Mall Customer Segmentation Data - Unsupervised ML technique (KMeans Clustering Algorithm)

In [ ]:

```
# Import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

In [ ]:

```
#Load the dataset
file_path = "Mall_Customers.csv"
data = pd.read_csv(file_path)
```

In [ ]:

```
data.head()
```

Out[ ]:

	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 [ ]:

```
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
```

In [ ]:

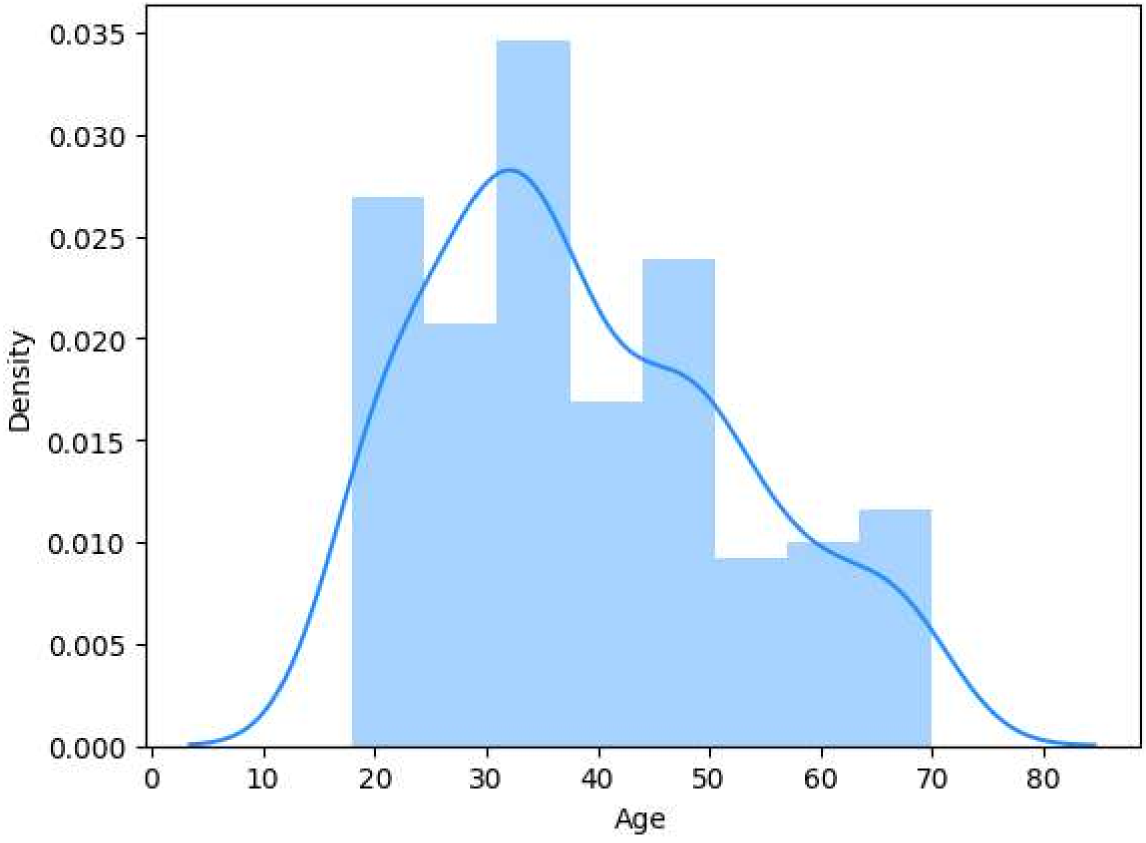
```
data.describe().style.format("{:.2f}")
```

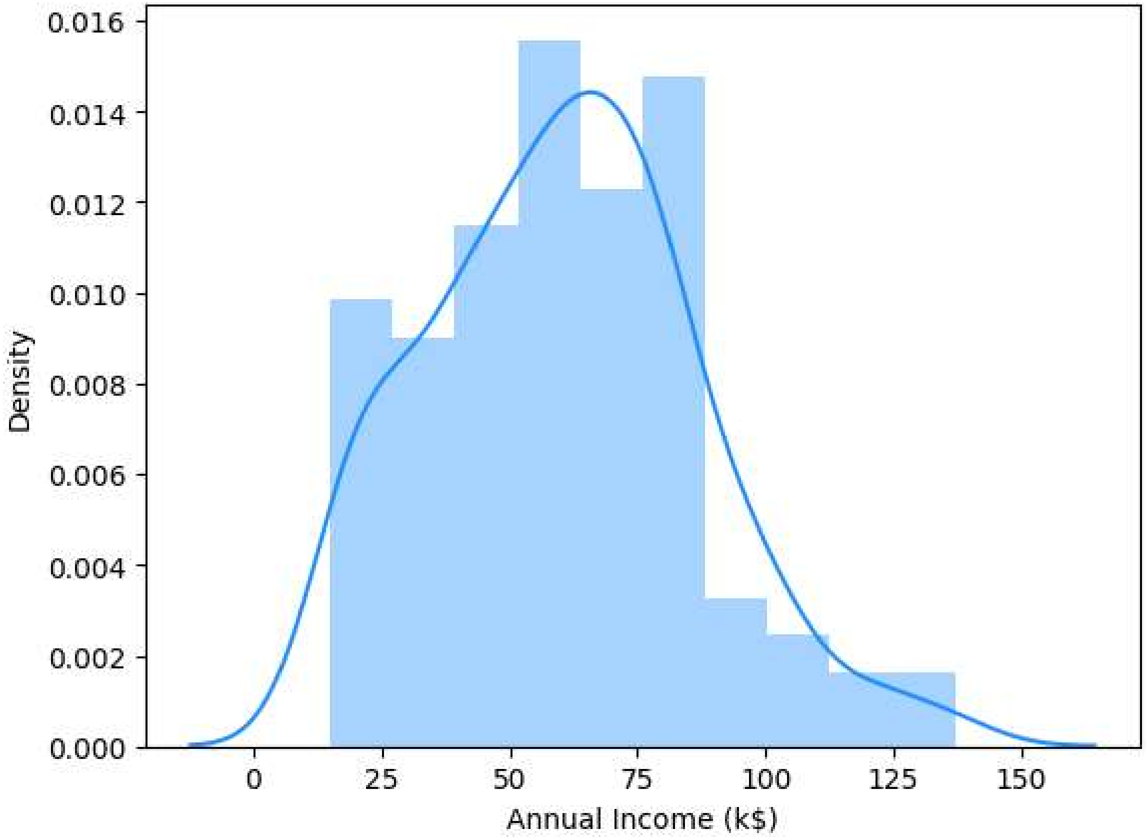
Out[ ]:

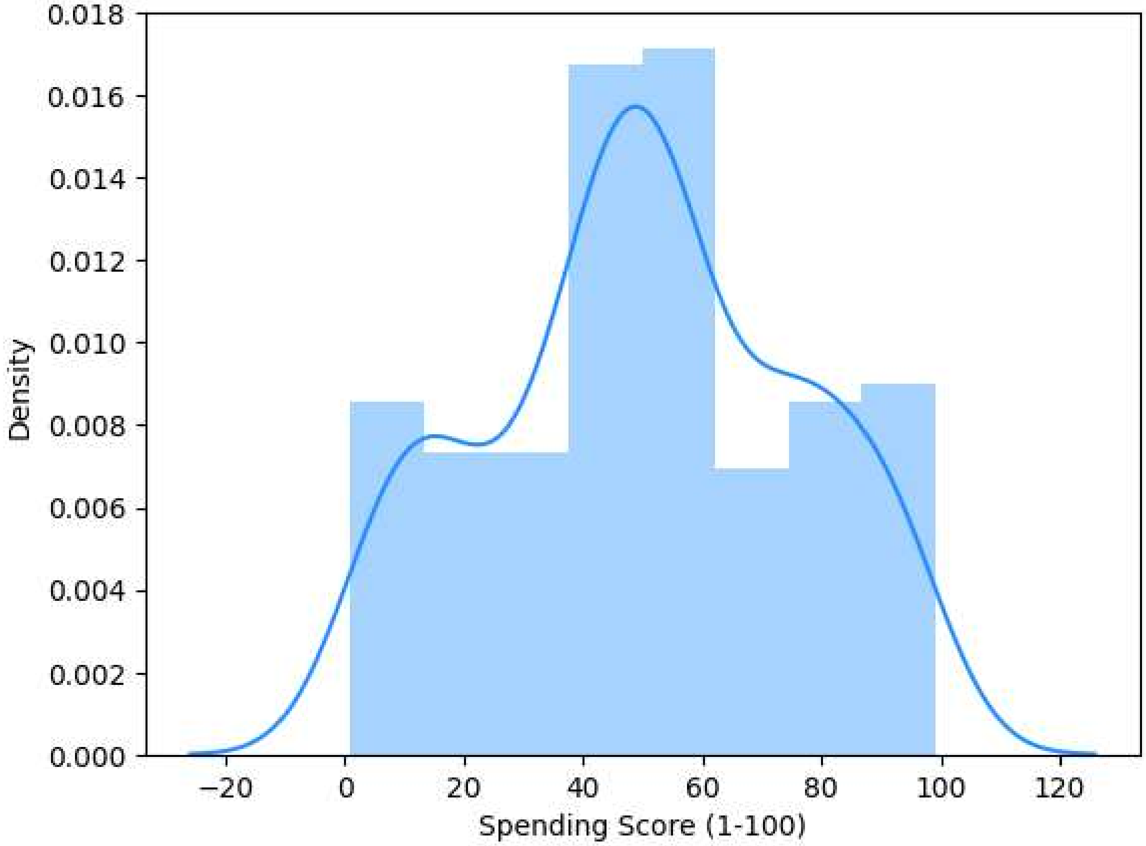
	CustomerID	Age	Annual Income (k$)	Spending Score (1-100)
count	200.00	200.00	200.00	200.00
mean	100.50	38.85	60.56	50.20
std	57.88	13.97	26.26	25.82
min	1.00	18.00	15.00	1.00
25%	50.75	28.75	41.50	34.75
50%	100.50	36.00	61.50	50.00
75%	150.25	49.00	78.00	73.00
max	200.00	70.00	137.00	99.00

In [ ]:

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

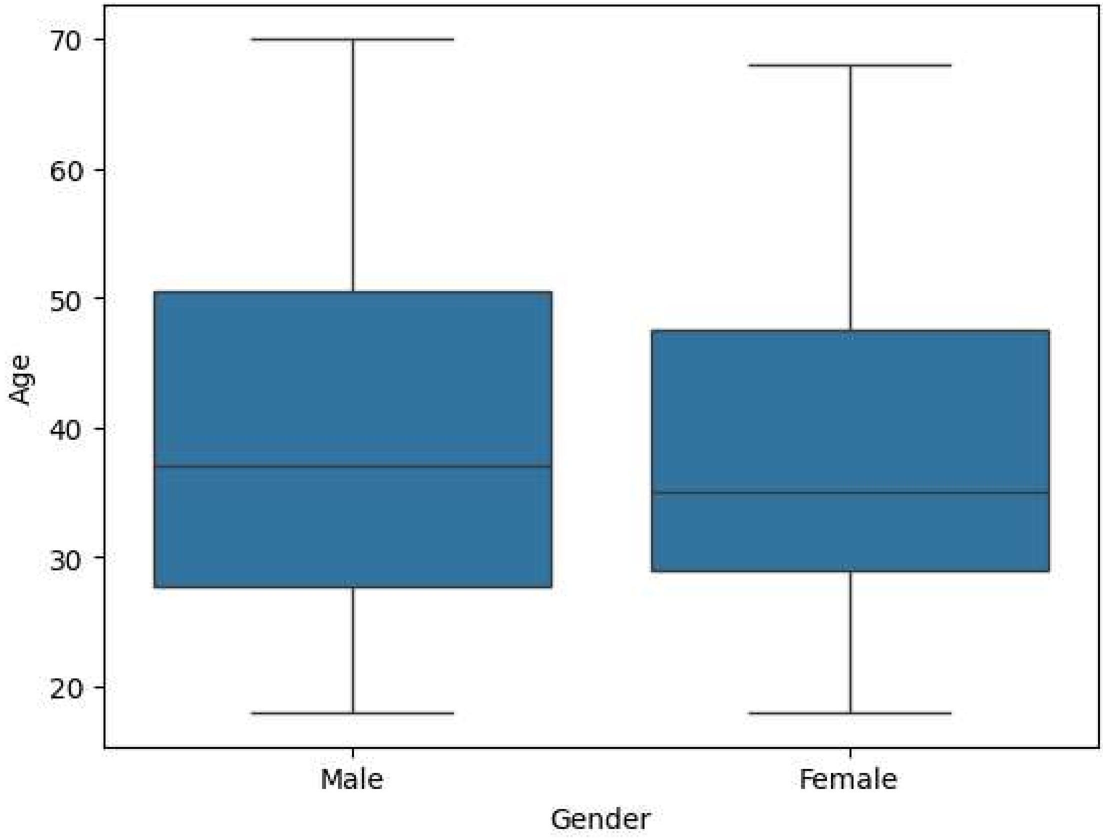


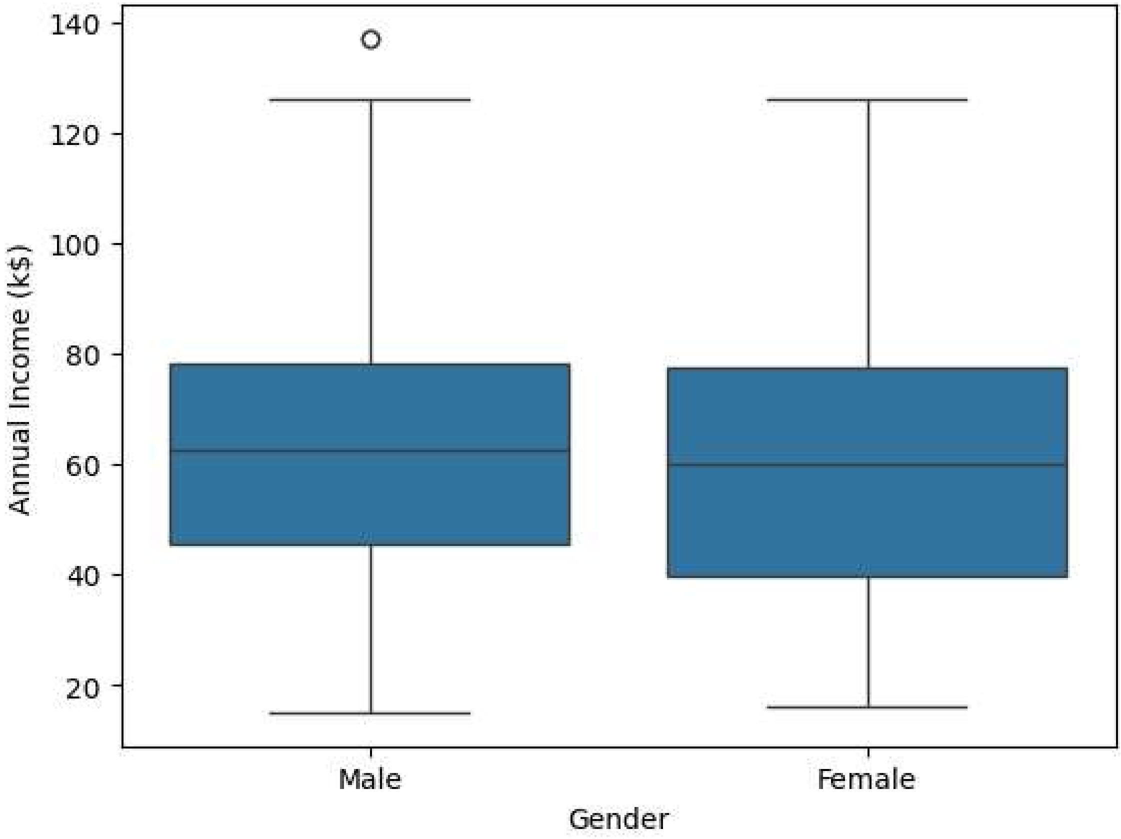


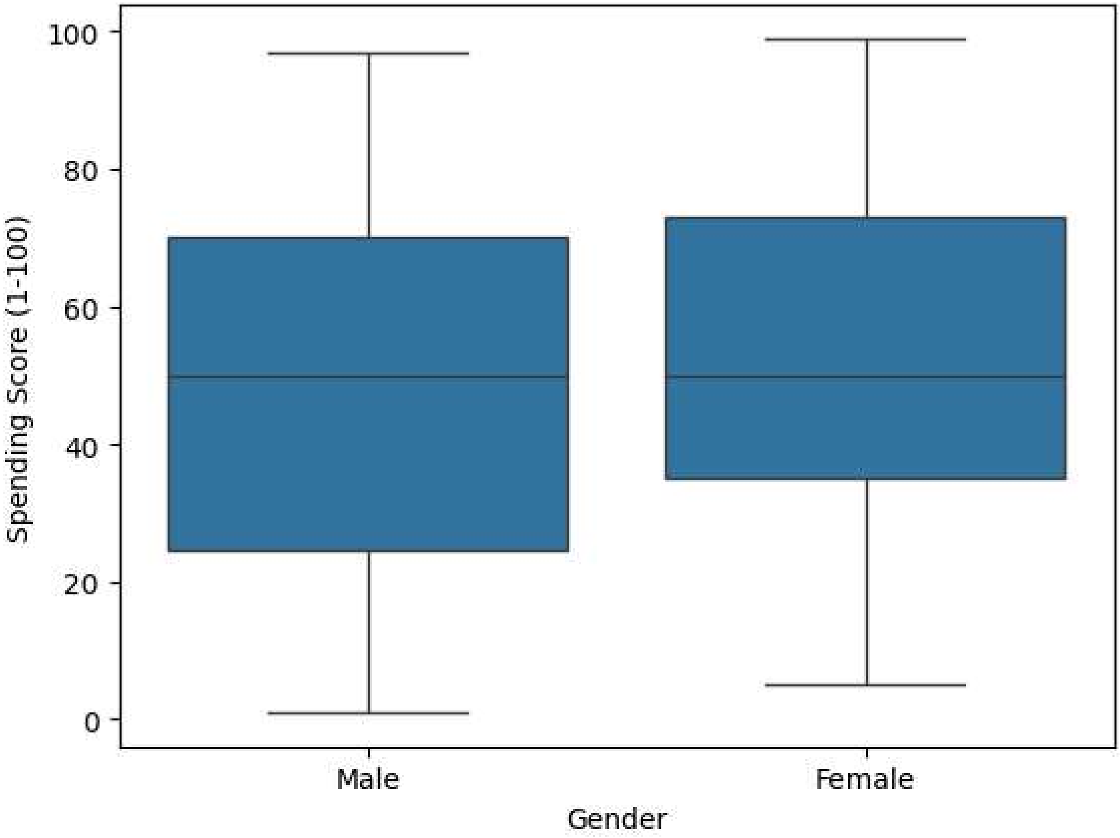


In [ ]:

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


70
60
50
40
30
20
Age
Male
Female
Gender


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



In [ ]:

```
# Data grouped by Gender
data.groupby(['Gender'])[['Age', 'Annual Income (k$)',
       'Spending Score (1-100)']].mean().style.format("{:.2f}").set_properties(*
```

Out[ ]:

Gender	Age	Annual Income (k$)	Spending Score (1-100)
Female	38.10	59.25	51.53
Male	39.81	62.23	48.51

In [ ]:

```
# Correlation
data.corr(numeric_only=True).style.format("{:.2f}").set_properties(**{'text-alig
```

Out[ ]:

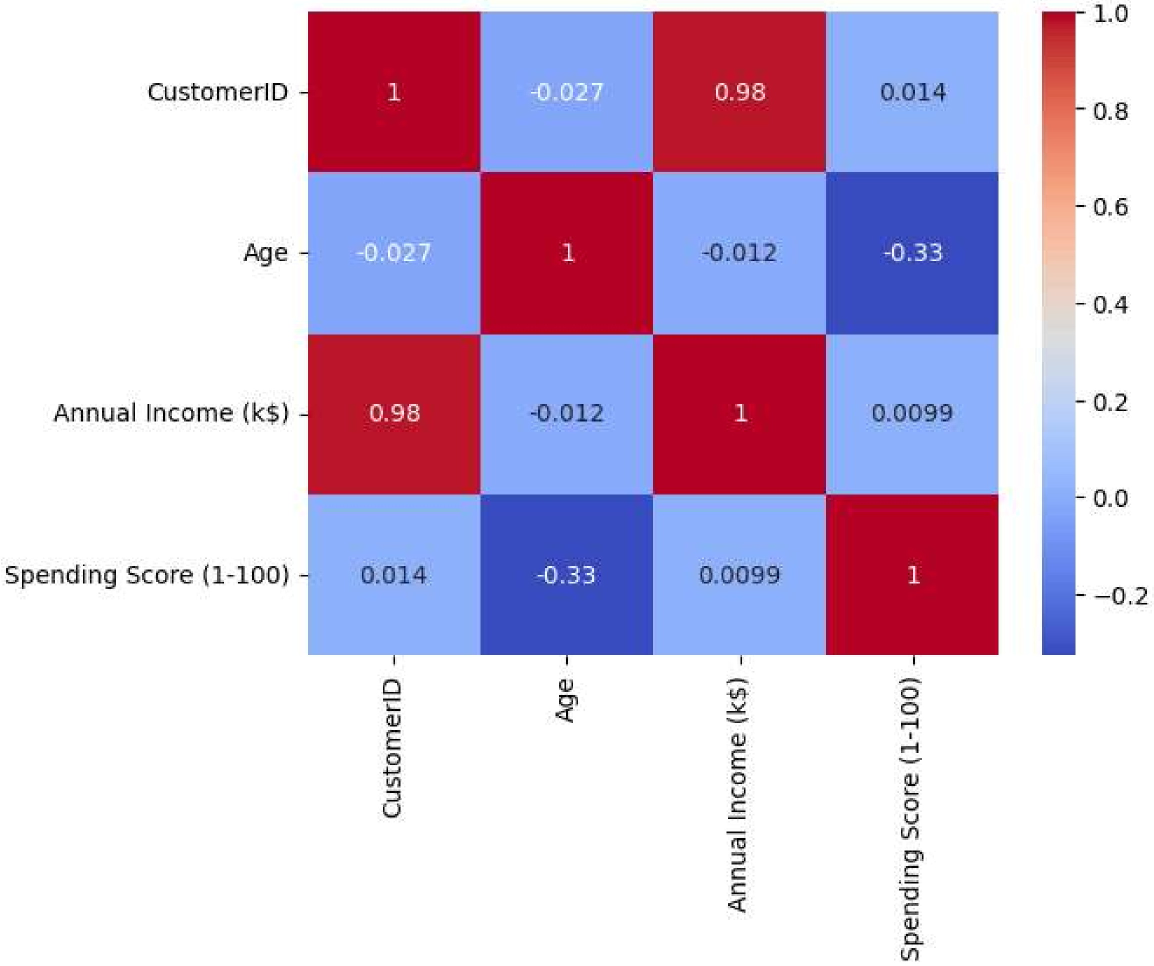
	CustomerID	Age	Annual Income (k$)	Spending Score (1-100)
CustomerID	1.00	-0.03	0.98	0.01
Age	-0.03	1.00	-0.01	-0.33
Annual Income (k$)	0.98	-0.01	1.00	0.01
Spending Score (1-100)	0.01	-0.33	0.01	1.00

In [ ]:

```
sns.heatmap(data.corr(numeric_only=True), annot=True, cmap="coolwarm")
```

Out[ ]:

```
<Axes: >
```

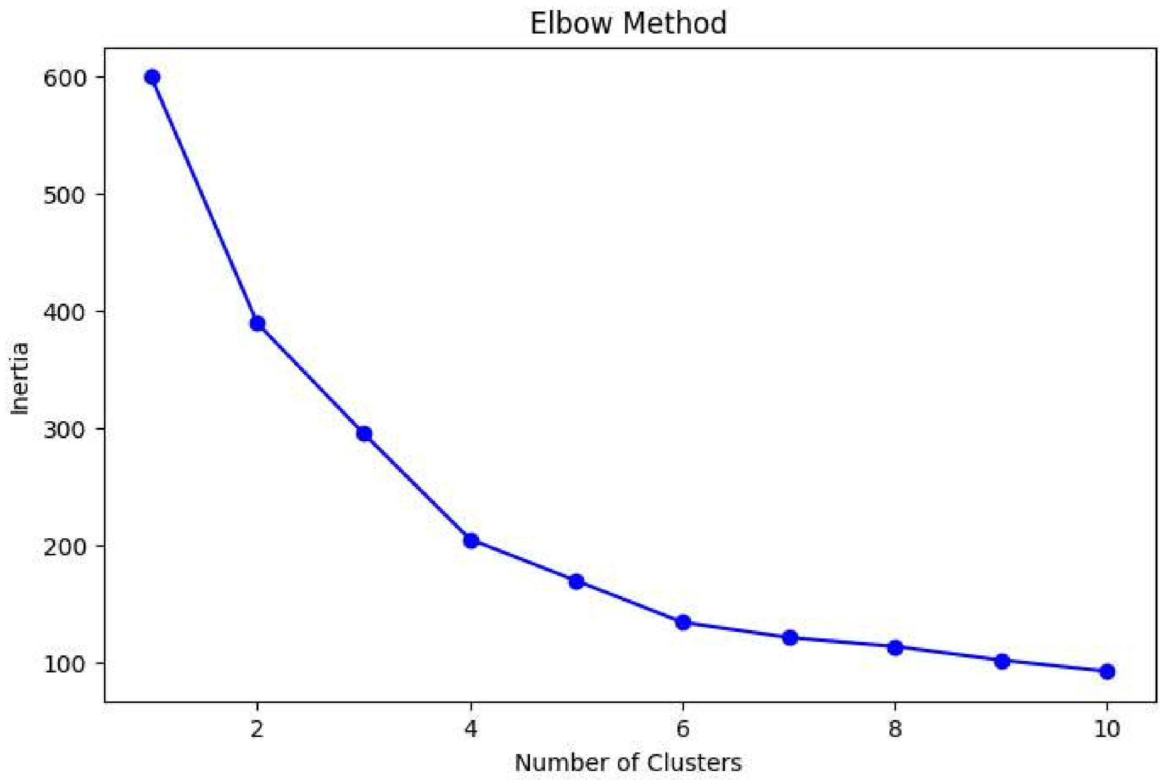


In [ ]:
```
# Select relevant features for clustering
# Assuming the dataset has columns like 'Age', 'Annual Income (k$)', 'Spending S
features = data[['Age', 'Annual Income (k$)', 'Spending Score (1-100)']]

# Preprocess the data
scaler = StandardScaler()
scaled_features = scaler.fit_transform(features)
```

In [ ]:
```
# Determine the optimal number of clusters using the Elbow method
inertia = []
K_range = range(1, 11)
for k in K_range:
    kmeans = KMeans(n_clusters=k, random_state=42)
    kmeans.fit(scaled_features)
    inertia.append(kmeans.inertia_)
```

In [ ]:
```
# Plot the Elbow curve
plt.figure(figsize=(8, 5))
plt.plot(K_range, inertia, marker='o', color='blue')
plt.title('Elbow Method')
plt.xlabel('Number of Clusters')
plt.ylabel('Inertia')
plt.show()
```



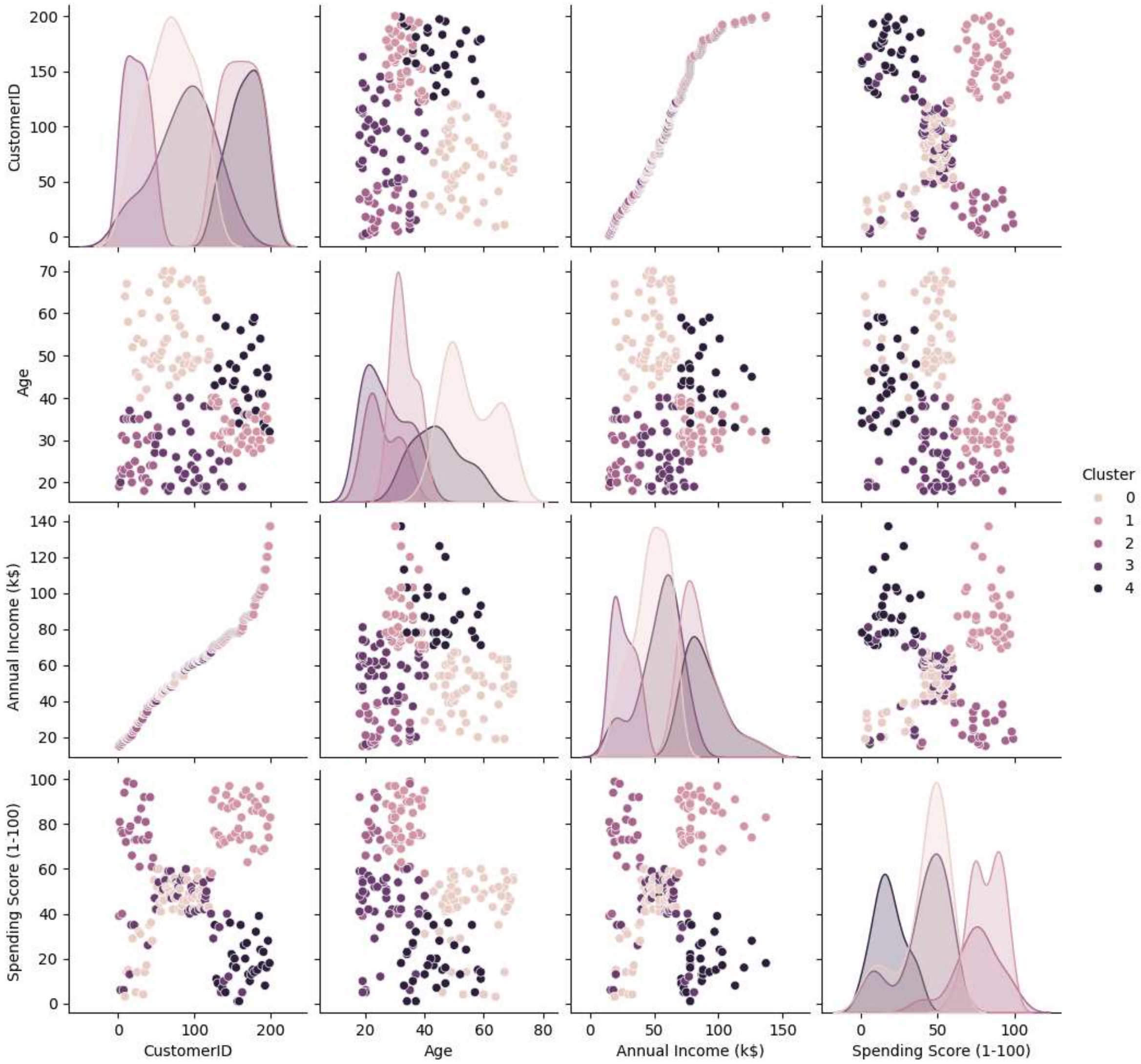
In [ ]:

```
# Apply K-means with optimal number of clusters (e.g., k=5 based on the Elbow me
optimal_k = 5
kmeans = KMeans(n_clusters=optimal_k, random_state=42)
clusters = kmeans.fit_predict(scaled_features)
```

In [ ]:

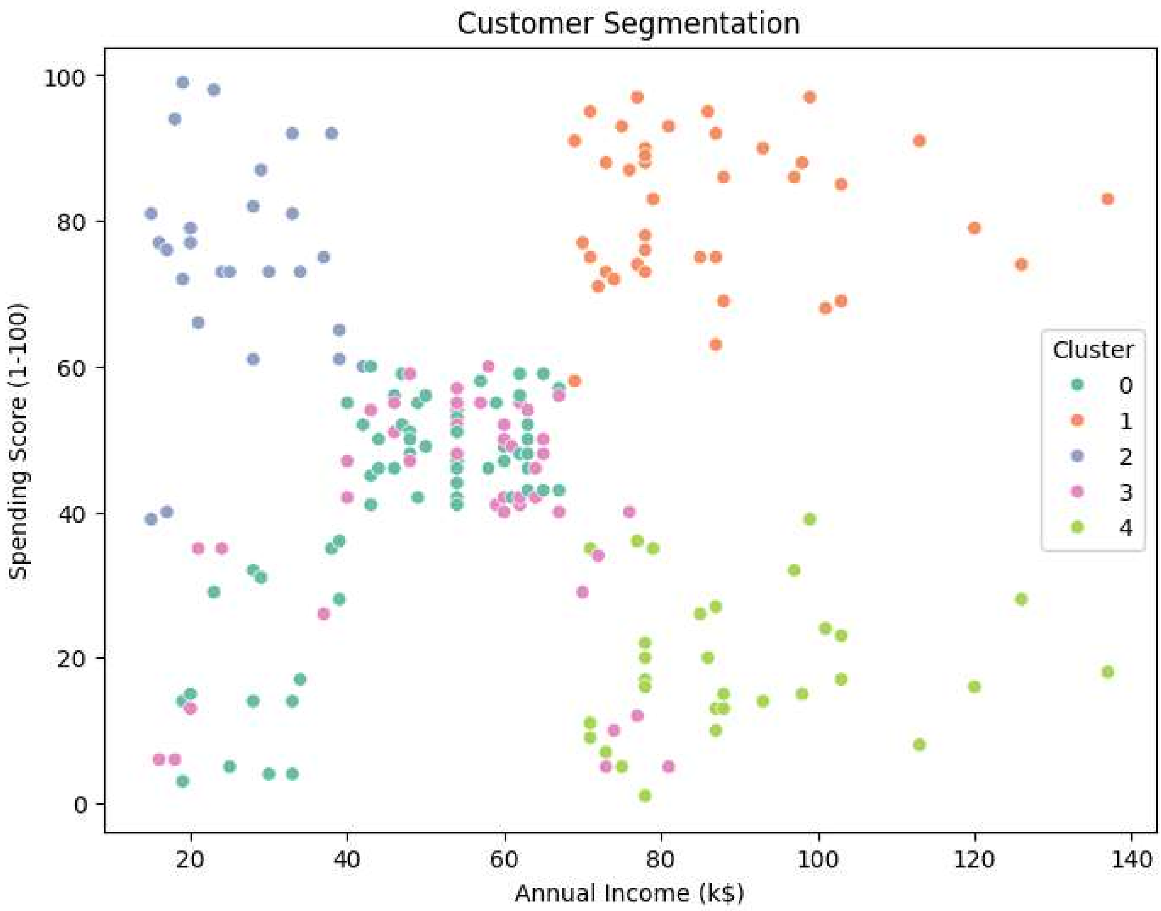
```
# Add cluster labels to the original dataset
data['Cluster'] = clusters

# Analyze and visualize the clusters
sns.pairplot(data, hue='Cluster', diag_kind='kde')
plt.show()
```



In [ ]:

```
# Additional visualization (e.g., income vs. spending score)
plt.figure(figsize=(8, 6))
sns.scatterplot(
    x='Annual Income (k$)', y='Spending Score (1-100)',
    hue='Cluster', data=data, palette='Set2'
)
plt.title('Customer Segmentation')
plt.show()
```



In [ ]:

```
#Save the clustered data
data.to_csv('clustered_customers.csv', index=False)
print("Clustered data saved to 'clustered_customers.csv'.")
```

```
Clustered data saved to 'clustered_customers.csv'.
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: