```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
from sklearn.cluster import KMeans
import warnings
warnings.filterwarnings('ignore')
```

```
df = pd.read_csv('/content/Mall_Customers.csv')
```

```
df.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

Next steps: Generate code with df   View recommended plots   New interactive sheet

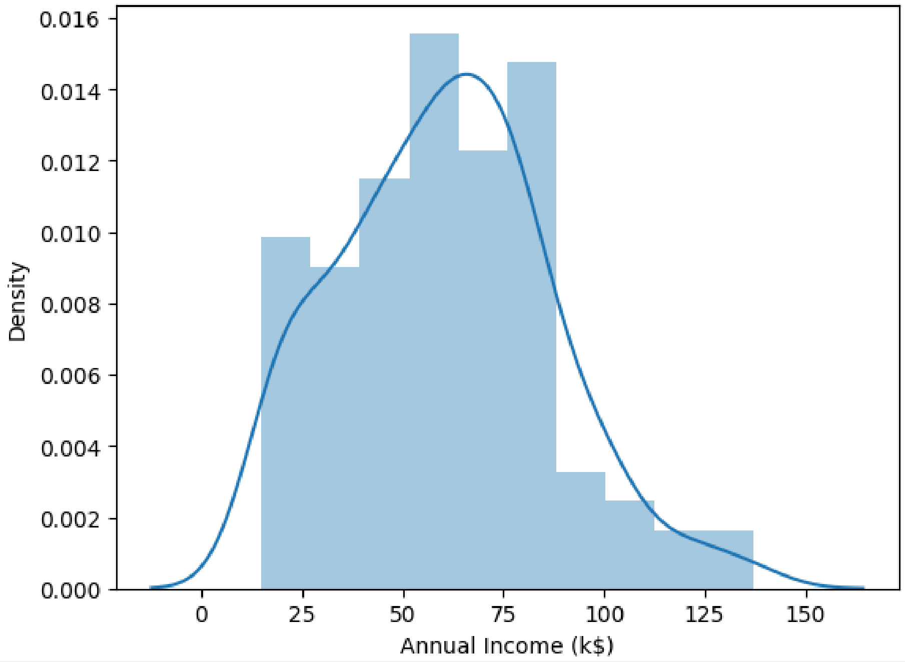
Univariate Analysis

```
df.describe()
```

	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

```
sns.distplot(df['Annual Income (k$)'])
```

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

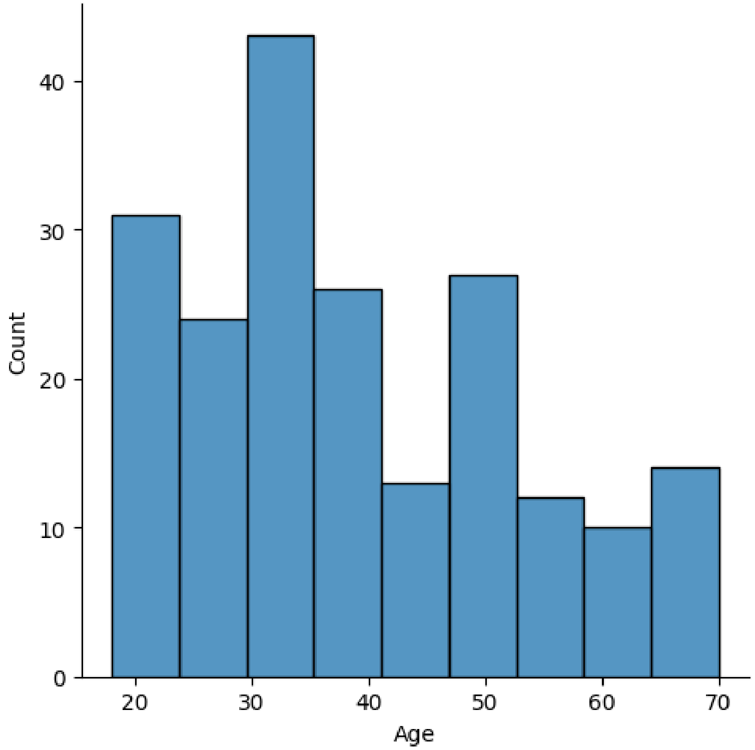


```
df.columns
```

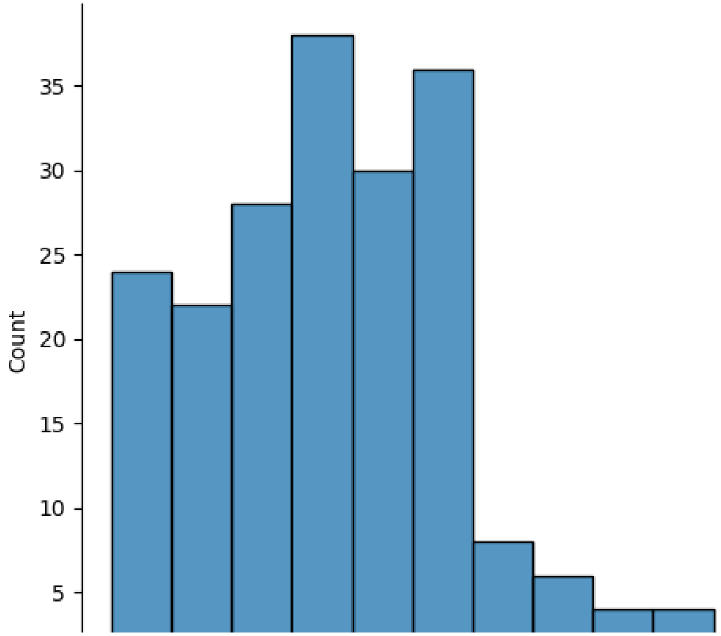
```
Index(['CustomerID', 'Gender', 'Age', 'Annual Income (k$)',
       'Spending Score (1-100)'],
      dtype='object')
```

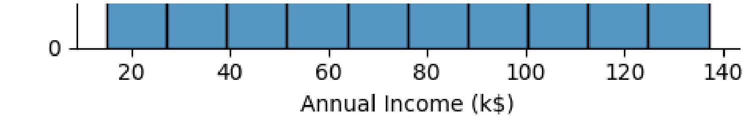
```
columns = ['Age','Annual Income (k$)','Spending Score (1-100)']
for i in columns:
  plt.figure()
  sns.displot(df[i])
```

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

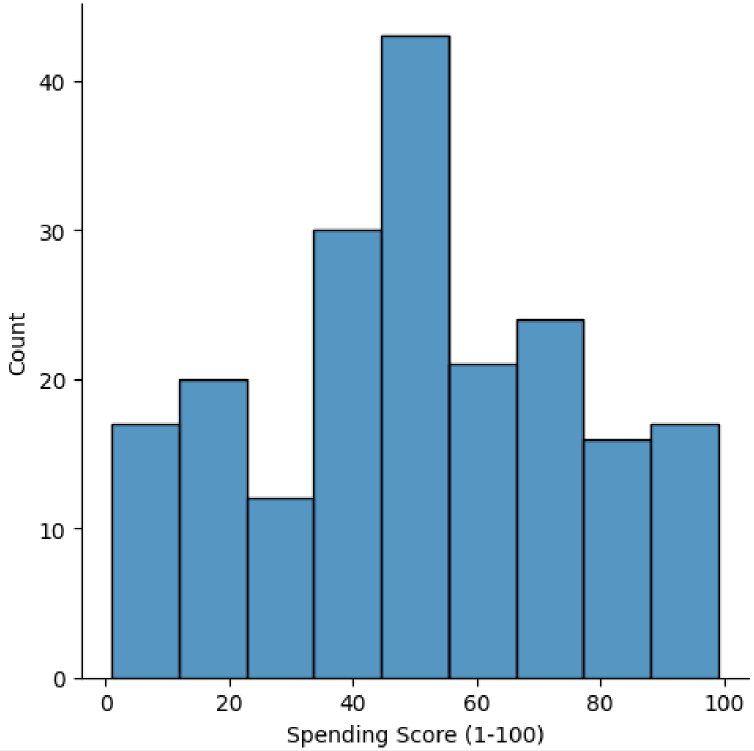

Count
0
10
20
30
40
20
30
40
50
60
70
Age

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


Count
5
10
15
20
25
30
35

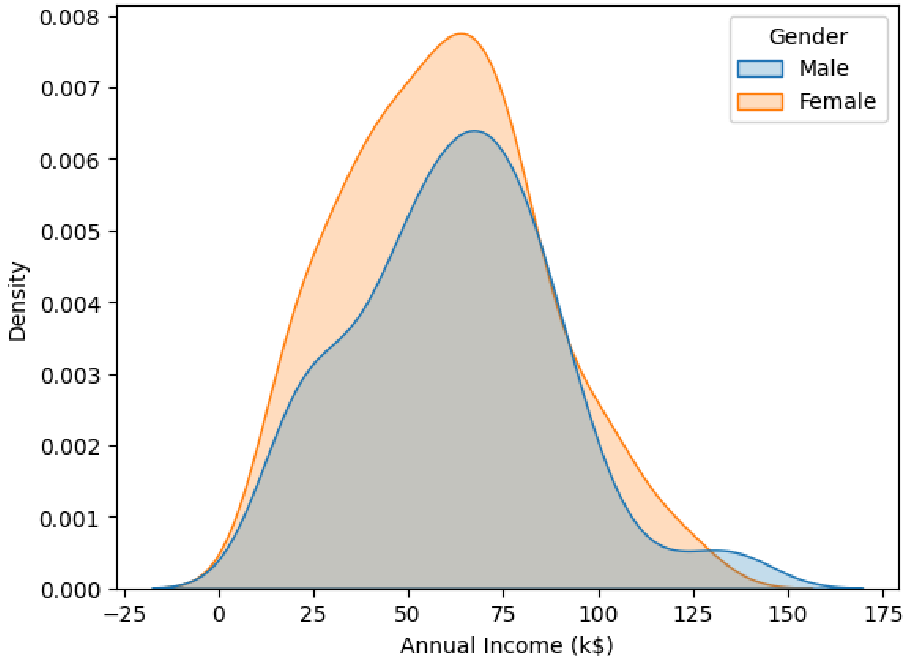

0
20
40
60
80
100
120
140
Annual Income (k$)

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


Count
0
10
20
30
40
0
20
40
60
80
100
Spending Score (1-100)

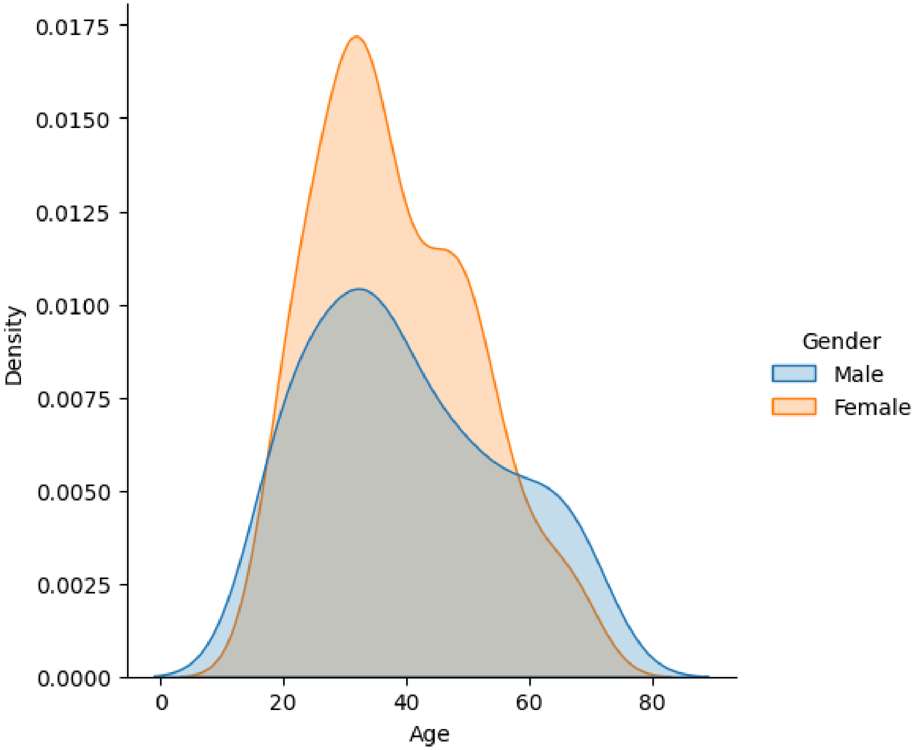
```
sns.kdeplot(x='Annual Income (k$)', hue='Gender', data=df, shade=True)
```

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

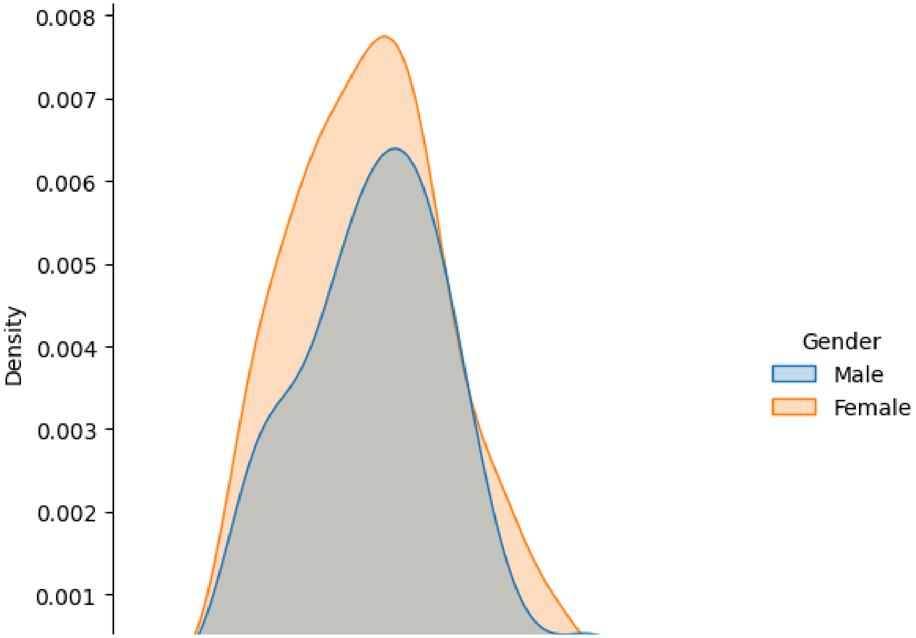


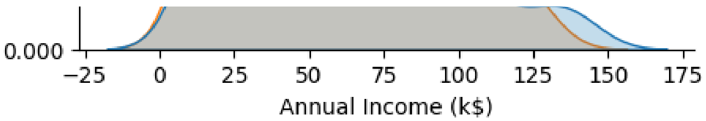
```
columns = ['Age','Annual Income (k$)','Spending Score (1-100)']
for i in columns:
  plt.figure()
  # Use displot for automatic hue handling
  sns.displot(data=df, x=i, hue='Gender', kind='kde', fill=True)
  # Alternatively, specify x and hue for kdeplot
  # sns.kdeplot(data=df, x=i, hue='Gender', shade=True)
```

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

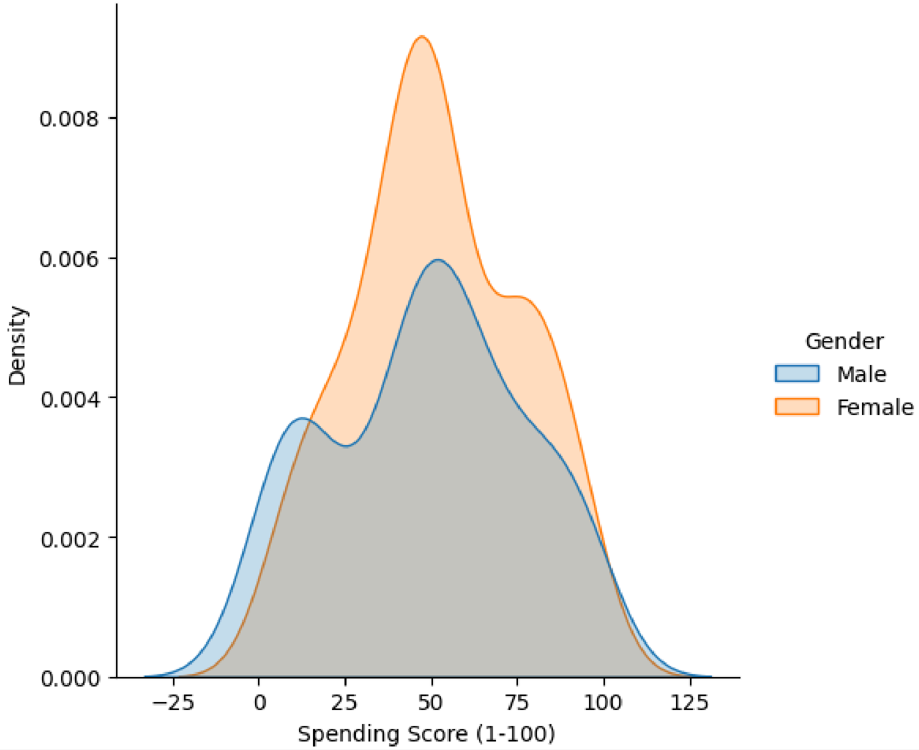

0.0175
0.0150
0.0125
0.0100
0.0075
0.0050
0.0025
0.0000
Density
0
20
40
60
80
Age
Gender
Male
Female

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

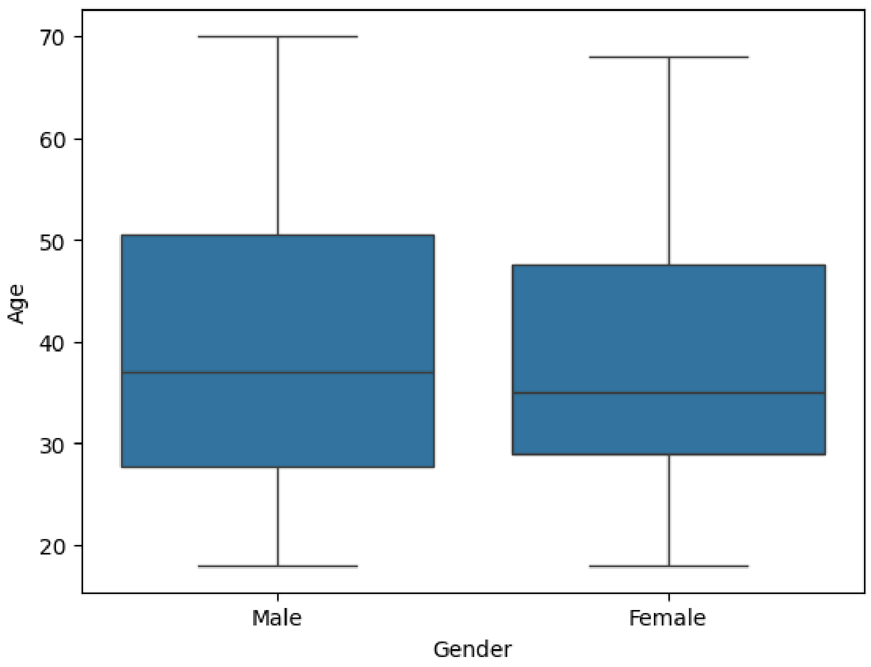

0.008
0.007
0.006
0.005
0.004
0.003
0.002
0.001
Density
Gender
Male
Female

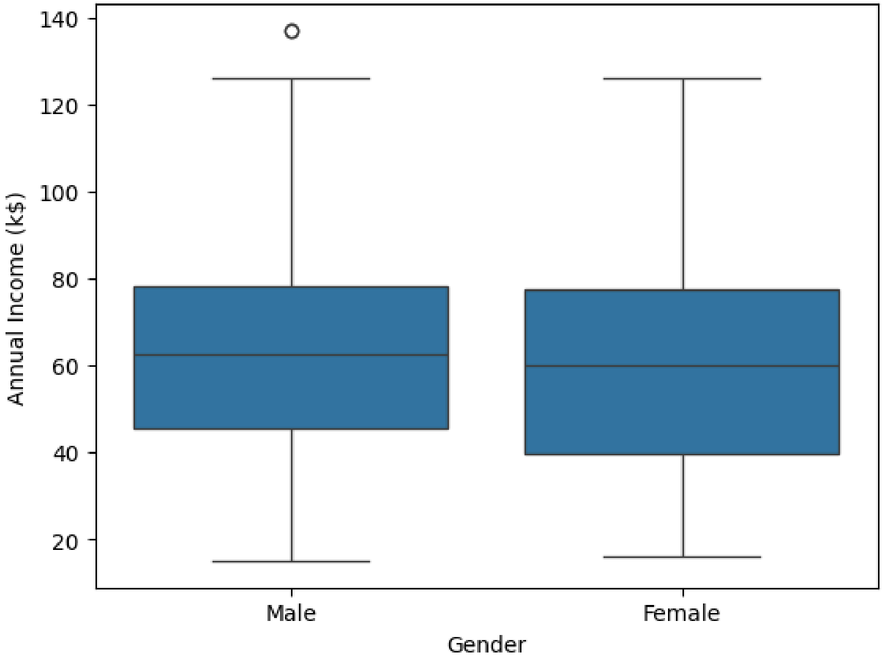

0.000
−25
0
25
50
75
100
125
150
175
Annual Income (k$)

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

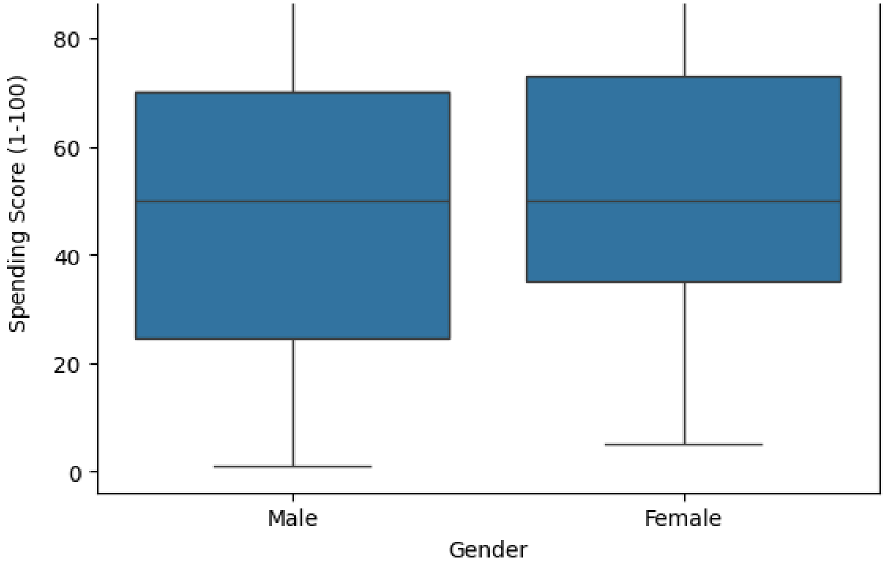

Density
0.008
0.006
0.004
0.002
0.000
−25
0
25
50
75
100
125
Spending Score (1-100)
Gender
Male
Female

```
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

100



```
df['Gender'].value_counts(normalize=True)
```

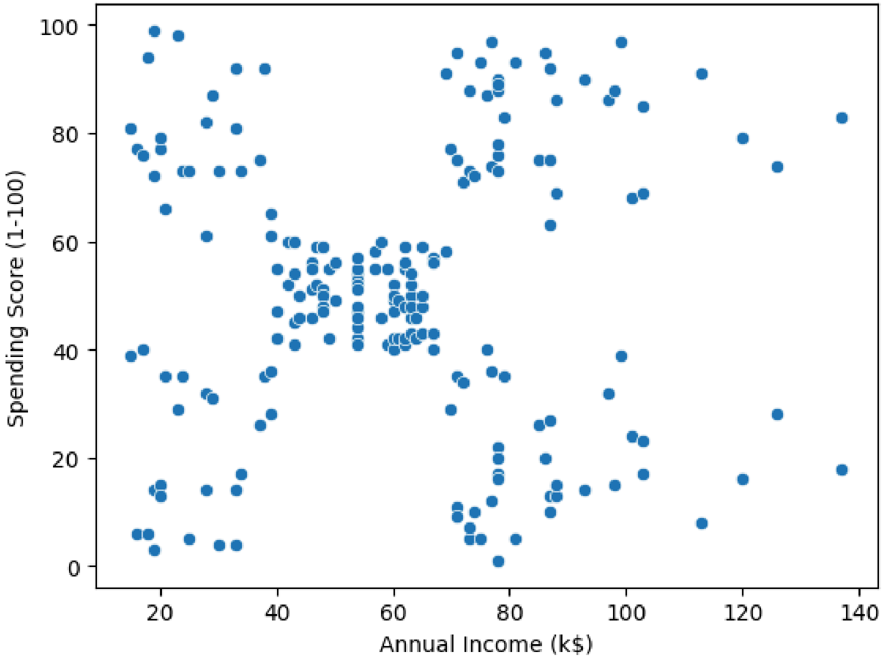
```
         proportion
Gender
Female         0.56
Male           0.44

dtype: float64
```

Bivariate Analysis

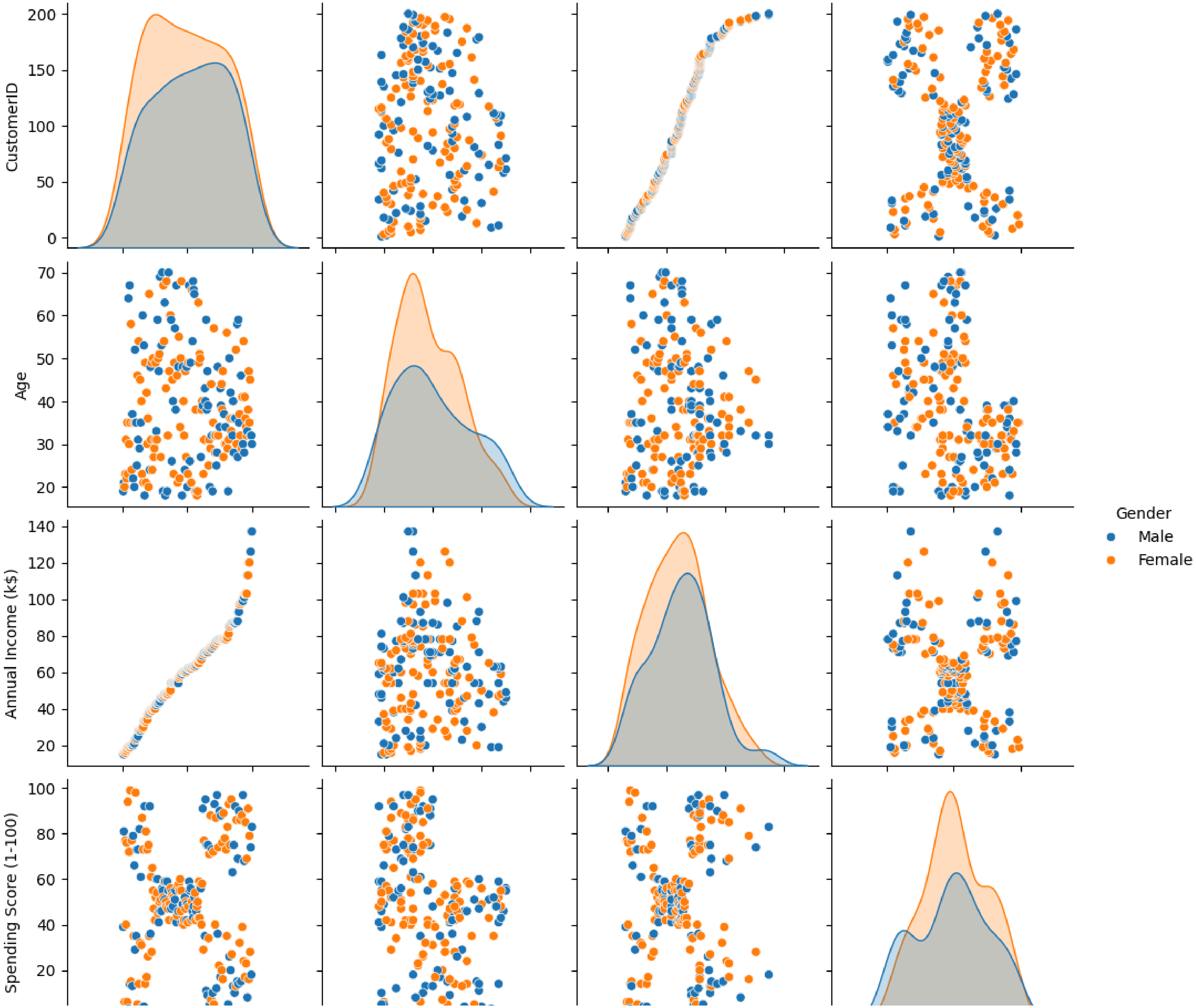
```
sns.scatterplot(data=df,x='Annual Income (k$)',y='Spending Score (1-100)')
```

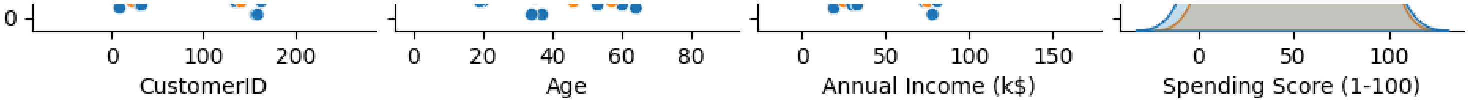
```
<Axes: xlabel='Annual Income (k$)', ylabel='Spending Score (1-100)'>
```



```
sns.pairplot(df,hue='Gender')
```

```
<seaborn.axisgrid.PairGrid at 0x7d4ce2f5bb20>
```


CustomerID
200
150
100
50
0
Age
70
60
50
40
30
20
Annual Income (k$)
140
120
100
80
60
40
20
Spending Score (1-100)
100
80
60
40
20
Gender
● Male
● Female



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

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

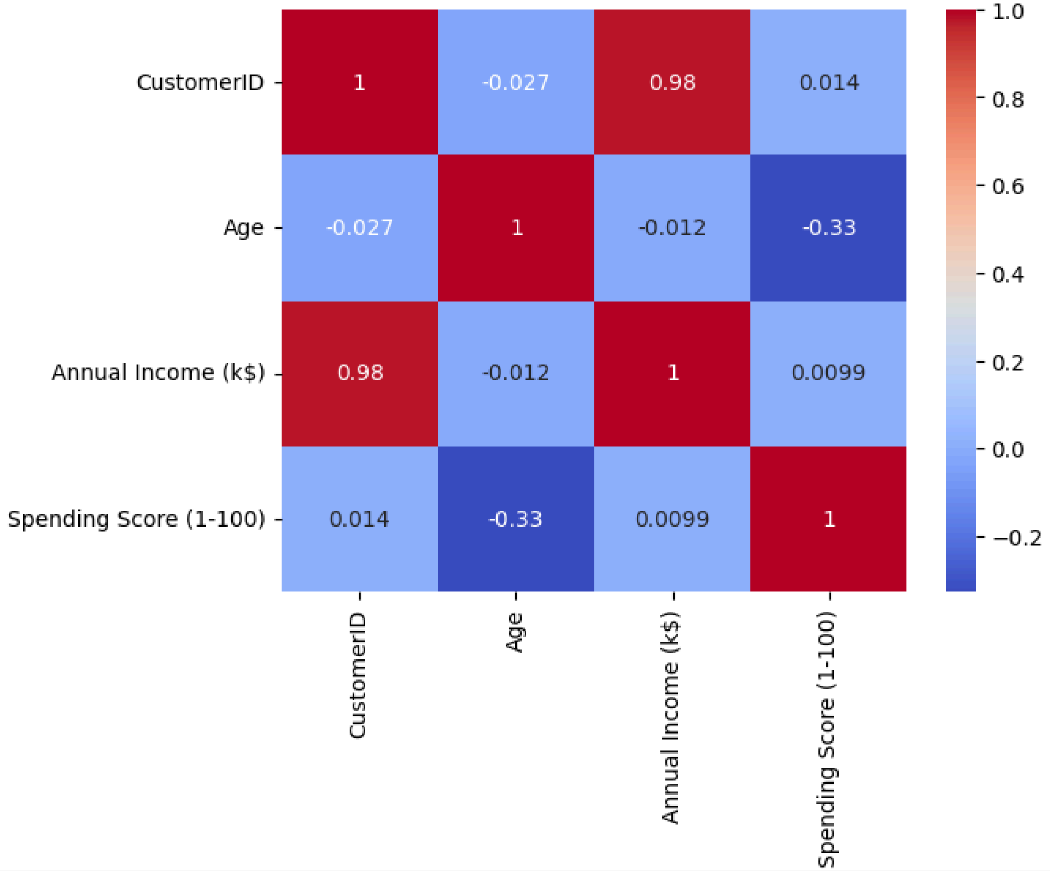
```
# Assuming 'df' is your DataFrame

# Select only the numeric columns for correlation calculation
numeric_df = df.select_dtypes(include=np.number)

# Calculate the correlation matrix
corr_matrix = numeric_df.corr()

# Generate the heatmap
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm')
```

```
<Axes: >
```



Clustering - Univariate , Bivariate , Multivariate

```
clustering1 = KMeans
```

```


clustering1 = KMeans(n_clusters=6)  # For example, 5 clusters

# Reshape the 'Annual Income (k$)' column into a 2D array
X = df[['Annual Income (k$)']]  # Use double brackets to create a DataFrame
# or
# X = df['Annual Income (k$)'].values.reshape(-1, 1)  # Reshape using NumPy

# Fit the KMeans model to the data
clustering1.fit(X)
```

```
▾ KMeans ⓘ ?
KMeans(n_clusters=6)
```

```
clustering1.labels_
```

```
array([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, 4, 4, 4, 4, 4, 4, 4, 4,
       4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
       4, 4, 4, 4, 4, 4, 4, 4, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
       3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
       3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 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, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 5, 5, 5, 5,
       5, 5], dtype=int32)
```

```
df['cluster'] = clustering1.labels_
```

```
df.head()
```

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

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
df['cluster'].value_counts()
```

	count
cluster	
0	54
3	50
4	38
1	36
2	16
5	6

dtype: int64

```
clustering1.inertia_
```

```
5496.533937621832
```

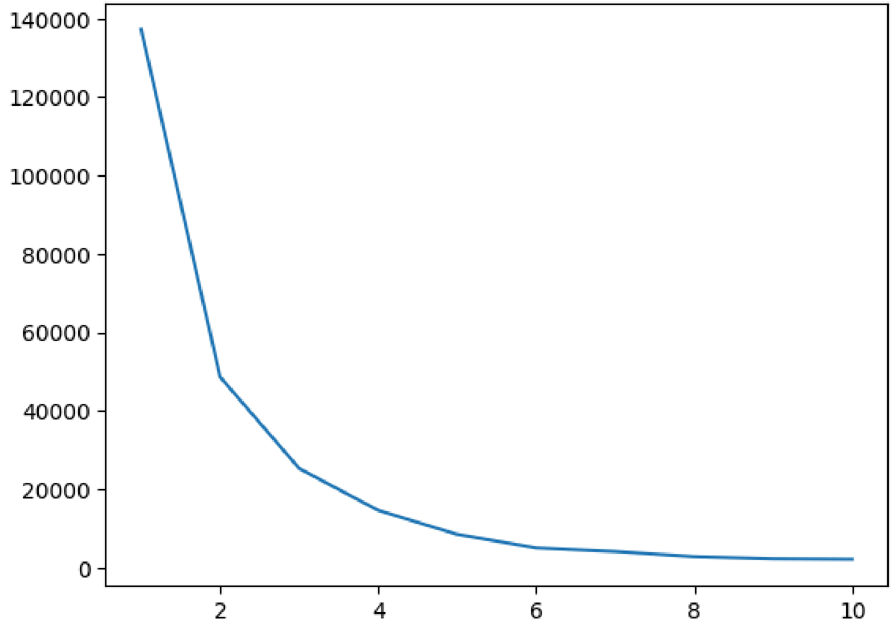
```
intrtia_scores = []
for i in range(1,11):
  clustering1 = KMeans(n_clusters=i)
  clustering1.fit(X)
  intrtia_scores.append(clustering1.inertia_)
```

```
intrtia_scores
```

```
[137277.2800000002,
 48660.88888888887,
 25341.285871863212,
 14656.333089668611,
 8534.41515455305,
 5081.484660267269,
 4151.620028011211,
 2836.339987789987,
 2296.2830808080807,
 2177.154004329006]
```

```
plt.plot(range(1,11),intrtia_scores)
```

```
[<matplotlib.lines.Line2D at 0x7d4cdb4d2890>]
```



```
df.groupby(['cluster'])[['Age','Annual Income (k$)','Spending Score (1-100)']].mean()
```

cluster	Age	Annual Income (k$)	Spending Score (1-100)
0	36.018519	78.370370	49.555556
1	34.944444	23.222222	49.444444
2	37.812500	100.875000	52.875000
3	41.520000	60.440000	50.060000
4	43.815789	43.210526	50.973684
5	36.833333	127.666667	49.666667

Bivariate Clustering

```
clustering2 = KMeans()
clustering2.fit(df[['Annual Income (k$)','Spending Score (1-100)']])
clustering2.labels_
```

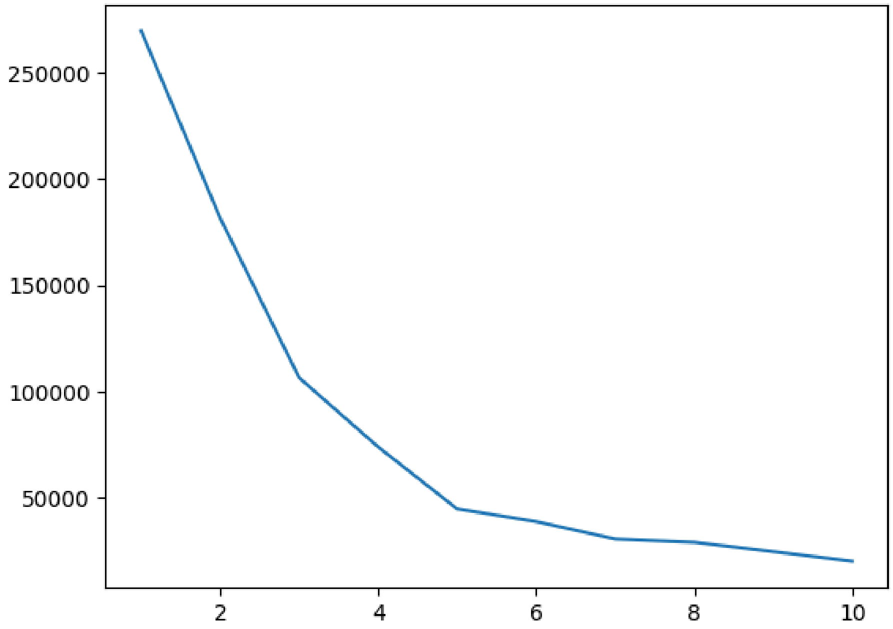
```
df['Spending_cluster'] = clustering2.labels_
df.head()
```

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

Next steps: Generate code with df View recommended plots New interactive sheet

```
intertia_scores2 =[]
for i in range(1,11):
  clustering2 = KMeans(n_clusters=i)
  clustering2.fit(df[['Annual Income (k$)','Spending Score (1-100)']])
  # Append the inertia score to the list
  intertia_scores2.append(clustering2.inertia_)
plt.plot(range(1,11),intertia_scores2)
```

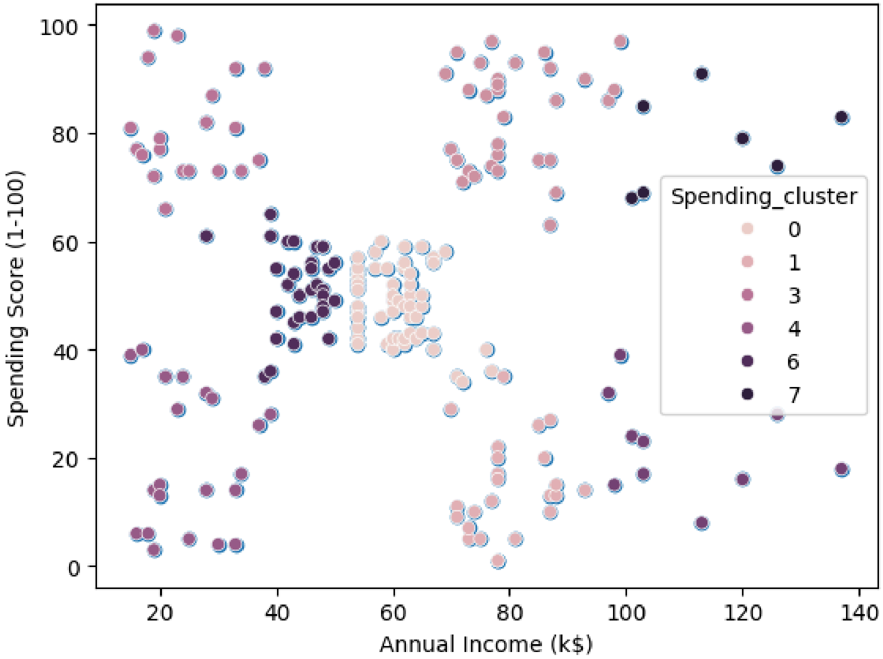
```
[<matplotlib.lines.Line2D at 0x7d4ce2f58580>]
```



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

```
# Corrected function name: 'scatter' instead of 'scatterplt'
plt.scatter(df['Annual Income (k$)'], df['Spending Score (1-100)'])
sns.scatterplot(data=df, x='Annual Income (k$)', y='Spending Score (1-100)', hue='Spending_cluster')
```

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



```
# Create a new column 'Spending and Income Cluster' based on 'Spending_cluster'
# Assuming 'Spending_cluster' is a relevant column for your analysis
df['Spending and Income Cluster'] = df['Spending_cluster']

# Now, you can create the crosstab
pd.crosstab(df['Spending and Income Cluster'], df['Gender'], normalize='index')
```

```
Gender                      Female      Male
Spending and Income Cluster
          0               0.566038  0.433962
          1               0.375000  0.625000
          2               0.531250  0.468750
```

```
df.groupby('Spending and Income Cluster')[['Age', 'Annual Income (k$)', 'Spending Score (1-100)']].mean()
# Changed the tuple of column names to a list by enclosing them in square brackets []
```

```
                                   Age  Annual Income (k$)  Spending Score (1-100)
Spending and Income Cluster
          0                  41.150943           61.301887               48.245283
          1                  40.875000           79.708333               14.291667
```