Customer Segmentation using Unsupervised Learning

Introduction:

In the ever-evolving landscape of retail, understanding customer behavior is crucial for businesses seeking a competitive edge. Selore Nigeria, an electronics retail chain, is utilizing unsupervised learning techniques to analyze demographic information and spending habits to identify customer segments, thereby enhancing personalized marketing and strategic optimization.

Problem Statement:

Despite its position as a leading retailer, Selore Nigeria faces the challenge of tailoring its marketing strategies to the diverse preferences and spending behaviors of its customers. A lack of insights into customer segmentation hinders the company's ability to deliver personalized experiences. This project addresses this problem by employing unsupervised learning to categorize customers based on their purchasing behavior and demographic characteristics.

Project Objectives:

The project aims to use unsupervised learning techniques to segment customers, uncover insights, develop targeted marketing strategies, and optimize pricing and product recommendations for Selore Nigeria, enhancing its customer understanding and business strategies.

In [1]:
```
# Import necessary Libraries

# For data analysis
import pandas as pd
import numpy as np

# For data visualization
import matplotlib.pyplot as plt
import seaborn as sns

# Machine learning and evaluation
from sklearn.cluster import KMeans
from scipy.cluster.hierarchy import dendrogram, linkage
from sklearn.metrics import silhouette_score, homogeneity_score

import warnings
warnings.filterwarnings("ignore")
```

In [2]:
```
# Load data
data = pd.read_csv(r"C:\Users\ADMIN\Desktop\Resources\10Alytics Data Science\Machine Learni
data.head()
```

Out[2]:

	CustomerID	Genre	Age	Annual_Income_(k$)	Spending_Score
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 [3]:
```
data.shape
```

Out[3]:
```
(200, 5)
```

In [4]:
```
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   Genre                   200 non-null    object
 2   Age                     200 non-null    int64 
 3   Annual_Income_(k$)      200 non-null    int64 
 4   Spending_Score          200 non-null    int64 
dtypes: int64(4), object(1)
memory usage: 7.9+ KB
```

In [5]:
```
data.describe()
```

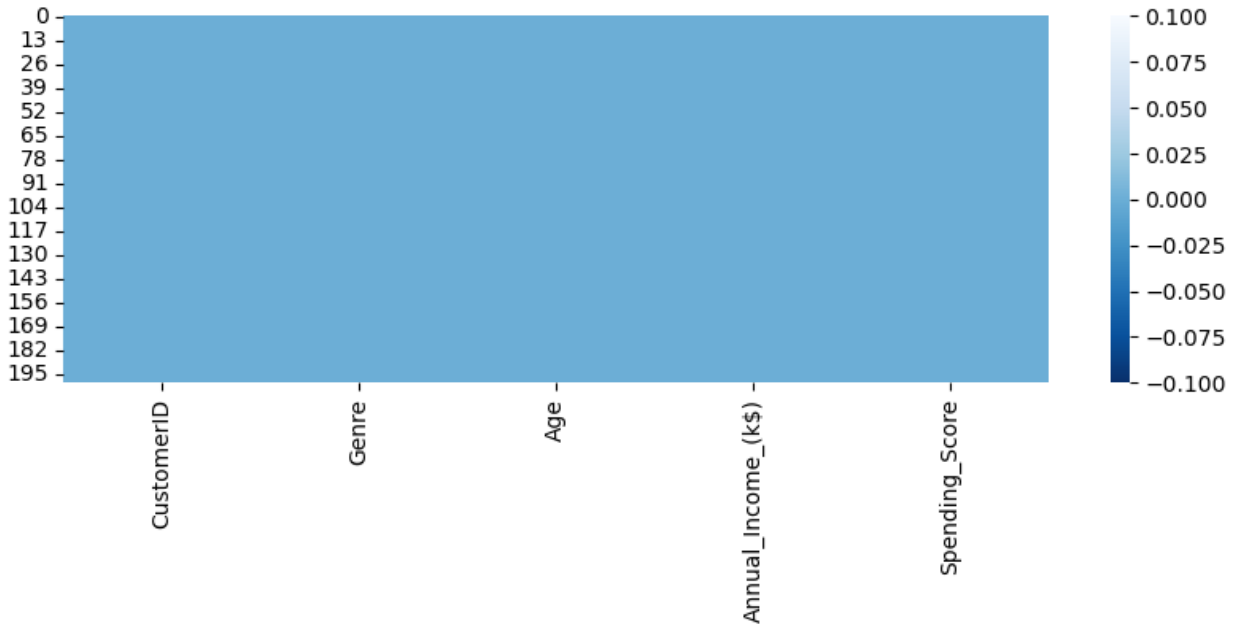
Out[5]:

	CustomerID	Age	Annual_Income_(k$)	Spending_Score
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 [6]:
```
# Check for missing values
print(data.isnull().sum())

# Visualization the missing data
plt.figure(figsize = (10,3))
sns.heatmap(data.isnull(), cbar=True, cmap="Blues_r")
```

```
CustomerID              0
Genre                   0
Age                     0
Annual_Income_(k$)      0
Spending_Score          0
dtype: int64
```

Out[6]: <Axes: >



Data Cleaning and Pre Processing

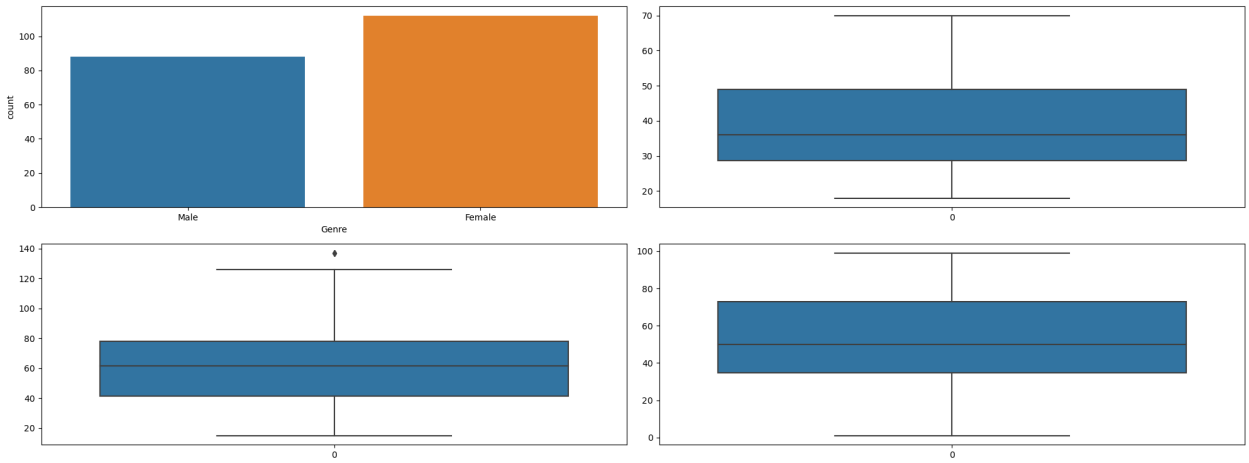
In [7]:
```
# Check for duplicate
print(data.duplicated().sum())
```

```
0
```

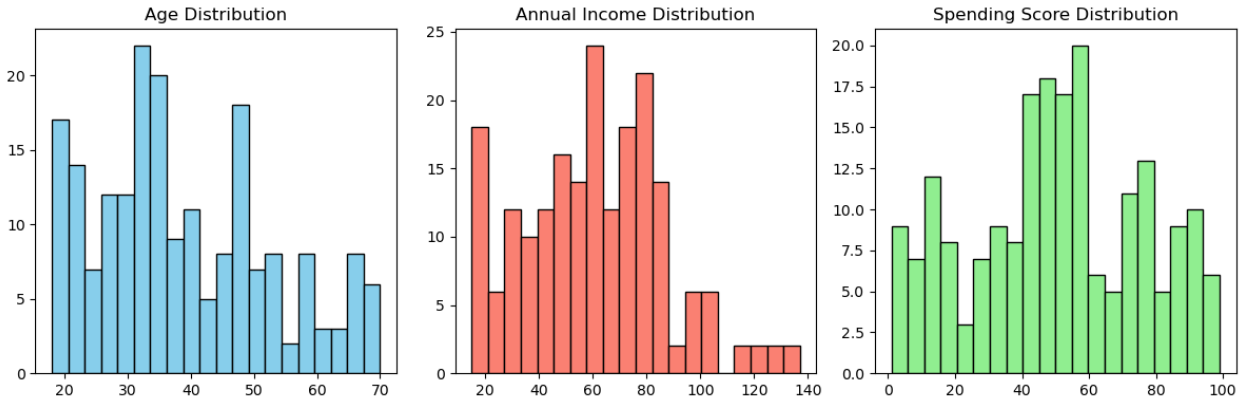
Exploratory Data Analysis

Univariate

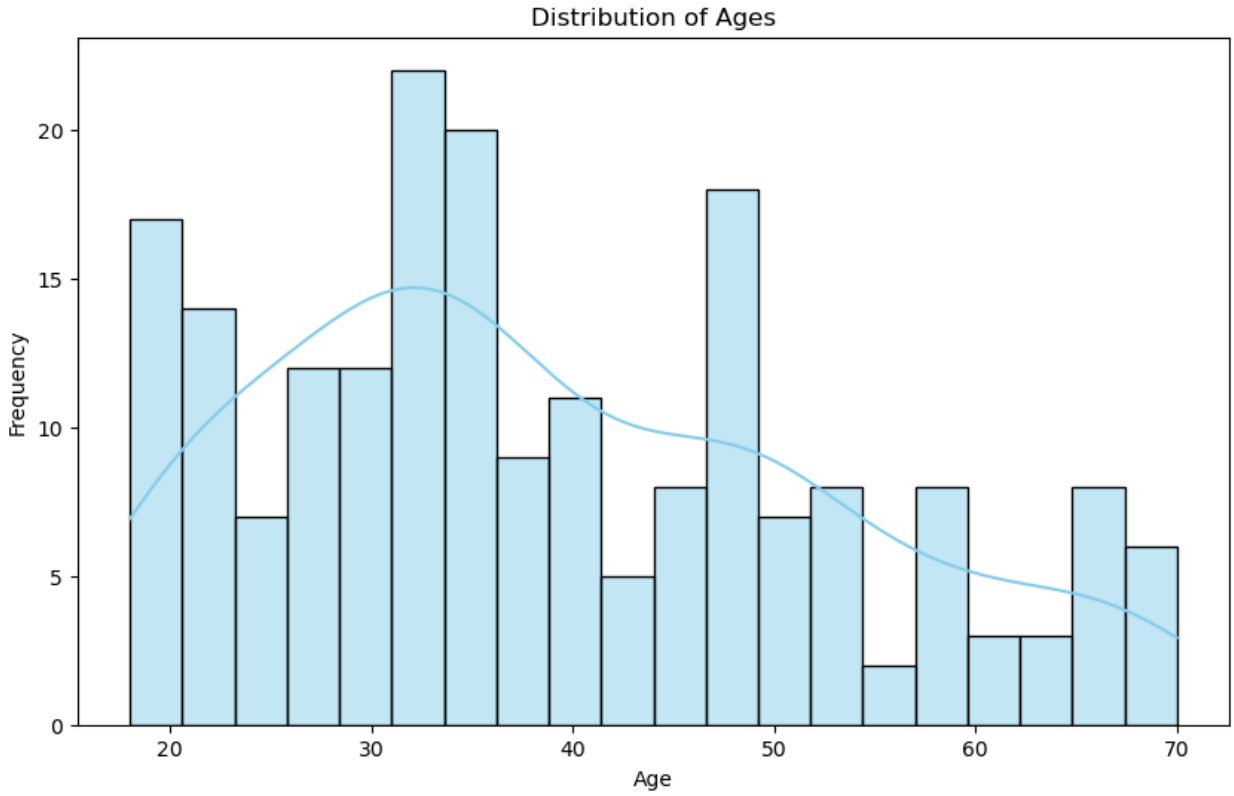
In [8]:
```
fig, axs = plt.subplots(2,2, figsize = (20, 7.5))
plot1 = sns.countplot(x=data['Genre'], ax = axs[0,0])
plot2 = sns.boxplot(data['Age'], ax = axs[0,1])
plot3 = sns.boxplot(data['Annual_Income_(k$)'], ax = axs[1,0])
plot4 = sns.boxplot(data['Spending_Score'], ax = axs[1,1])

plt.tight_layout()
```



In [9]:
```
import matplotlib.pyplot as plt

# Histograms
plt.figure(figsize=(12, 4))

plt.subplot(1, 3, 1)
plt.hist(data['Age'], bins=20, color='skyblue', edgecolor='black')
plt.title('Age Distribution')

plt.subplot(1, 3, 2)
plt.hist(data['Annual_Income_(k$)'], bins=20, color='salmon', edgecolor='black')
plt.title('Annual Income Distribution')

plt.subplot(1, 3, 3)
plt.hist(data['Spending_Score'], bins=20, color='lightgreen', edgecolor='black')
plt.title('Spending Score Distribution')

plt.tight_layout()
plt.show()
```

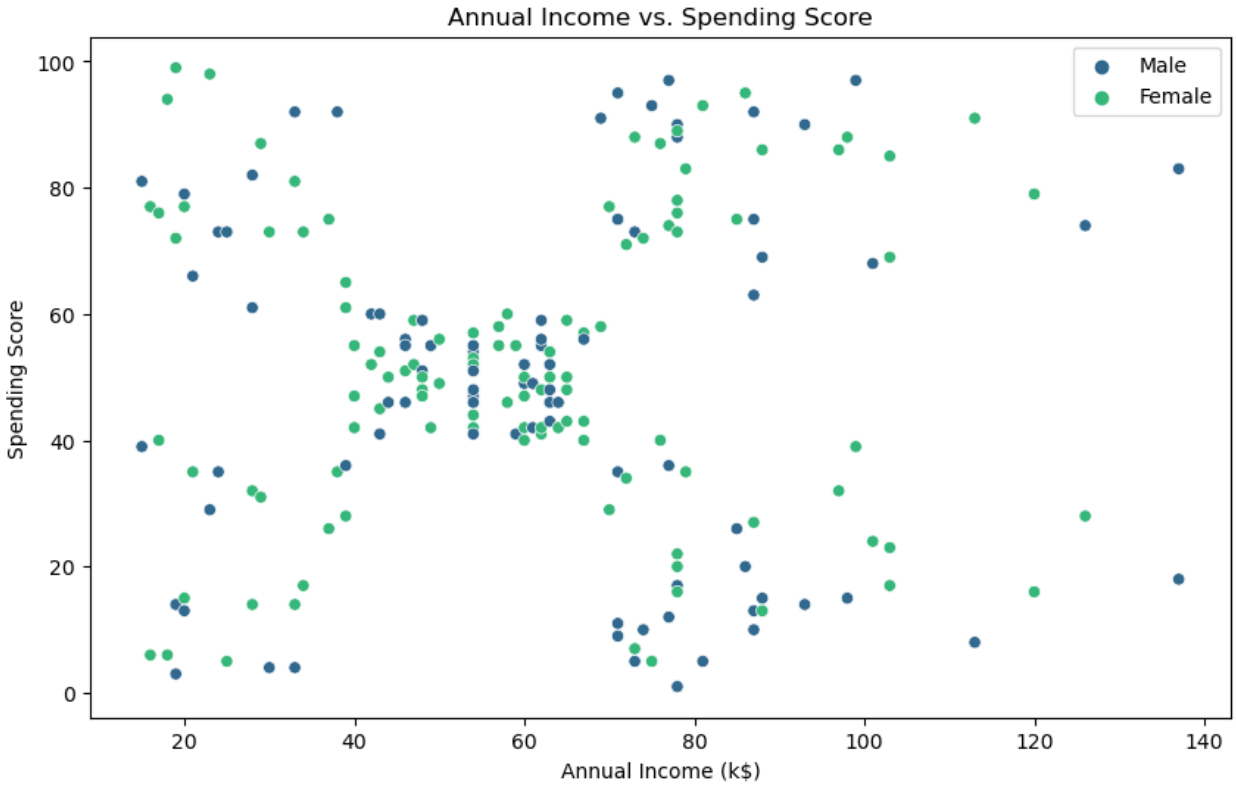


In [10]:
```
import matplotlib.pyplot as plt
import seaborn as sns

# Univariate Analysis - Age Distribution
plt.figure(figsize=(10, 6))
sns.histplot(data['Age'], bins=20, kde=True, color='skyblue')
plt.title('Distribution of Ages')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.show()
```



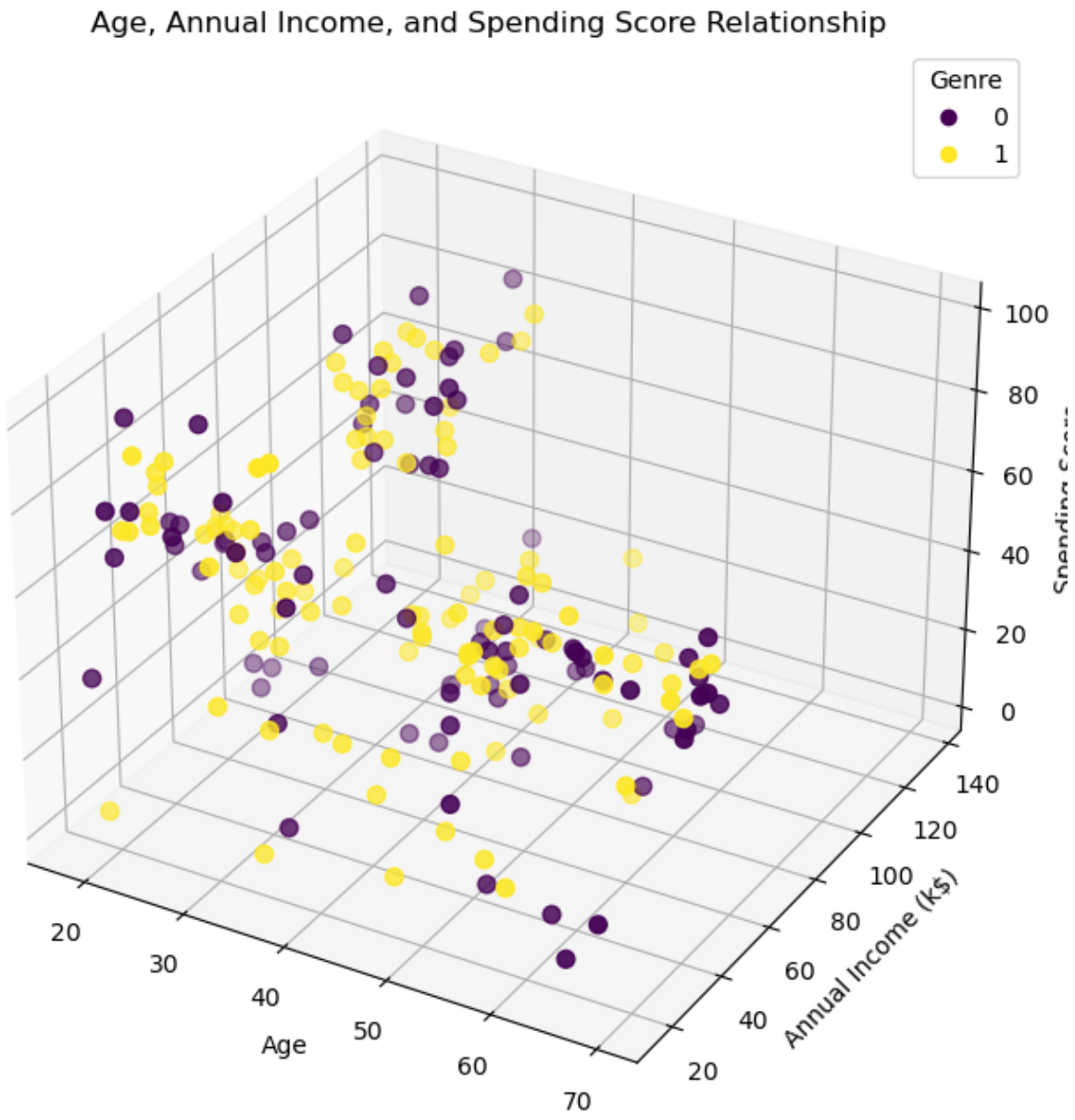
Bivariate Analysis

In [11]:
```
# Bivariate Analysis - Annual Income vs. Spending Score
plt.figure(figsize=(10, 6))
sns.scatterplot(x='Annual_Income_(k$)', y='Spending_Score', data=data, hue='Genre', palette
plt.title('Annual Income vs. Spending Score')
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score')
plt.legend()
plt.show()
```

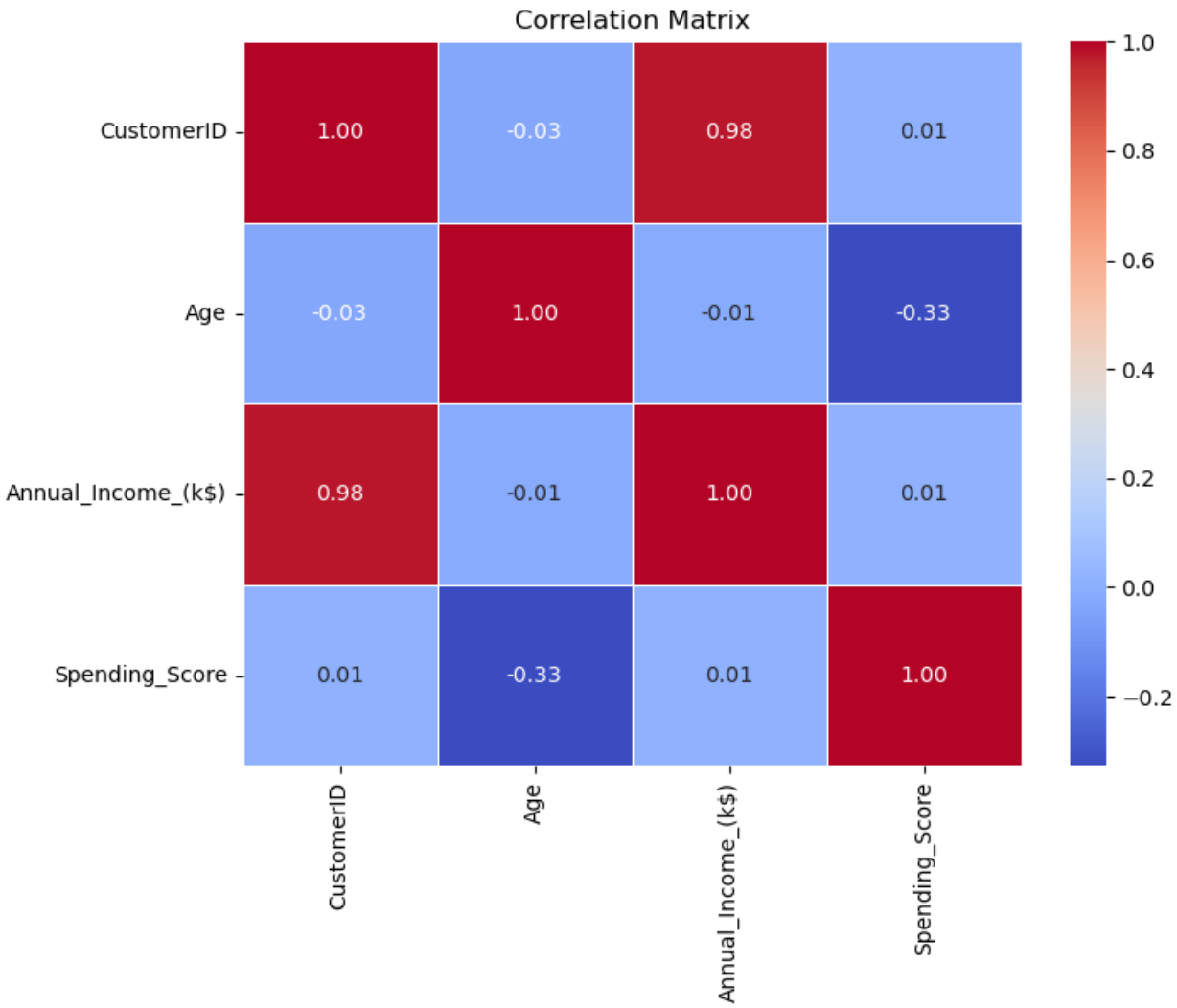


Multivariate Analysis

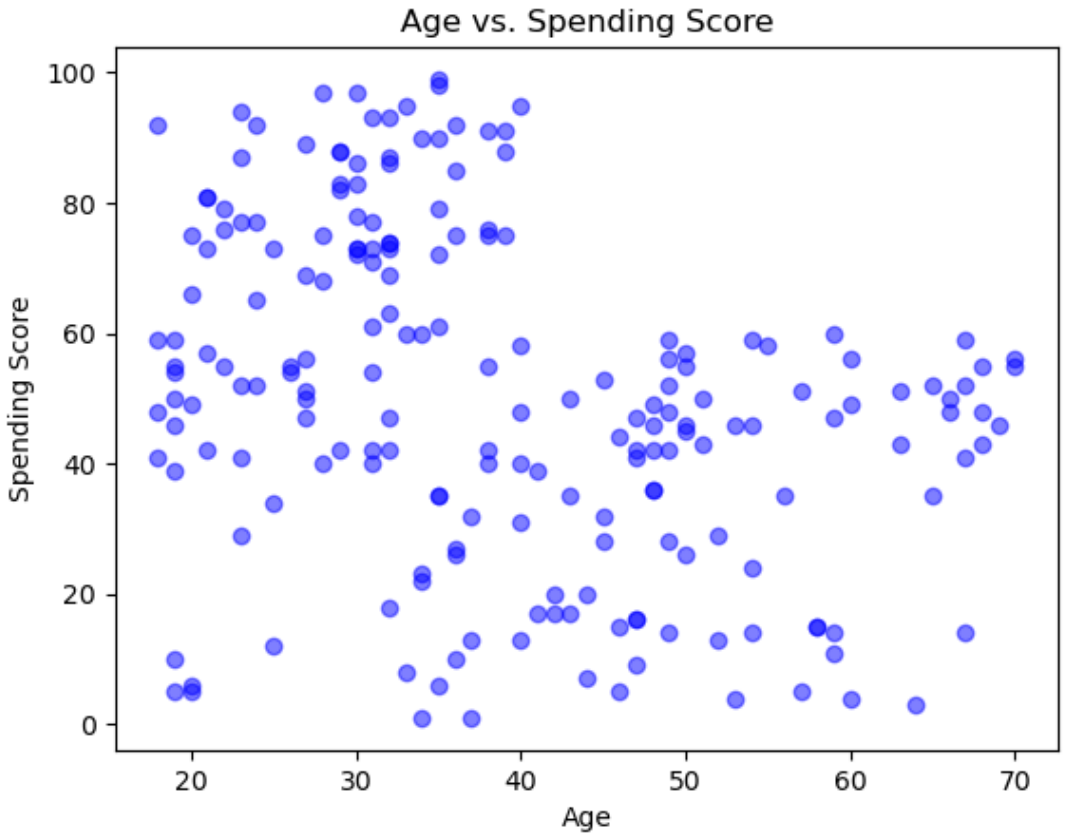
In [12]:
```
from mpl_toolkits.mplot3d import Axes3D

# Multivariate Analysis - Age, Annual Income, and Spending Score Relationship
fig = plt.figure(figsize=(12, 8))
ax = fig.add_subplot(111, projection='3d')
scatter = ax.scatter(data['Age'], data['Annual_Income_(k$)'], data['Spending_Score'], c=dat

ax.set_xlabel('Age')
ax.set_ylabel('Annual Income (k$)')
ax.set_zlabel('Spending Score')
ax.set_title('Age, Annual Income, and Spending Score Relationship')
ax.legend(*scatter.legend_elements(), title='Genre')

plt.show()
```



In [13]:
```
# Correlation matrix heatmap
plt.figure(figsize=(8, 6))
correlation_matrix = data.corr()
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f", linewidths=.5)
plt.title('Correlation Matrix')
plt.show()
```

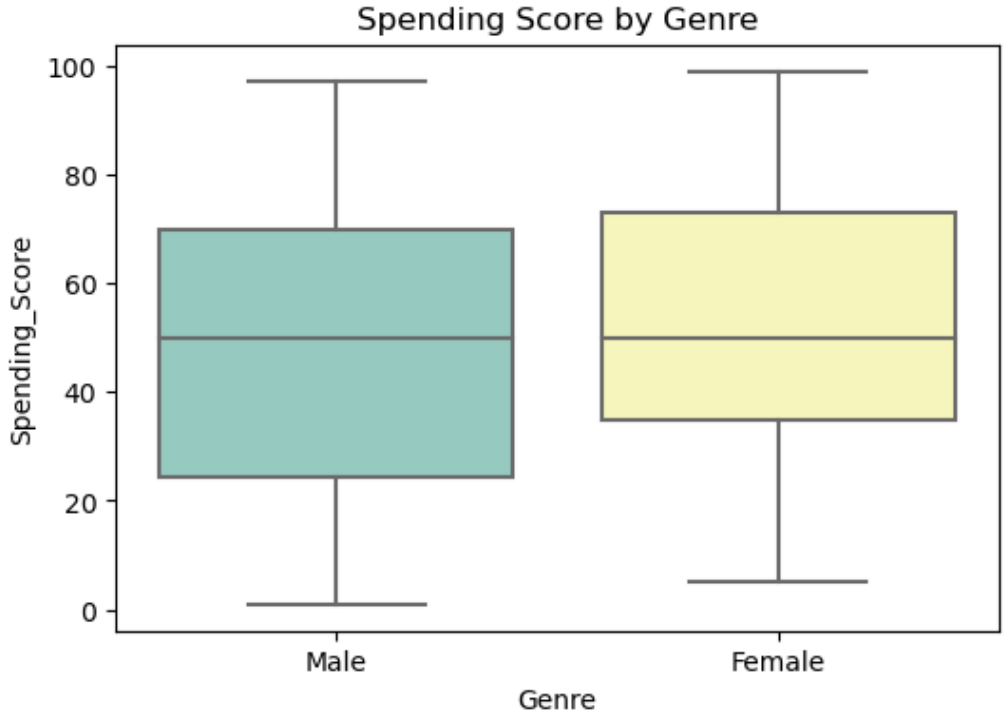


In [14]:
```
# Scatter plot for Age vs. Spending_Score
plt.scatter(data['Age'], data['Spending_Score'], color='blue', alpha=0.5)
plt.title('Age vs. Spending Score')
plt.xlabel('Age')
plt.ylabel('Spending Score')
plt.show()
```



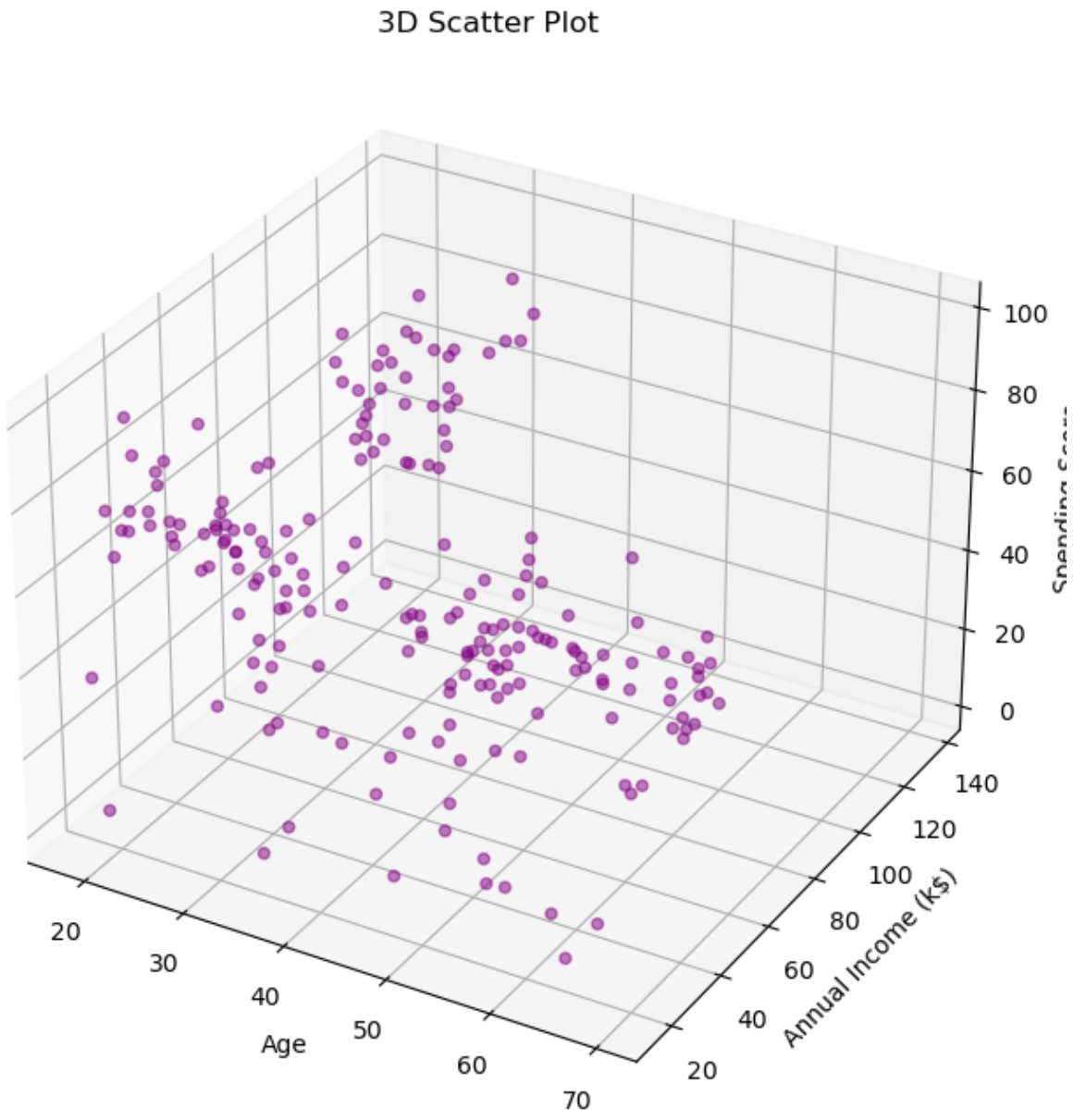
In [15]:
```
# Scatter plot for Annual_Income_(k$) vs. Spending_Score
plt.scatter(data['Annual_Income_(k$)'], data['Spending_Score'], color='red', alpha=0.5)
plt.title('Annual Income vs. Spending Score')
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score')
plt.show()
```

In [16]:
```
# Box plot for Spending_Score by Genre
plt.figure(figsize=(6, 4))
sns.boxplot(x='Genre', y='Spending_Score', data=data, palette='Set3')
plt.title('Spending Score by Genre')
plt.show()
```

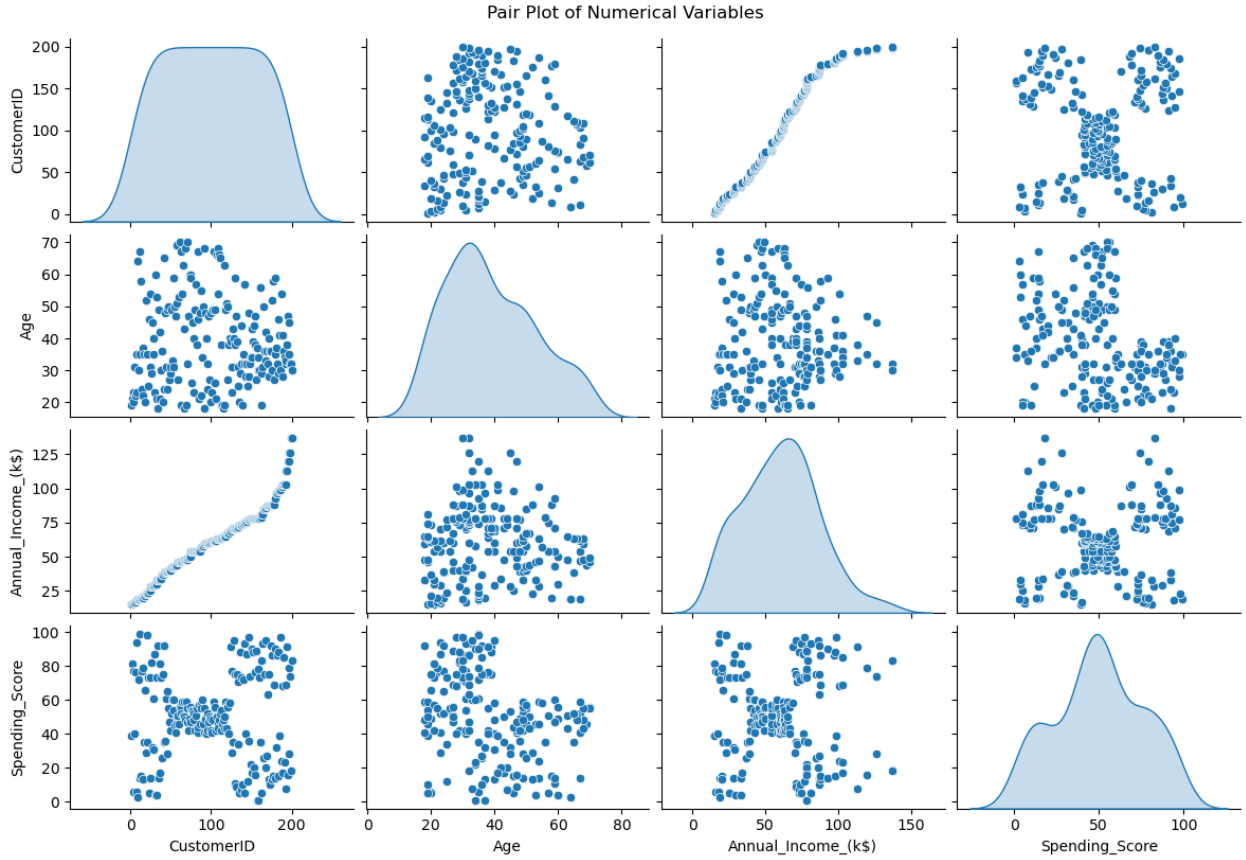


In [17]:
```
from mpl_toolkits.mplot3d import Axes3D

# 3D Scatter plot
```

```
fig = plt.figure(figsize=(10, 8))
ax = fig.add_subplot(111, projection='3d')
ax.scatter(data['Age'], data['Annual_Income_(k$)'], data['Spending_Score'], c='purple', mar
ax.set_xlabel('Age')
ax.set_ylabel('Annual Income (k$)')
ax.set_zlabel('Spending Score')
plt.title('3D Scatter Plot')
plt.show()
```



In [18]:
```
# Pair plot for all numerical variables
sns.pairplot(data, height=2, aspect=1.5, diag_kind='kde')
plt.suptitle('Pair Plot of Numerical Variables', y=1.02)
plt.show()
```



In [19]:
```
## Model Building

from sklearn.preprocessing import LabelEncoder

# Label Encoding (when you're doing ML, change the categorical variables (male & female) to

encoder = LabelEncoder() # method initialization

# Looping for columns except survived

for c in data.columns[1:]:
    if(data[c].dtype=='object'):
        data[c] = encoder.fit_transform(data[c])
    else:
        data[c] = data[c]

data.head()
```

Out[19]:

	CustomerID	Genre	Age	Annual_Income_(k$)	Spending_Score
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

In [20]:
```
# Remove CustomerID column
data.drop('CustomerID', axis=1, inplace=True)
```

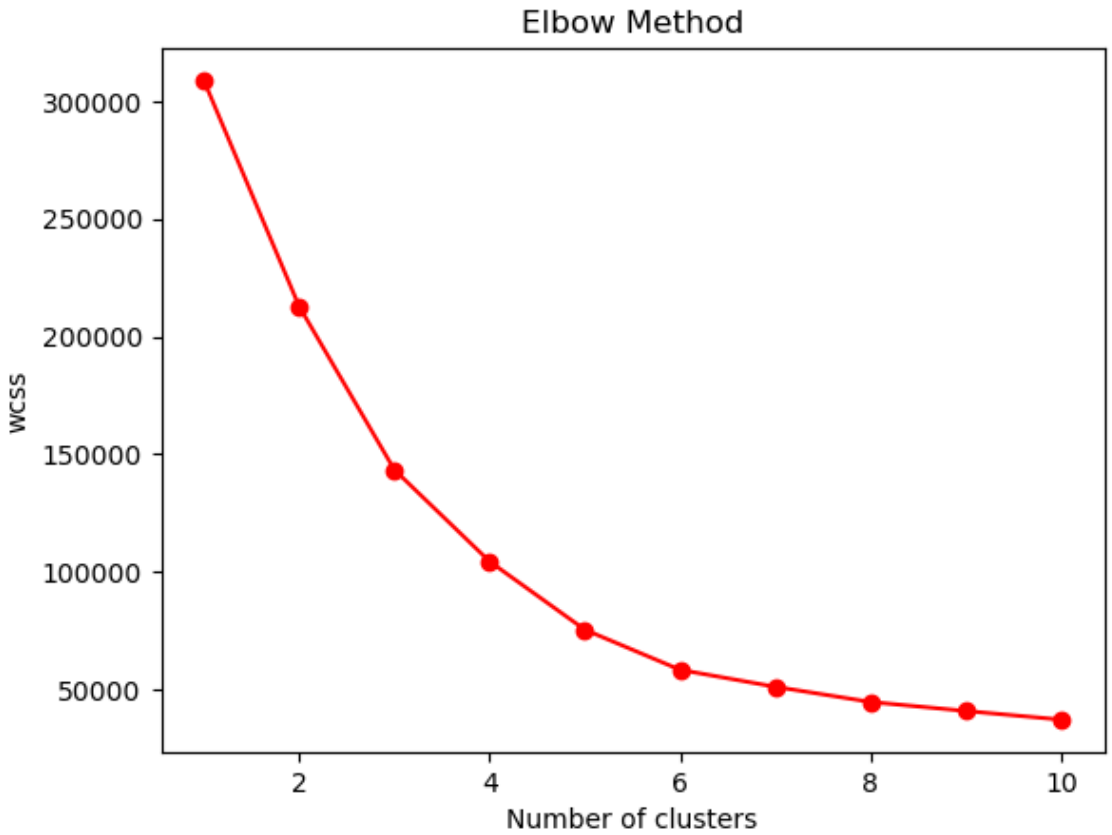
```
data.head()
```

Out[20]:

	Genre	Age	Annual_Income_(k$)	Spending_Score
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

In [21]:

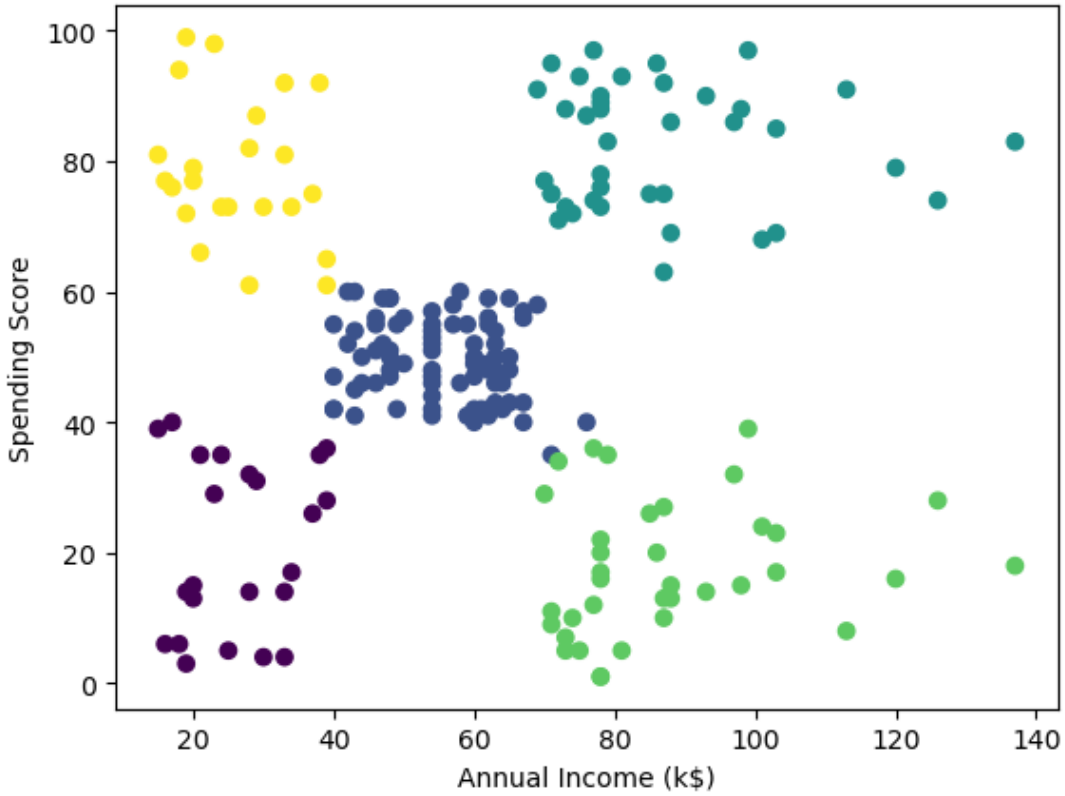
```
# Determine optimal number of clusters

wcss = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters=i, init='k-means++', random_state=42)
    kmeans.fit(data)
    wcss.append(kmeans.inertia_)
plt.plot(range(1, 11), wcss, color='red', marker='o')
plt.title('Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('wcss')
plt.show()
```



In [22]:

```
# Fit K-means clustering model
kmeans = KMeans(n_clusters=5, init='k-means++', random_state=42)
kmeans.fit(data)

# Add cluster label to data
data['Cluster'] = kmeans.labels_
```

In [23]:
```
# Visualize cluster results
plt.scatter(data['Annual_Income_(k$)'], data['Spending_Score'], c=data['Cluster'], label='C
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score')
plt.show()

# Calculate silhoutte score
from sklearn.metrics import silhouette_score

silhouette_score(data, kmeans.labels_)
```



Out[23]:
```
0.44473703994455477
```

Key Insights:

The analysis has revealed five distinct customer segments with unique characteristics:

- Segment dark green: Customers in this segment exhibit high spending scores and moderate to high annual incomes. They represent a prime audience for premium products and personalized services.
- Segment blue: This segment consists of customers with moderate spending scores and annual incomes. Targeted marketing campaigns focusing on value and affordability are likely to resonate with this group.
- Segment pink: Customers in this segment display low spending scores and lower annual incomes. Strategies should be devised to encourage increased spending, potentially through budget-friendly offers and incentives.
- Segment yellow: This segment showcases high spending scores but with lower annual incomes. Crafting promotions and offers that align with their preferences while considering financial constraints is key.

- Segment light green: Customers in this segment have moderate spending scores and higher annual incomes. They may respond well to exclusive offers and loyalty programs.

The silhouette score of 0.44 indicates a good separation between the clusters, reinforcing the robustness of the segmentation model. It suggests that the clusters in the data have a decent level of separation and cohesion. This indicates that the data points within each cluster are relatively close to each other, and the clusters are well-separated from each other.

Conclusions:

The customer segmentation model has successfully identified distinct groups within Selore Nigeria's customer base. By understanding the diverse needs and preferences of these segments, the company can now craft targeted marketing campaigns to enhance customer engagement and satisfaction. The identified clusters provide a foundation for strategic decision-making, allowing Selore Nigeria to allocate resources effectively and optimize its approach to customer interactions.

Recommendations:

- Tailored Marketing Campaigns: Develop personalized marketing campaigns for each segment, emphasizing products and promotions that resonate with the specific preferences and purchasing behaviors of each group.
- Optimized Pricing Strategies: Adjust pricing strategies based on the identified segments. Implement tiered pricing or bundle offers to cater to the diverse financial capacities and spending tendencies of each segment.
- Enhanced After-Sales Support: Strengthen after-sales services, particularly for customers in Segment 1 who are likely to invest in premium products. Offering extended warranties, exclusive repair services, or loyalty programs can further enhance customer satisfaction.
- Product Recommendations: Leverage insights from each segment to enhance the product recommendation engine. Provide personalized suggestions based on the preferences and behaviors of customers in different clusters.
- Continuous Monitoring: Regularly reassess and refine the segmentation model as customer behaviors evolve over time. Stay adaptive to market trends and changes in consumer preferences.

By implementing these recommendations, Selore Nigeria can create a more personalized and responsive shopping experience, fostering customer loyalty and contributing to long-term business success.

In [ ]: