```

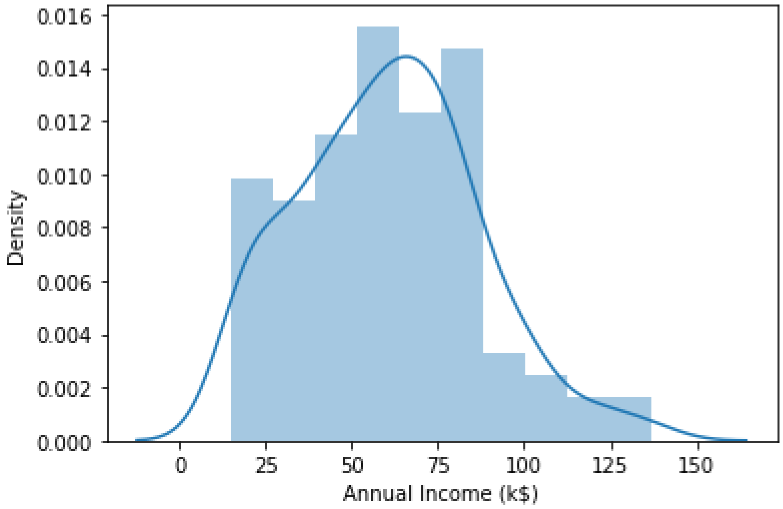
```
Out[18]: <Axes: xlabel='Spending Score (1-100)', ylabel='Density'>
```



```
In [19]: sns.distplot(df['Annual Income (k$)'], kde =True)
```

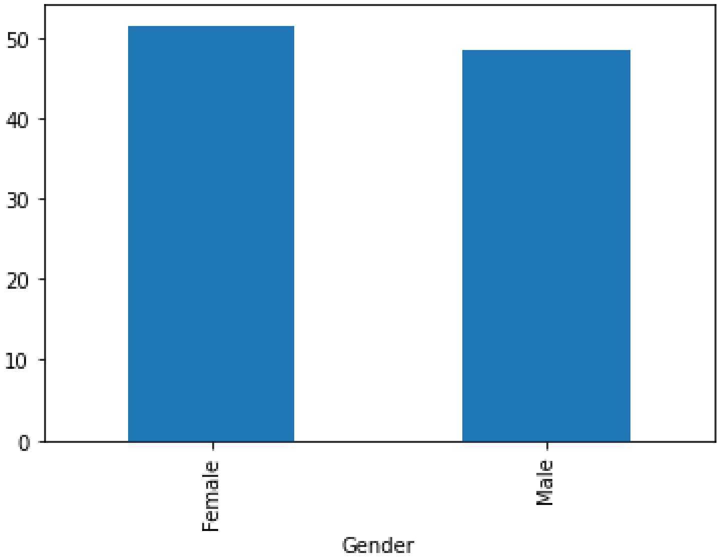
```
C:\Users\amits\AppData\Local\Temp/ipykernel_20348/3193301502.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df['Annual Income (k$)'], kde =True)
```

```
Out[19]: <Axes: xlabel='Annual Income (k$)', ylabel='Density'>
```



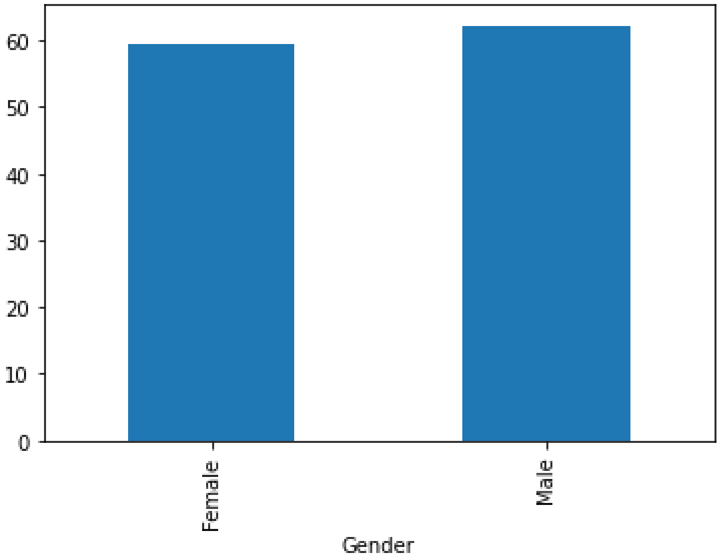
In [20]:
```
df.groupby('Gender')['Spending Score (1-100)'].mean().plot(kind = 'bar')
```

Out[20]:
```
<Axes: xlabel='Gender'>
```



In [21]:
```
df.groupby('Gender')['Annual Income (k$)'].mean().plot(kind = 'bar')
```

Out[21]:
```
<Axes: xlabel='Gender'>
```



In [22]:
```
# Feature Engineering
# Handling categorical column
```

In [23]:

```
df['Gender'].unique()
```

Out[23]:

```
array(['Male', 'Female'], dtype=object)
```

In [24]:

```
df['Gender'] = df['Gender'].map({'Male':0, 'Female':1})
```

In [25]:

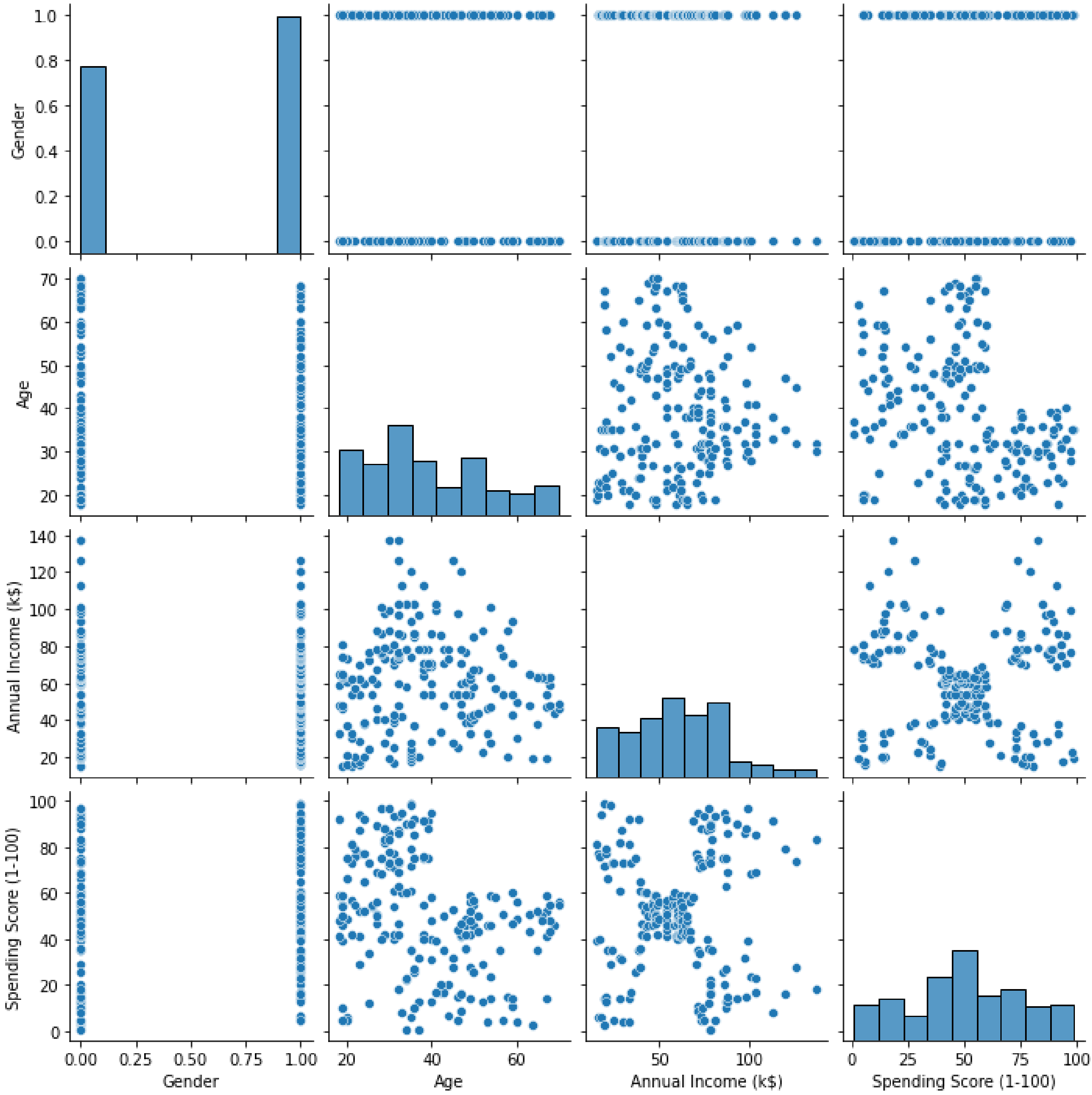
```
# Pair Plot
```

In [26]:

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

Out[26]:

```
<seaborn.axisgrid.PairGrid at 0x241f8ee3340>
```



In [27]:

```
# Correlation
```

In [28]:

```
df.corr()
```

Out[28]:

	Gender	Age	Annual Income (k$)	Spending Score (1-100)
Gender	1.000000	-0.060867	-0.056410	0.058109

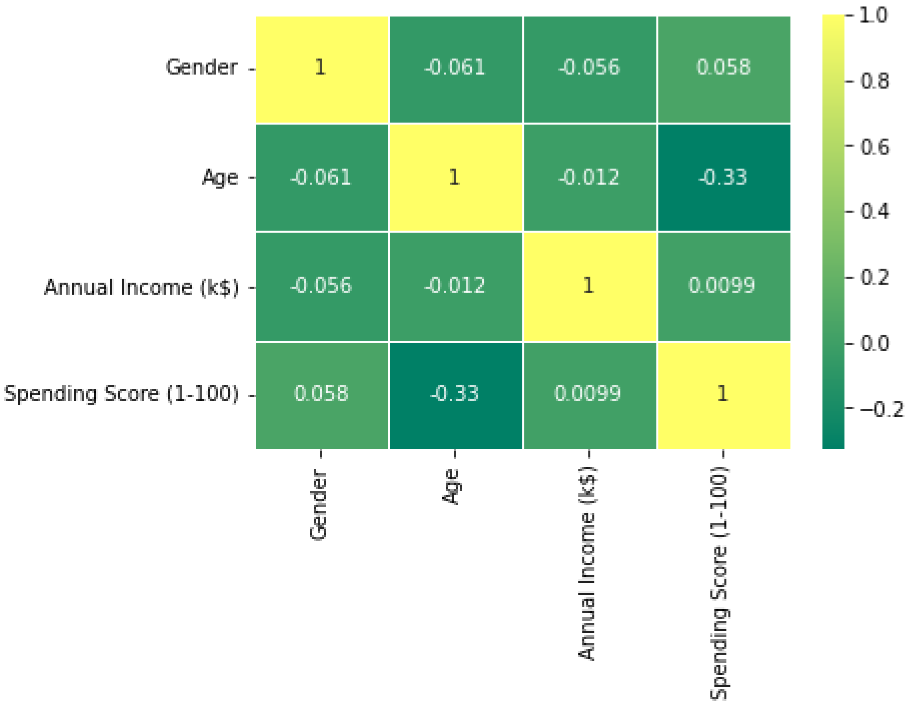
Age	-0.060867	1.000000	-0.012398	-0.327227
Annual Income (k$)	-0.056410	-0.012398	1.000000	0.009903
Spending Score (1-100)	0.058109	-0.327227	0.009903	1.000000

In [29]:
```
# Matrix Plot
```

In [30]:
```
# Heatmap
```

In [31]:
```
sns.heatmap(df.corr(), annot = True, linewidth=0.5, cmap = 'summer')
```

Out[31]:
```
<Axes: >
```



In [32]:
```
# Choosing the Age, Annual Income And Spening Score Features
```

In [33]:
```
df.head()
```

Out[33]:

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

In [34]:
```
x = df.iloc[:,[1,2,3]]
```

In [35]:
```
print(x)
```

```
     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 x 3 columns]
```

In [36]:

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
scaled_x = scaler.fit_transform(x)
```

In [37]:

```
scaled_x
```

Out[37]:

```
array([[-1.42456879, -1.73899919, -0.43480148],
       [-1.28103541, -1.73899919,  1.19570407],
       [-1.3528021 , -1.70082976, -1.71591298],
       [-1.13750203, -1.70082976,  1.04041783],
       [-0.56336851, -1.66266033, -0.39597992],
       [-1.20926872, -1.66266033,  1.00159627],
       [-0.27630176, -1.62449091, -1.71591298],
       [-1.13750203, -1.62449091,  1.70038436],
       [ 1.80493225, -1.58632148, -1.83237767],
       [-0.6351352 , -1.58632148,  0.84631002],
       [ 2.02023231, -1.58632148, -1.4053405 ],
       [-0.27630176, -1.58632148,  1.89449216],
       [ 1.37433211, -1.54815205, -1.36651894],
       [-1.06573534, -1.54815205,  1.04041783],
       [-0.13276838, -1.54815205, -1.44416206],
       [-1.20926872, -1.54815205,  1.11806095],
       [-0.27630176, -1.50998262, -0.59008772],
       [-1.3528021 , -1.50998262,  0.61338066],
       [ 0.94373197, -1.43364376, -0.82301709],
       [-0.27630176, -1.43364376,  1.8556706 ],
       [-0.27630176, -1.39547433, -0.59008772],
       [-0.99396865, -1.39547433,  0.88513158],
       [ 0.51313183, -1.3573049 , -1.75473454],
       [-0.56336851, -1.3573049 ,  0.88513158],
       [ 1.08726535, -1.24279661, -1.4053405 ],
       [-0.70690189, -1.24279661,  1.23452563],
       [ 0.44136514, -1.24279661, -0.7065524 ],
       [-0.27630176, -1.24279661,  0.41927286],
       [ 0.08253169, -1.20462718, -0.74537397],
       [-1.13750203, -1.20462718,  1.42863343],
       [ 1.51786549, -1.16645776, -1.7935561 ],
       [-1.28103541, -1.16645776,  0.88513158],
       [ 1.01549866, -1.05194947, -1.7935561 ],
       [-1.49633548, -1.05194947,  1.62274124],
       [ 0.7284319 , -1.05194947, -1.4053405 ],
       [-1.28103541, -1.05194947,  1.19570407],
       [ 0.22606507, -1.01378004, -1.28887582],
       [-0.6351352 , -1.01378004,  0.88513158],
       [-0.20453507, -0.89927175, -0.93948177],
       [-1.3528021 , -0.89927175,  0.96277471],
       [ 1.87669894, -0.86110232, -0.59008772],
       [-1.06573534, -0.86110232,  1.62274124],
       [ 0.65666521, -0.82293289, -0.55126616],
       [-0.56336851, -0.82293289,  0.41927286],
       [ 0.7284319 , -0.82293289, -0.86183865],
       [-1.06573534, -0.82293289,  0.5745591 ],
       [ 0.80019859, -0.78476346,  0.18634349],
       [-0.85043527, -0.78476346, -0.12422899],
       [-0.70690189, -0.78476346, -0.3183368 ],
       [-0.56336851, -0.78476346, -0.3183368 ],
       [ 0.7284319 , -0.70842461,  0.06987881],
       [-0.41983513, -0.70842461,  0.38045129],
       [-0.56336851, -0.67025518,  0.14752193],
       [ 1.4460988 , -0.67025518,  0.38045129],
       [ 0.80019859, -0.67025518, -0.20187212],
       [ 0.58489852, -0.67025518, -0.35715836],
       [ 0.87196528, -0.63208575, -0.00776431],
       [ 2.16376569, -0.63208575, -0.16305055],
       [-0.85043527, -0.55574689,  0.03105725],
       [ 1.01549866, -0.55574689, -0.16305055],
       [ 2.23553238, -0.55574689,  0.22516505],
       [-1.42456879, -0.55574689,  0.18634349],
       [ 2.02023231, -0.51757746,  0.06987881],
       [ 1.08726535, -0.51757746,  0.34162973],
       [ 1.73316556, -0.47940803,  0.03105725],
       [-1.49633548, -0.47940803,  0.34162973],
       [ 0.29783176, -0.47940803, -0.00776431],
       [ 2.091999  , -0.47940803, -0.08540743],
       [-1.42456879, -0.47940803,  0.34162973],
       [-0.49160182, -0.47940803, -0.12422899],
       [ 2.23553238, -0.4412386 ,  0.18634349],
       [ 0.58489852, -0.4412386 , -0.3183368 ],
       [ 1.51786549, -0.40306917, -0.04658587],
       [ 1.51786549, -0.40306917,  0.22516505],
       [ 1.4460988 , -0.25039146, -0.12422899],
       [-0.92220196, -0.25039146,  0.14752193],
       [ 0.44136514, -0.25039146,  0.10870037],
       [ 0.08253169, -0.25039146, -0.08540743],
       [-1.13750203, -0.25039146,  0.06987881],
       [ 0.7284319 , -0.25039146, -0.3183368 ],
       [ 1.30256542, -0.25039146,  0.03105725],
       [-0.06100169, -0.25039146,  0.18634349],
       [ 2.02023231, -0.25039146, -0.35715836],
       [ 0.51313183, -0.25039146, -0.24069368],
       [-1.28103541, -0.25039146,  0.26398661],
       [ 0.65666521, -0.25039146, -0.16305055],
       [ 1.15903204, -0.13588317,  0.30280817],
       [-1.20926872, -0.13588317,  0.18634349],
       [-0.34806844, -0.09771374,  0.38045129],
       [ 0.80019859, -0.09771374, -0.16305055],
       [ 2.091999  , -0.05954431,  0.18634349],
       [-1.49633548, -0.05954431, -0.35715836],
       [ 0.65666521, -0.02137488, -0.04658587],
       [ 0.08253169, -0.02137488, -0.39597992],
       [-0.49160182, -0.02137488, -0.3183368 ],
       [-1.06573534, -0.02137488,  0.06987881],
       [ 0.58489852, -0.02137488, -0.12422899],
       [-0.85043527, -0.02137488, -0.00776431],
       [ 0.65666521,  0.01679455, -0.3183368 ],
       [-1.3528021 ,  0.01679455, -0.04658587],
       [-1.13750203,  0.05496398, -0.35715836],
       [ 0.7284319 ,  0.05496398, -0.08540743],
       [ 2.02023231,  0.05496398,  0.34162973],
       [-0.92220196,  0.05496398,  0.18634349],
       [ 0.7284319 ,  0.05496398,  0.22516505],
       [-1.28103541,  0.05496398, -0.3183368 ],
       [ 1.94846562,  0.09313341, -0.00776431],
       [ 1.08726535,  0.09313341, -0.16305055],
       [ 2.091999  ,  0.09313341, -0.27951524],
       [ 1.94846562,  0.09313341, -0.08540743],
       [ 1.87669894,  0.09313341,  0.06987881],
       [-1.42456879,  0.09313341,  0.14752193],
       [-0.06100169,  0.13130284, -0.3183368 ],
       [-1.42456879,  0.13130284, -0.16305055],
       [-1.49633548,  0.16947227, -0.08540743],
       [-1.42456879,  0.16947227, -0.00776431],
       [ 1.73316556,  0.16947227, -0.27951524],
       [ 0.7284319 ,  0.16947227,  0.34162973],
       [ 0.87196528,  0.24581112, -0.27951524],
       [ 0.80019859,  0.24581112,  0.26398661],
       [-0.85043527,  0.24581112,  0.22516505],
       [-0.06100169,  0.24581112, -0.39597992],
       [ 0.08253169,  0.32214998,  0.30280817],
       [ 0.010765  ,  0.32214998,  1.58391968],
       [-1.13750203,  0.36031941, -0.82301709],
       [-0.56336851,  0.36031941,  1.04041783],
       [ 0.29783176,  0.39848884, -0.59008772],
       [ 0.08253169,  0.39848884,  1.73920592],
       [ 1.4460988 ,  0.39848884, -1.52180518],
       [-0.06100169,  0.39848884,  0.96277471],
       [ 0.58489852,  0.39848884, -1.5994483 ],
       [ 0.010765  ,  0.39848884,  0.96277471],
       [-0.99396865,  0.43665827, -0.62890928],
       [-0.56336851,  0.43665827,  0.80748846],
       [-1.3528021 ,  0.4748277 , -1.75473454],
       [-0.70690189,  0.4748277 ,  1.46745499],
       [ 0.36959845,  0.4748277 , -1.67709142],
       [-0.49160182,  0.4748277 ,  0.88513158],
       [-1.42456879,  0.51299713, -1.56062674],
       [-0.27630176,  0.51299713,  0.84631002],
       [ 1.30256542,  0.55116656, -1.75473454],
       [-0.49160182,  0.55116656,  1.6615628 ],
       [-0.77866858,  0.58933599, -0.39597992],
       [-0.49160182,  0.58933599,  1.42863343],
       [-0.99396865,  0.62750542, -1.48298362],
       [-0.77866858,  0.62750542,  1.81684904],
       [ 0.65666521,  0.62750542, -0.55126616],
       [-0.49160182,  0.62750542,  0.92395314],
       [-0.34806844,  0.66567484, -1.09476801],
       [-0.34806844,  0.66567484,  1.54509812],
       [ 0.29783176,  0.66567484, -1.28887582],
       [ 0.010765  ,  0.66567484,  1.46745499],
       [ 0.36959845,  0.66567484, -1.17241113],
       [-0.06100169,  0.66567484,  1.00159627],
       [ 0.58489852,  0.66567484, -1.32769738],
       [-0.85043527,  0.66567484,  1.50627656],
       [-0.13276838,  0.66567484, -1.91002079],
       [-0.6351352 ,  0.66567484,  1.07923939],
       [-0.34806844,  0.66567484, -1.91002079],
       [-0.6351352 ,  0.66567484,  0.88513158],
       [ 1.23079873,  0.70384427, -0.59008772],
       [-0.70690189,  0.70384427,  1.27334719],
       [-1.42456879,  0.78018313, -1.75473454],
       [-0.56336851,  0.78018313,  1.6615628 ],
       [ 0.80019859,  0.93286085, -0.93948177],
       [-0.20453507,  0.93286085,  0.96277471],
       [ 0.22606507,  0.97103028, -1.17241113],
       [-0.41983513,  0.97103028,  1.73920592],
       [-0.20453507,  1.00919971, -0.90066021],
       [-0.49160182,  1.00919971,  0.49691598],
       [ 0.08253169,  1.00919971, -1.44416206],
       [-0.77866858,  1.00919971,  0.96277471],
       [-0.20453507,  1.00919971, -1.56062674],
       [-0.20453507,  1.00919971,  1.62274124],
       [ 0.94373197,  1.04736914, -1.44416206],
       [-0.6351352 ,  1.04736914,  1.38981187],
       [ 1.37433211,  1.04736914, -1.36651894],
       [-0.85043527,  1.04736914,  0.72984534],
       [ 1.4460988 ,  1.23821628, -1.4053405 ],
       [-0.27630176,  1.23821628,  1.54509812],
       [-0.13276838,  1.390894  , -0.7065524 ],
       [-0.49160182,  1.390894  ,  1.38981187],
       [ 0.51313183,  1.42906343, -1.36651894],
              [-0.70690189,  1.42906343,  1.46745499],
              [ 0.15429838,  1.46723286, -0.43480148],
              [-0.6351352 ,  1.46723286,  1.81684904],
              [ 1.08726535,  1.54357172, -1.01712489],
              [-0.77866858,  1.54357172,  0.69102378],
              [ 0.15429838,  1.61991057, -1.28887582],
              [-0.20453507,  1.61991057,  1.35099031],
              [-0.34806844,  1.61991057, -1.05594645],
              [-0.49160182,  1.61991057,  0.72984534],
              [-0.41983513,  2.00160487, -1.63826986],
              [-0.06100169,  2.00160487,  1.58391968],
              [ 0.58489852,  2.26879087, -1.32769738],
              [-0.27630176,  2.26879087,  1.11806095],
              [ 0.44136514,  2.49780745, -0.86183865],
              [-0.49160182,  2.49780745,  0.92395314],
              [-0.49160182,  2.91767117, -1.25005425],
              [-0.6351352 ,  2.91767117,  1.27334719]])
```

In [38]:

```
# Choosing the number of cluster
```

In [39]:

```
from sklearn.cluster import DBSCAN
```

In [40]:

```
dbscan = DBSCAN(eps = 0.5, min_samples = 5)

# Fit the model to your data
labels = dbscan.fit_predict(scaled_x)
```

In [41]:

```
labels
```

Out[41]:

```
array([-1,  0, -1,  0, -1,  0, -1, -1, -1,  0, -1, -1, -1,  0, -1,  0,  1,
        0, -1, -1,  1,  0, -1,  0, -1,  0,  1, -1,  1,  0, -1,  0, -1,  0,
       -1,  0, -1,  0,  1,  0, -1,  0,  2,  3,  2, -1,  2,  3,  3,  3,  2,
        3,  3,  2,  2,  2,  2,  2,  3,  2,  2,  3,  2,  2,  2,  3,  2,  2,
        3,  3,  2,  2,  2,  2,  2,  3,  2,  2,  3,  2,  2,  2,  2,  2,  3,
        2,  2,  3, -1,  2,  2,  3,  2,  2,  2,  3,  2,  3,  2,  3,  3,  2,
        2,  3,  2,  3,  2,  2,  2,  2,  2,  3,  2,  3,  3,  3,  2,  2,  2,
        2,  3,  2, -1,  4, -1,  4, -1,  4, -1,  4,  5,  4,  3,  4, -1,  4,
        5,  4, -1,  4, -1,  4, -1,  4, -1,  4, -1,  4, -1,  4,  5,  4,  5,
        4,  5,  4, -1,  4, -1,  4, -1,  4, -1,  4, -1,  4,  5,  4, -1, -1,
        5,  4, -1,  4, -1,  4, -1,  4, -1,  4, -1,  4, -1,  4, -1,  4, -1,
       -1, -1,  4, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1], dtype=int64)
```

In [42]:

```
labels = dbscan.labels_
labels
```

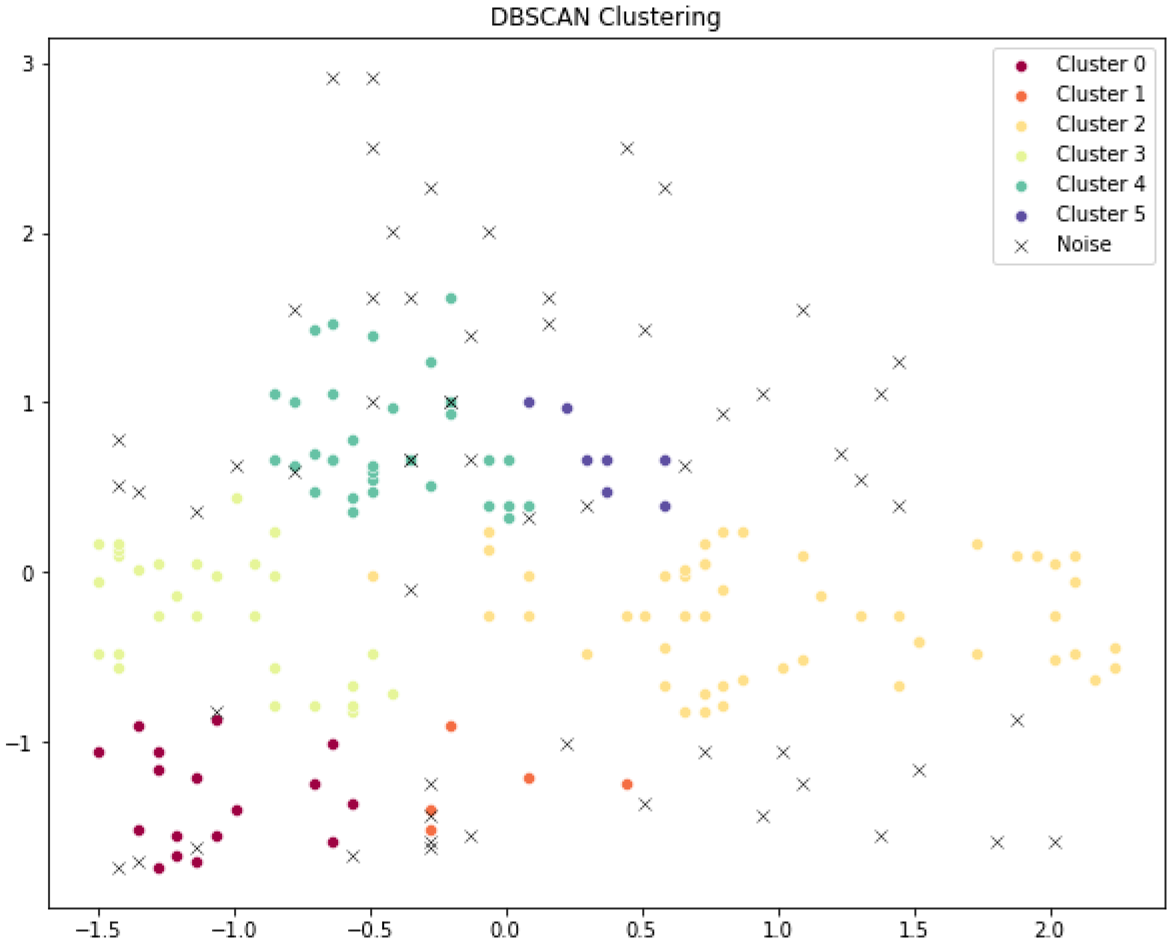
Out[42]:

```
array([-1,  0, -1,  0, -1,  0, -1, -1, -1,  0, -1, -1, -1,  0, -1,  0,  1,
        0, -1, -1,  1,  0, -1,  0, -1,  0,  1, -1,  1,  0, -1,  0, -1,  0,
       -1,  0, -1,  0,  1,  0, -1,  0,  2,  3,  2, -1,  2,  3,  3,  3,  2,
        3,  3,  2,  2,  2,  2,  2,  3,  2,  2,  3,  2,  2,  2,  3,  2,  2,
        3,  3,  2,  2,  2,  2,  2,  3,  2,  2,  3,  2,  2,  2,  2,  2,  3,
        2,  2,  3, -1,  2,  2,  3,  2,  2,  2,  3,  2,  3,  2,  3,  3,  2,
        2,  3,  2,  3,  2,  2,  2,  2,  2,  3,  2,  3,  3,  3,  2,  2,  2,
        2,  3,  2, -1,  4, -1,  4, -1,  4, -1,  4,  5,  4,  3,  4, -1,  4,
        5,  4, -1,  4, -1,  4, -1,  4, -1,  4, -1,  4, -1,  4,  5,  4,  5,
        4,  5,  4, -1,  4, -1,  4, -1,  4, -1,  4, -1,  4,  5,  4, -1, -1,
        5,  4, -1,  4, -1,  4, -1,  4, -1,  4, -1,  4, -1,  4, -1,  4, -1,
       -1, -1,  4, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1], dtype=int64)
```

In [43]:

```
import seaborn as sns
import matplotlib.pyplot as plt

# Extract unique labels, excluding noise (-1)
unique_labels = set(labels) - {-1}

plt.figure(figsize=(10,8))
```

```
# Create a list of colors for plotting
colors = [plt.cm.Spectral(each) for each in np.linspace(0, 1, len(unique_labels))]

# Create a scatter plot for each cluster
for label, color in zip(unique_labels, colors):
    cluster_points = scaled_x[labels == label]
    sns.scatterplot(x=cluster_points[:, 0], y=cluster_points[:, 1], color=color, label='

# Plot noise points
noise_points = scaled_x[labels == -1]
sns.scatterplot(x=noise_points[:, 0], y=noise_points[:, 1], color='k', marker='x', label

plt.legend()
plt.title('DBSCAN Clustering')
plt.show()
```


DBSCAN Clustering
● Cluster 0
● Cluster 1
● Cluster 2
● Cluster 3
● Cluster 4
● Cluster 5
× Noise