Python Project Mall I

May 6, 2023

1 Shopping Customer Segmentation Project

2 Segment Shopping Customer

- **Problem Statement:** Understand the target Customer for the marketing team to plan a strategy
- **Context:** Your boss wants you to identify the most important shopping groups based on income, age, and the mall shopping score.
- He wants the ideal number of groups with a label for each.

2.1 Objective Market Segmentation

- Divide your mall target market into approachable groups. Create subsets of a market bases on demographics behavioral criteria to better understand the target for marketing

2.2 The Approach

1. Perform some quick EDA (Exploratory Data Analysis)
2. Use KMEANS Clustering Algorithm to create our segment
3. Use Summary Statistics on the cluster
4. Visualize

- Importing libraries and dateset

[1]:
```
import pandas as pd # Data manipulation
import seaborn as sns # Statistical visualization library
import matplotlib.pyplot as plt # Another visualization library
from sklearn.cluster import KMeans # For create clusters
import warnings
warnings.filterwarnings("ignore")
```

[2]:
```
df = pd.read_csv("C:/Users/Drac_/OneDrive/Desktop/Python_Project_Data/
↪Mall_Customers.csv")
```

Now that we have that variable df, that is our data frame, we will now display the first five rows of our data using head function.

[3]:
```
df.head()
```

[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

3 Univariate Analysis

This for looking at one variable, and now we will look for the mean, standard deviation, etc., using the describe function

[4]:
```
df.describe()
```

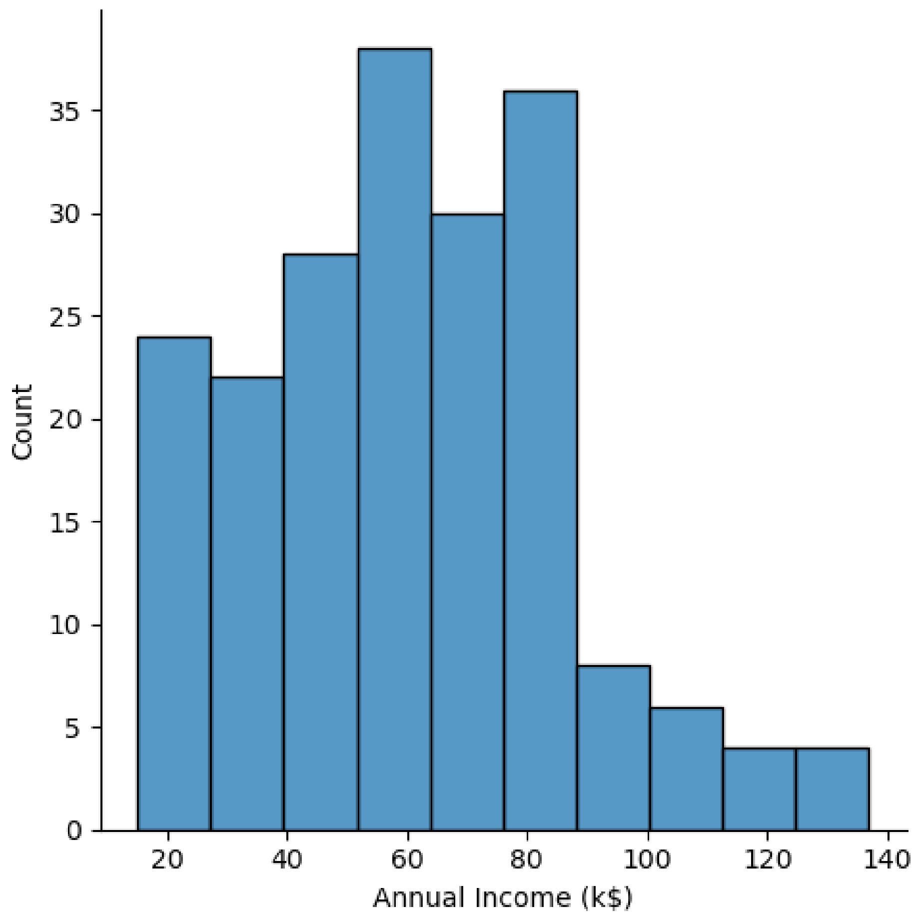
[4]:

	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

Now we are going to create a histogram, to be able to see the annual income, using seaborn library saved as sns.

[5]:
```
sns.displot(df["Annual Income (k$)"])
```

[5]: <seaborn.axisgrid.FacetGrid at 0x28a8b36f820>



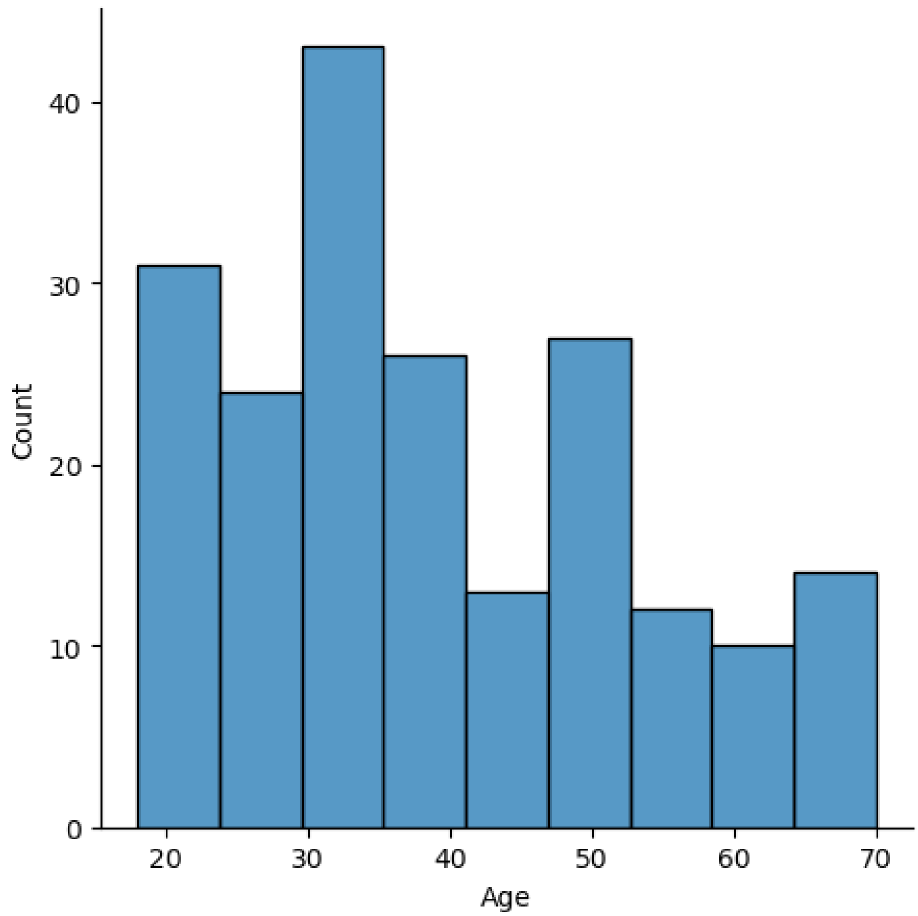
[6]:
```
df.columns
```

[6]:
```
Index(['CustomerID', 'Gender', 'Age', 'Annual Income (k$)',
       'Spending Score (1-100)'],
      dtype='object')
```

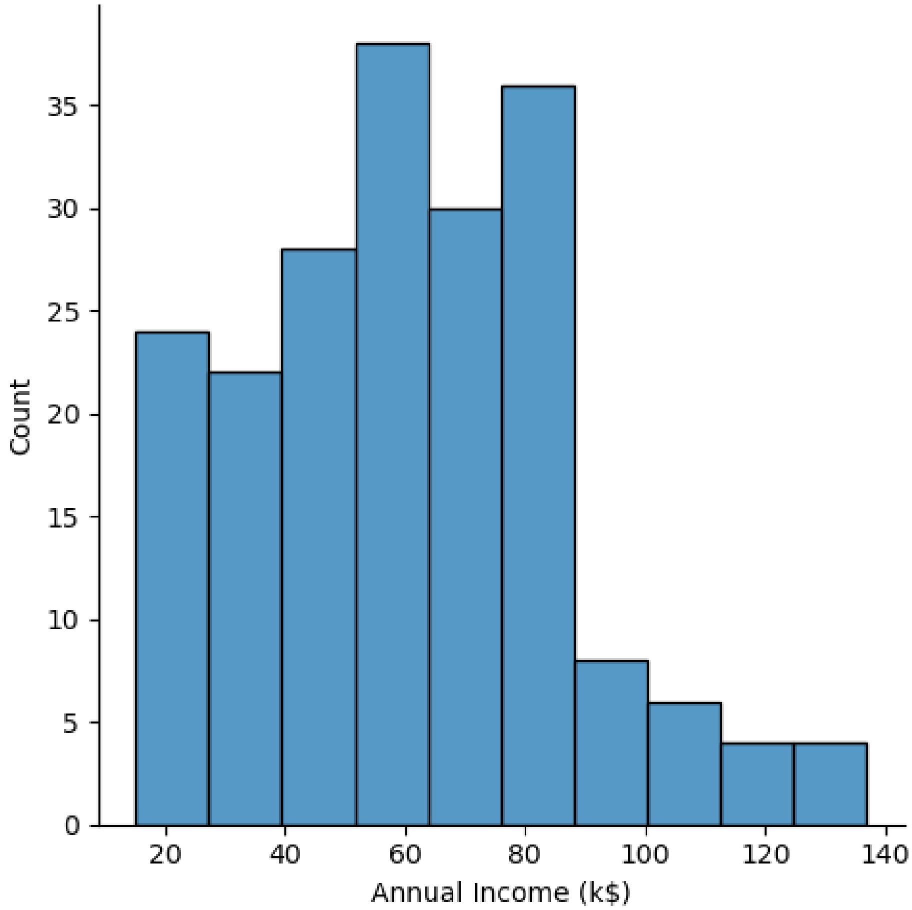
We're going to do the same, with the others columns

[7]:
```
columns = ['Age', 'Annual Income (k$)','Spending Score (1-100)']
for i in columns:
    plt.figure()
    sns.displot(df[i])
```

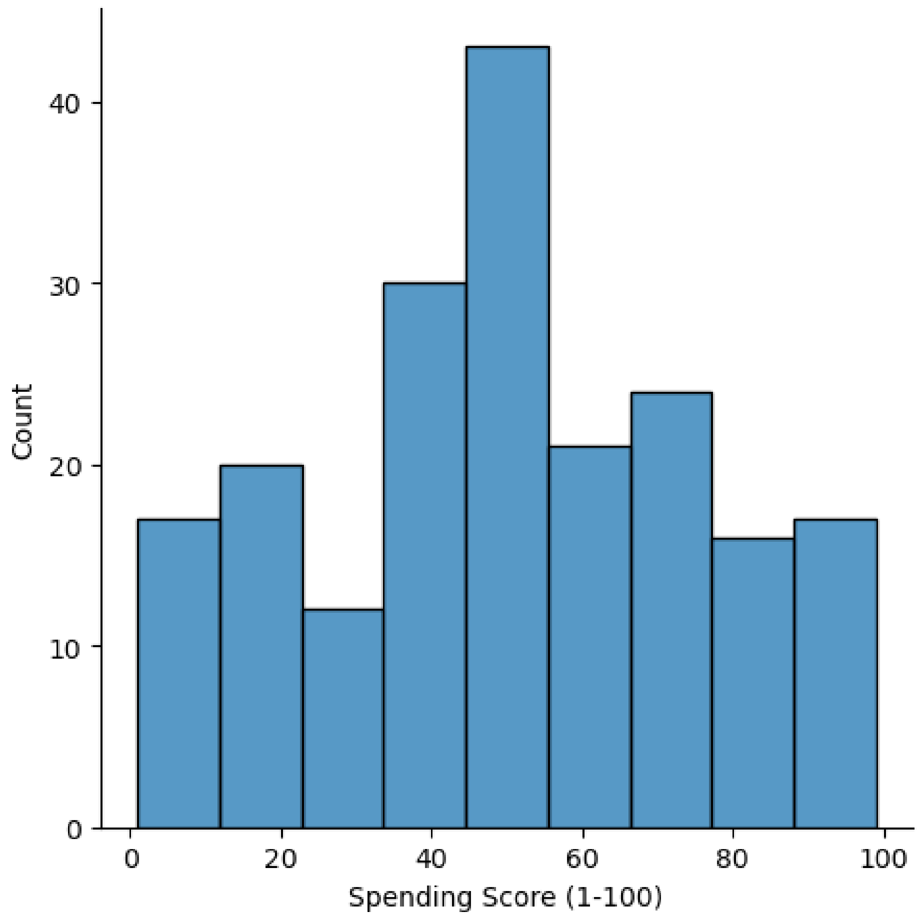
```
<Figure size 640x480 with 0 Axes>
```




```
<Figure size 640x480 with 0 Axes>
```

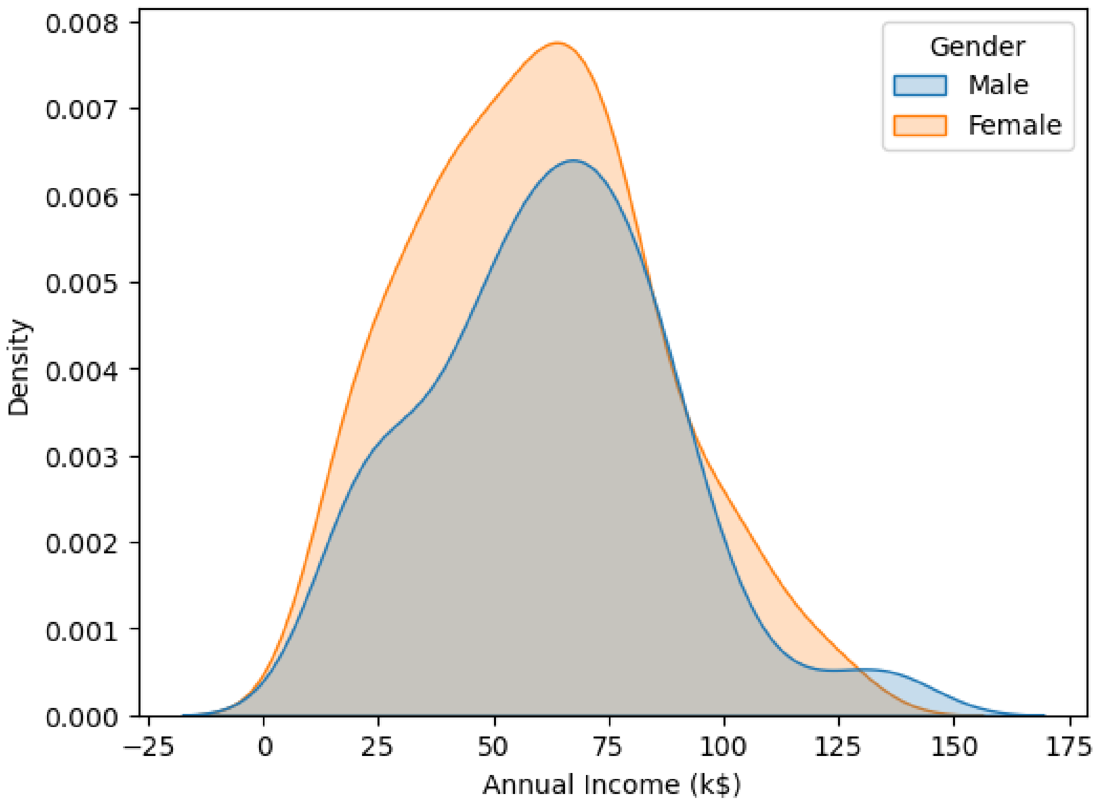



```
<Figure size 640x480 with 0 Axes>
```



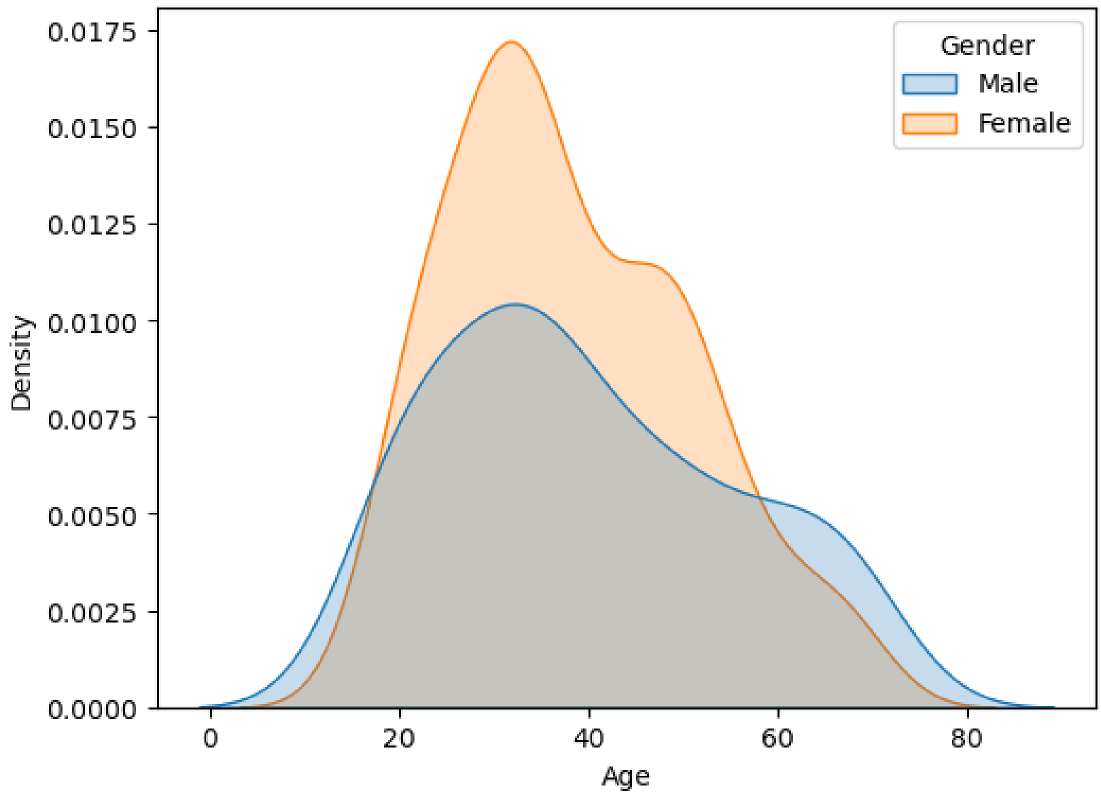
Now we will isolate the column Gender for see the difference between female and male.

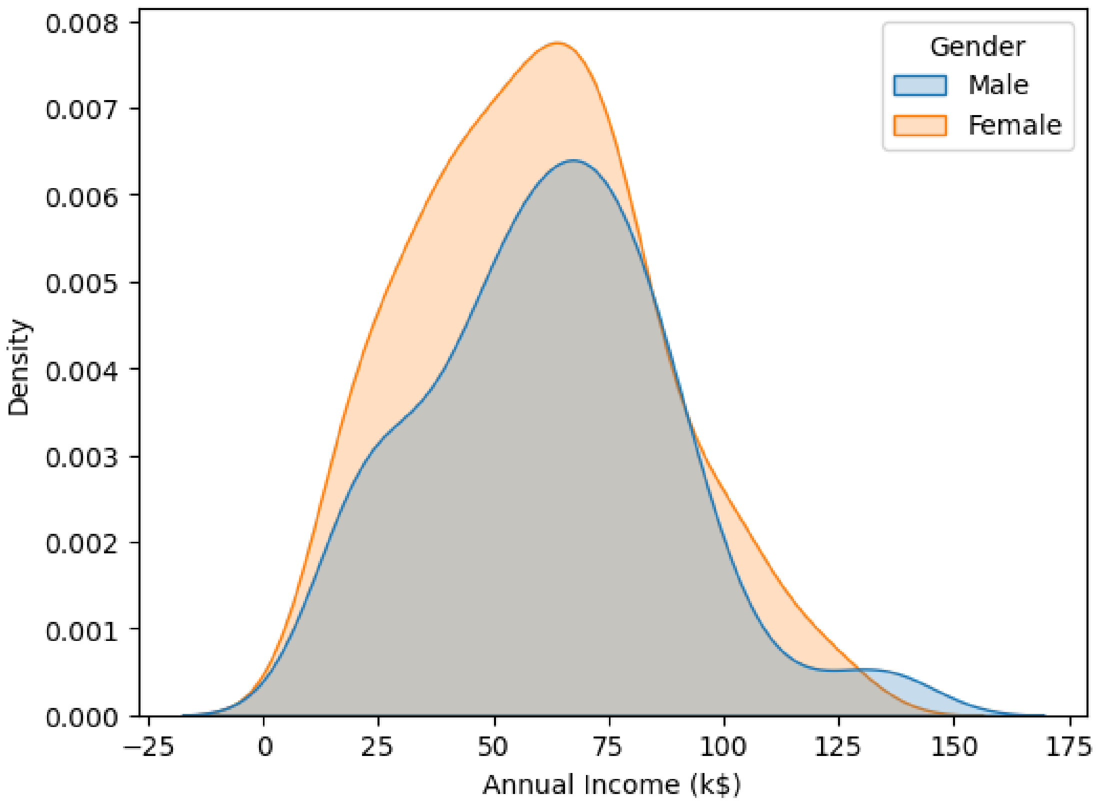
[8]:
```
sns.kdeplot(x=df["Annual Income (k$)"],shade=True,hue=df["Gender"]);
```

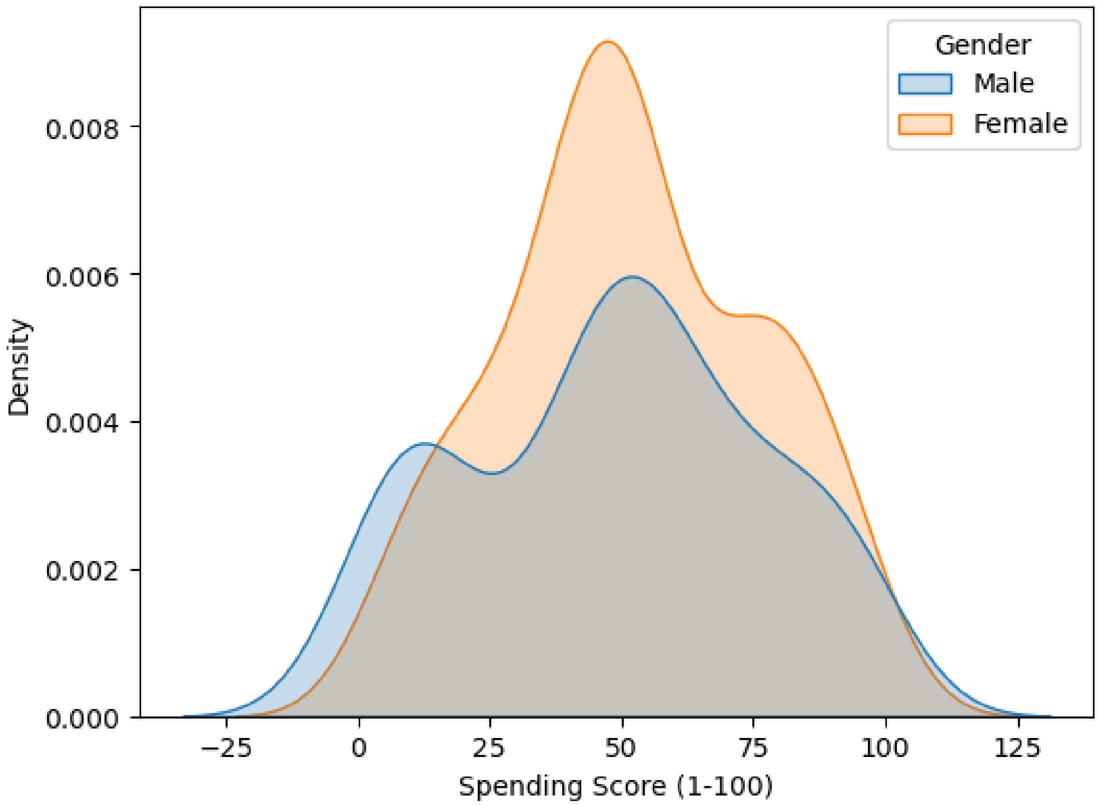


[9]:

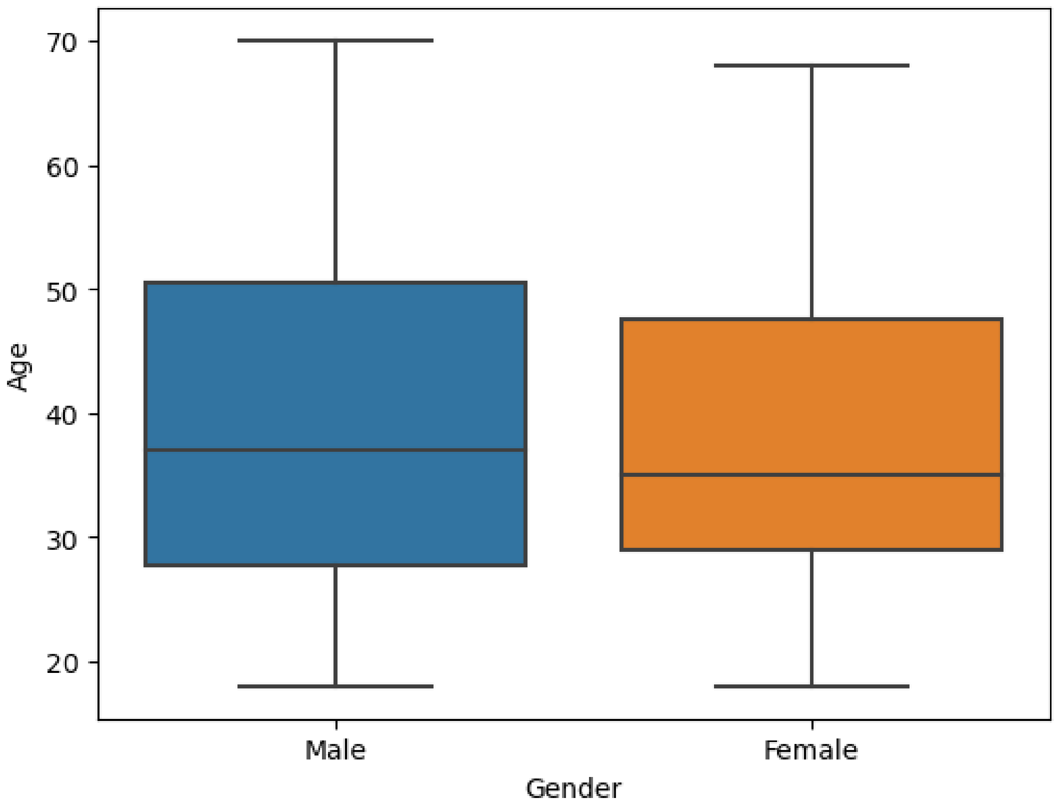
```
columns = ['Age', 'Annual Income (k$)','Spending Score (1-100)']
for i in columns:
    plt.figure()
    sns.kdeplot(x=df[i],shade=True,hue=df["Gender"]);
```


Gender
Male
Female
Density
0.0000
0.0025
0.0050
0.0075
0.0100
0.0125
0.0150
0.0175
0
20
40
60
80
Age

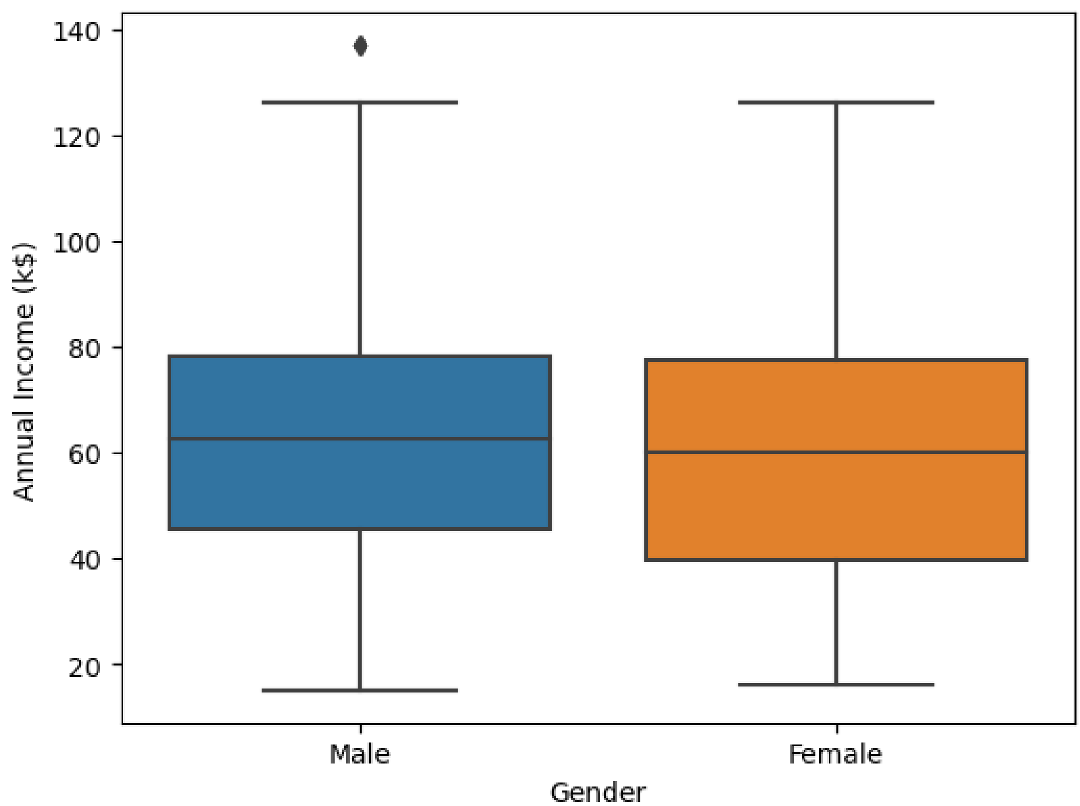

Gender
Male
Female
Density
0.000
0.001
0.002
0.003
0.004
0.005
0.006
0.007
0.008
−25
0
25
50
75
100
125
150
175
Annual Income (k$)

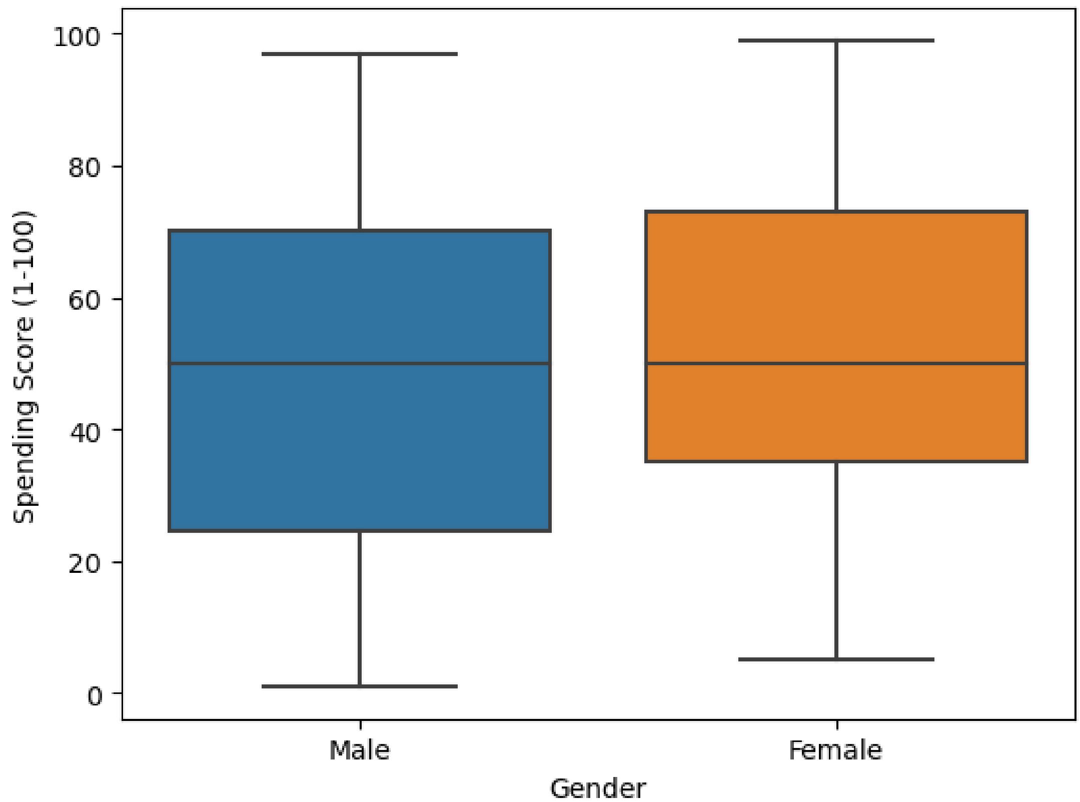


[10]:
```
columns = ['Age', 'Annual Income (k$)','Spending Score (1-100)']
for i in columns:
    plt.figure()
    sns.boxplot(data=df,x="Gender",y=df[i]);
```



70
60
50
40
30
20
Age
Male
Female
Gender


140
120
100
80
60
40
20
Annual Income (k$)
Male
Female
Gender



[11]:
```
df["Gender"].value_counts(normalize=True)
```

[11]:
```
Female    0.56
Male      0.44
Name: Gender, dtype: float64
```

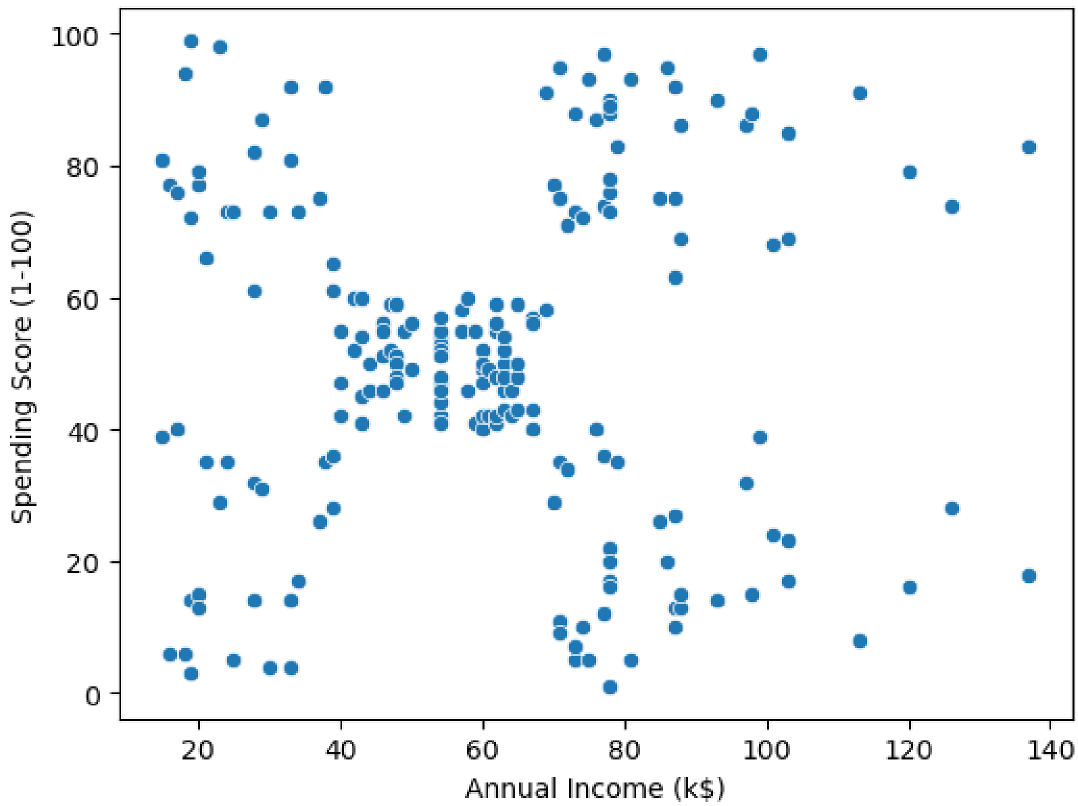
Here we can see the percentage, we have 56% of our data is female and 44% is male.

4 Bivariate Analysis

With a bivariate analysis we're looking for two variables, we are going to star with one plot, bivariate analysis usually is a scatter plot, and It's very helpful to look at.

[12]:
```
sns.scatterplot(data=df, x="Annual Income (k$)", y='Spending Score (1-100)')
```

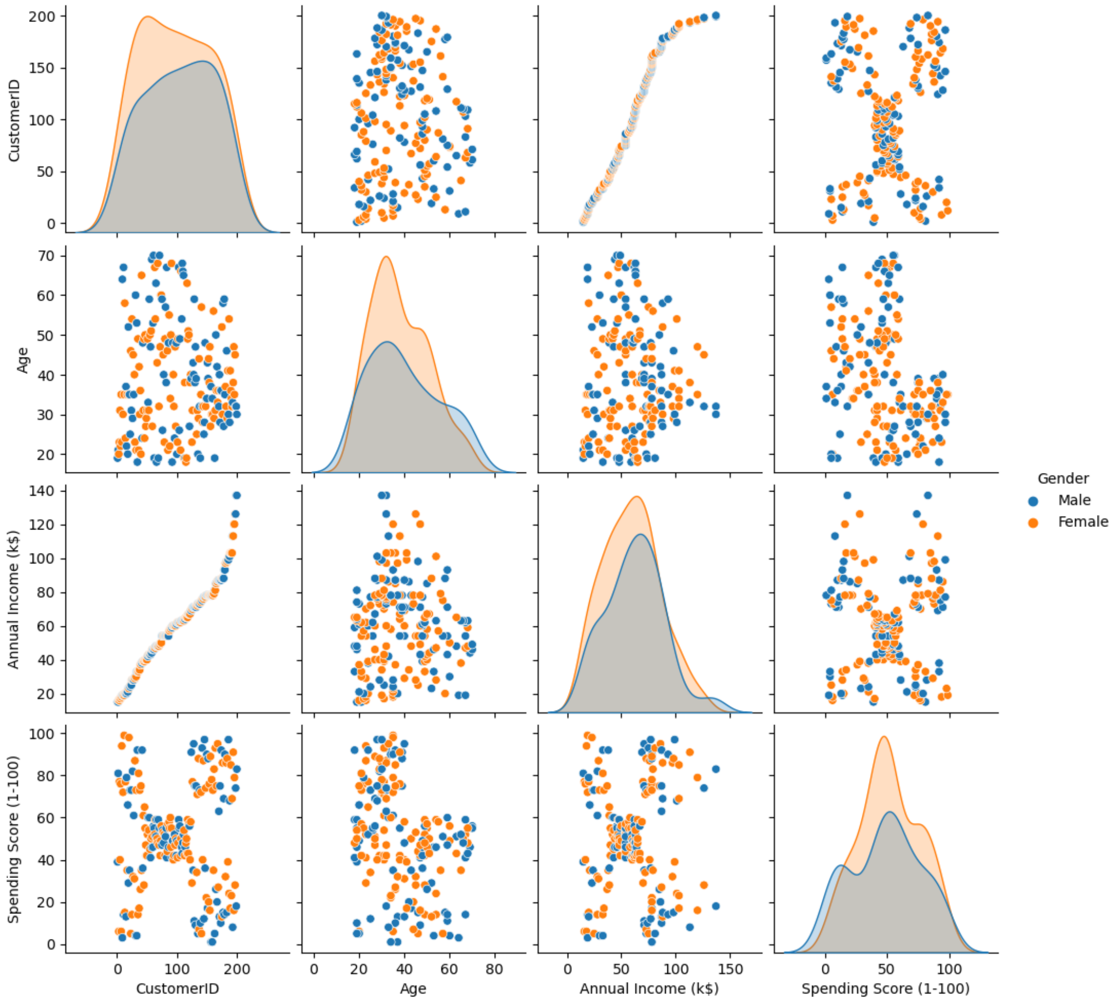
[12]:
```
<Axes: xlabel='Annual Income (k$)', ylabel='Spending Score (1-100)'>
```



We can see some cluster between these two variables, from here we can make some loops, but, in one way we can generate a lot of more importation information with a pair plot for our analysis

[13]:
```
sns.pairplot(df, hue="Gender")
```

[13]: <seaborn.axisgrid.PairGrid at 0x28a8cc60310>



Another thing that we need to see, is the mean values for our data, using "groupby", and grouping it by gender, we will also obtain the correlation between these two.

[14]:
```python
df.groupby(["Gender"])["Age", "Annual Income (k$)",
        "Spending Score (1-100)"].mean()
```

[14]:
```
              Age  Annual Income (k$)  Spending Score (1-100)
Gender
Female  38.098214           59.250000               51.526786
Male    39.806818           62.227273               48.511364
```

[15]:
```python
df.corr() # <- Correlation function
```

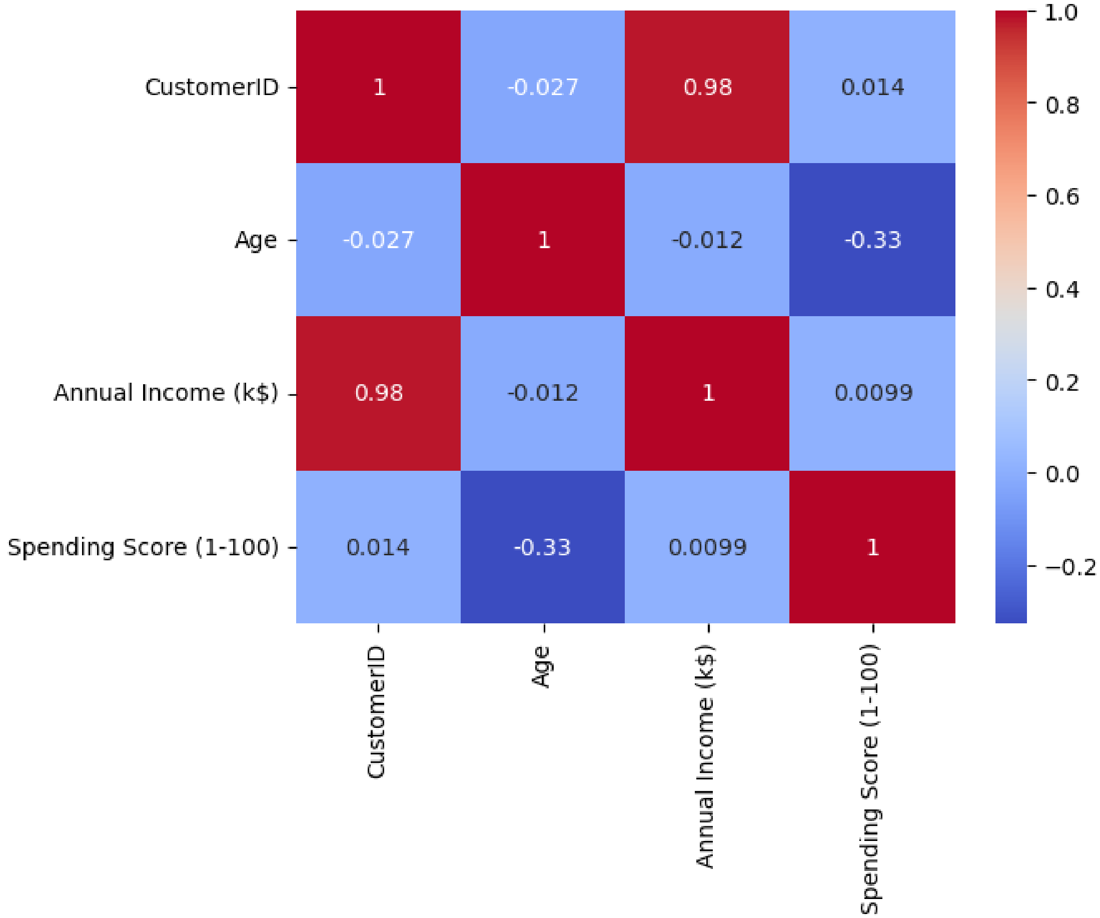
[15]:
```
                        CustomerID       Age  Annual Income (k$)  \
CustomerID                1.000000 -0.026763            0.977548
Age                      -0.026763  1.000000           -0.012398
```

```
Annual Income (k$)        0.977548 -0.012398            1.000000
Spending Score (1-100)    0.013835 -0.327227            0.009903

                        Spending Score (1-100)
CustomerID                            0.013835
Age                                  -0.327227
Annual Income (k$)                    0.009903
Spending Score (1-100)                1.000000
```

[16]:
```
sns.heatmap(df.corr(), annot=True, cmap="coolwarm")
```

[16]: <Axes: >



5 Clustering - Univariate, Bivariate, Multivariate

The first thing we going to do, is get the K-Means agorithm; we calculate the clustering algorithm for several values of k. This can be done by creating a variation within k from 1 to 10 clusters. We then calculate the total intra-cluster sum of square (iss). Then, we proceed to plot is based on the number of k clusters. This plot denotes the appropriate number of clusters required in our model. In the plot, the location of a bend or a knee is the indication of the optimum number of clusters.

[17]:
```
clustering1 = KMeans(n_clusters=3)
```

[18]:
```
clustering1.fit(df[["Annual Income (k$)"]])
```

[18]:
```
KMeans(n_clusters=3)
```

[19]:
```
clustering1.labels_
```

[19]:
```
array([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, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2])
```

[20]:
```
df["Income Cluster"] = clustering1.labels_
df.head()
```

[20]:
```
   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

   Income Cluster
0               0
1               0
2               0
3               0
4               0
```

[21]:
```
df["Income Cluster"].value_counts()
```

[21]:
```
1    92
0    72
2    36
Name: Income Cluster, dtype: int64
```

[22]:
```
clustering1.inertia_ #Inertia represents is the distance between centroids
```

[22]: 23528.152173913048

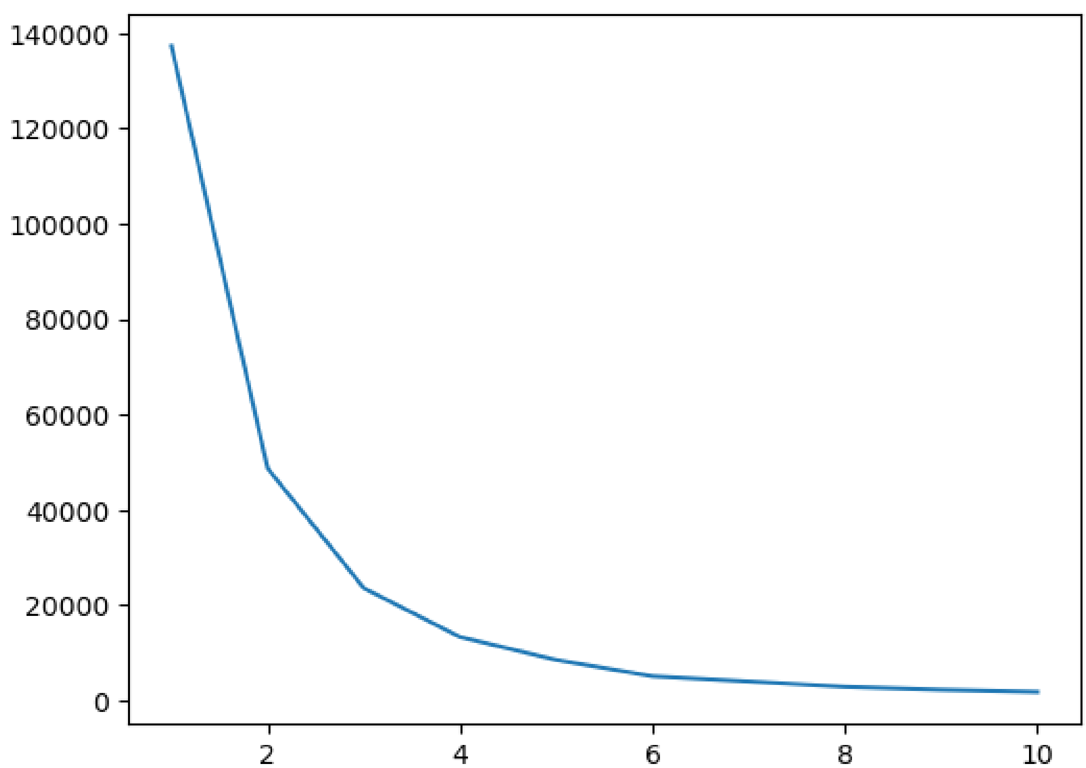
[23]:
```
intertia_scores=[]
for i in range(1,11):
    kmeans = KMeans(n_clusters=i)
    kmeans.fit(df[["Annual Income (k$)"]])
    intertia_scores.append(kmeans.inertia_)
```

[24]:
```
intertia_scores
```

[24]:
```
[137277.28000000003,
 48660.88888888888,
 23528.152173913048,
 13278.112713472487,
 8481.496190476191,
 5050.9047619047615,
 3955.2566544566553,
 2827.308424908425,
 2208.812049062049,
 1774.5010822510822]
```

[25]:
```
plt.plot(range(1,11),intertia_scores)
```

[25]: [<matplotlib.lines.Line2D at 0x28a8ca2fdf0>]



[26]:
```
df.columns
```

[26]:
```
Index(['CustomerID', 'Gender', 'Age', 'Annual Income (k$)',
       'Spending Score (1-100)', 'Income Cluster'],
      dtype='object')
```

[27]:
```
df.groupby("Income Cluster")['Age', 'Annual Income (k$)', 'Spending Score␣
 ↪(1-100)'].mean()
```

[27]:

Income Cluster	Age	Annual Income (k$)	Spending Score (1-100)
0	38.930556	33.027778	50.166667
1	39.184783	66.717391	50.054348
2	37.833333	99.888889	50.638889

6 Bivariate Clustering

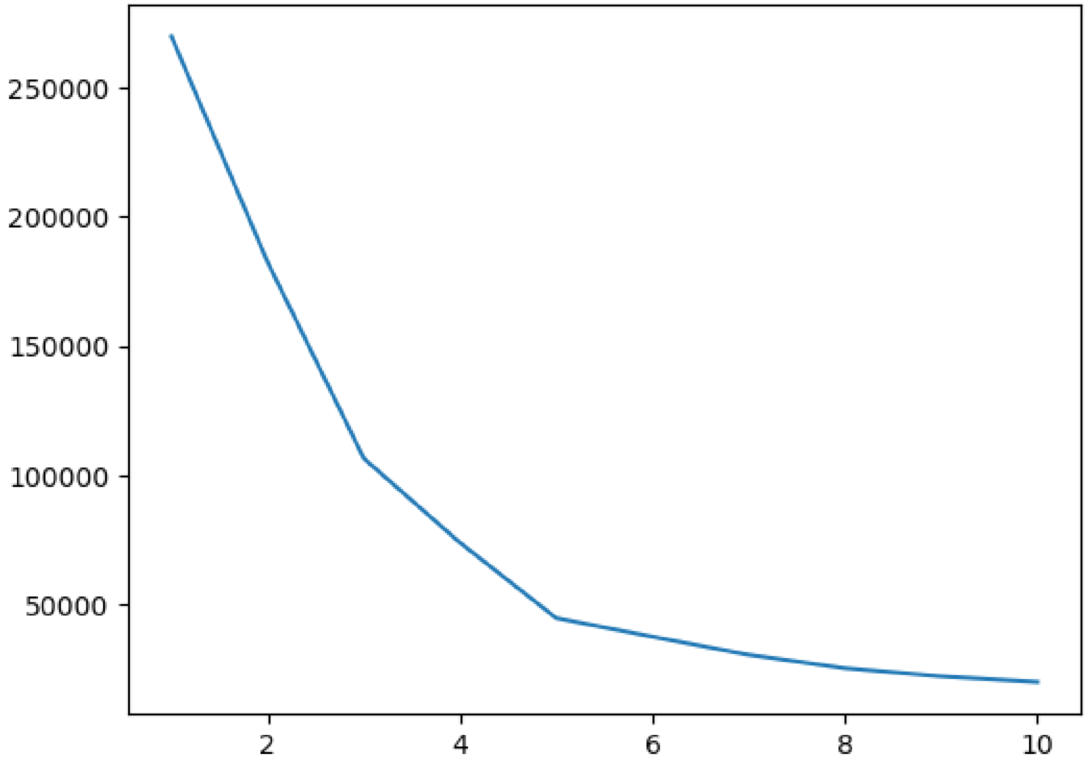
[28]:
```
clustering2 = KMeans(n_clusters=5)
clustering2.fit(df[['Annual Income (k$)','Spending Score (1-100)']])
df["Spending and Income Cluster"] = clustering2.labels_
df.head()
```

[28]:

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

[29]:
```
intertia_scores2=[]
for i in range(1,11):
    kmeans2 = KMeans(n_clusters=i)
    kmeans2.fit(df[["Annual Income (k$)","Spending Score (1-100)"]])
    intertia_scores2.append(kmeans2.inertia_)
plt.plot(range(1,11),intertia_scores2)
```

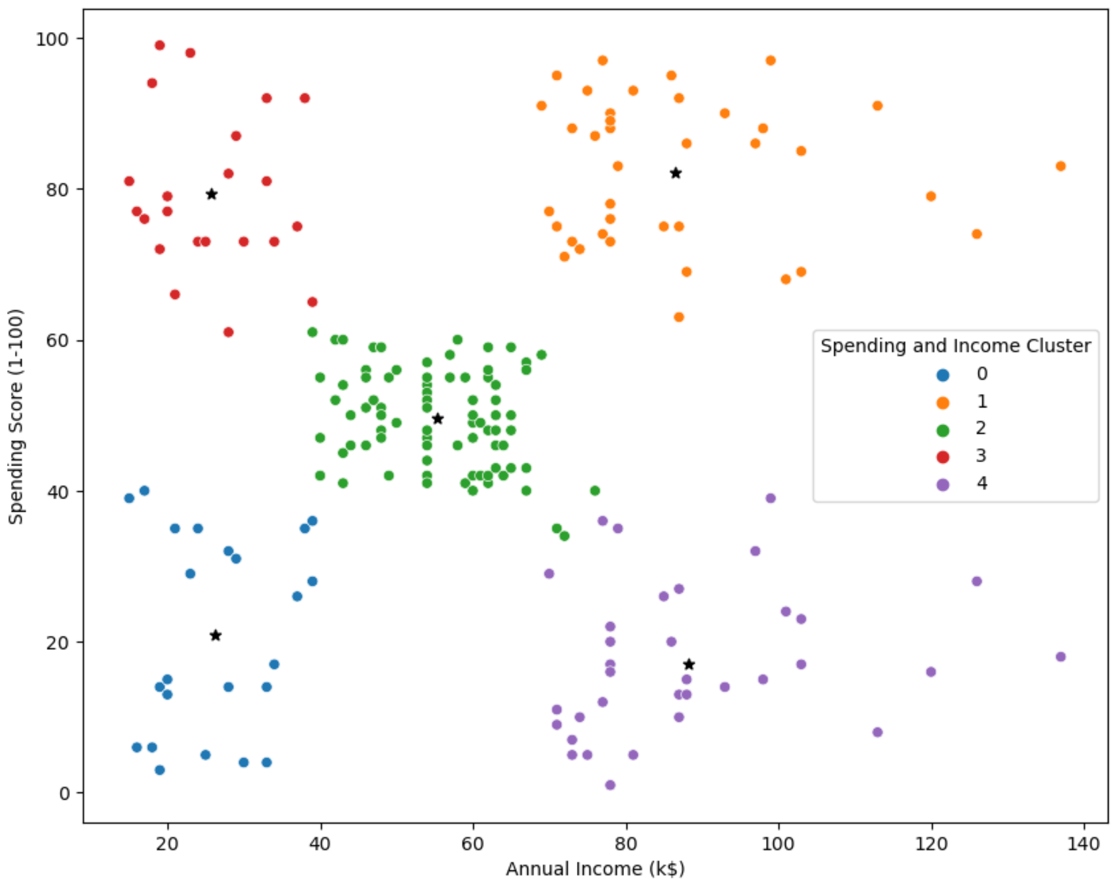
[29]:
```
[<matplotlib.lines.Line2D at 0x28a8cae0be0>]
```



- From the above graph, we conclude that 5 is the appropriate number of clusters since it seems to be appearing at the bend in the elbow plot.

[30]:
```
centers = pd.DataFrame(clustering2.cluster_centers_)
centers.columns = ["x","y"]
```

[31]:
```
plt.figure(figsize = (10,8))
plt.scatter(x=centers["x"], y=centers["y"],c="black",marker="*")
sns.scatterplot(data=df, x = "Annual Income (k$)", y = "Spending Score␣
↪(1-100)", hue = "Spending and Income Cluster", palette = "tab10")
plt.savefig("Clustering_bivariate_.png")
```



[32]:
```
pd.crosstab(df["Spending and Income Cluster"],df["Gender"], normalize="index")
```

[32]:
```
Gender                        Female      Male
Spending and Income Cluster
0                           0.608696  0.391304
1                           0.538462  0.461538
2                           0.592593  0.407407
3                           0.590909  0.409091
4                           0.457143  0.542857
```

[33]:
```
df.groupby("Spending and Income Cluster")["Age","Annual Income (k$)",
       "Spending Score (1-100)"].mean()
```

[33]:
```
                                   Age  Annual Income (k$)  \
Spending and Income Cluster
0                            45.217391           26.304348
1                            32.692308           86.538462
2                            42.716049           55.296296
3                            25.272727           25.727273
```

```
4                            41.114286          88.200000

                             Spending Score (1-100)
Spending and Income Cluster
0                                         20.913043
1                                         82.128205
2                                         49.518519
3                                         79.363636
4                                         17.114286
```

7 Analysis

- Target group would be cluster 1 which has a high spending score and high income.
- 54 percent of cluster 1 are women, a market strategy should be generated for this group, targeting popular items in this cluster.
- Cluster 3 presents an interesting business opportunity for the company, targeting this sector by selling popular items.

[ ]:

```

```