eda-analysis-task-2

November 6, 2024

EDA Task on Mall Customers4.visualizing relationships between variables 5.understanding multi-variate analysis 6.Correlation analysis 7.feature importance, and data visualization 8.Clustering, customer segmentation

[42]:
```
import pandas as pd


file_path = r"C:\Users\divaa\OneDrive\Desktop\pri\Bliend\Bliend␣
↪dataset\Mall_Customers.csv"
df = pd.read_csv(file_path)


df.head()
```

[42]:
```
   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
```

[43]:
```
# Check for missing values
df.isnull().sum()
```

[43]:
```
CustomerID                0
Gender                    0
Age                       0
Annual Income (k$)        0
Spending Score (1-100)    0
dtype: int64
```

[44]:
```
# Check for duplicate records
df.duplicated().sum()
```

[44]:
```
0
```

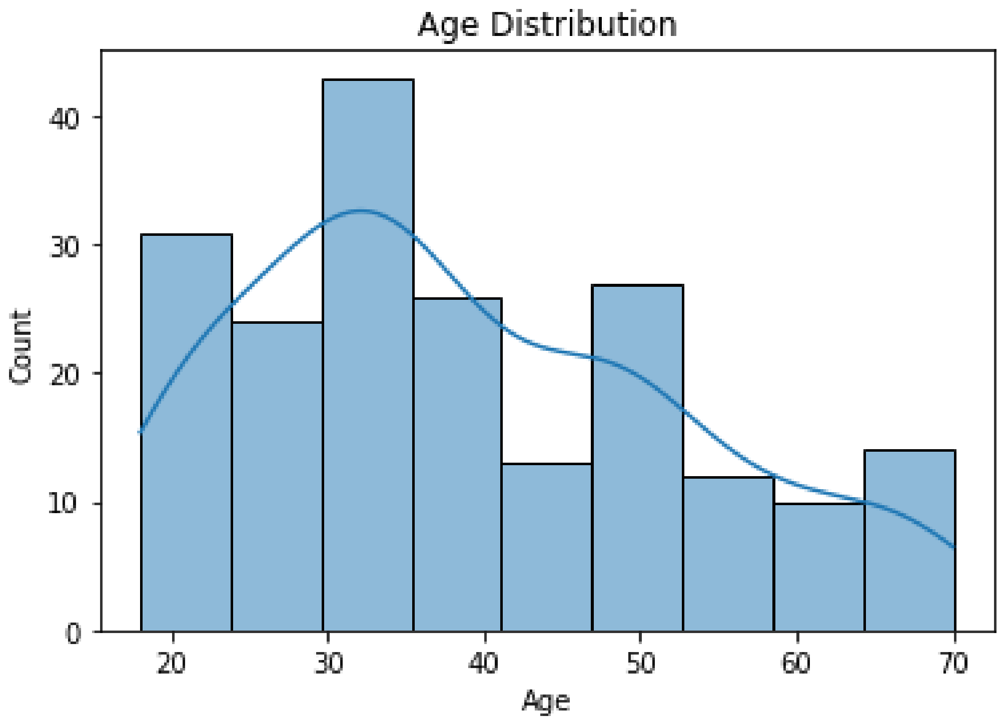
[45]:
```
df.dtypes
```

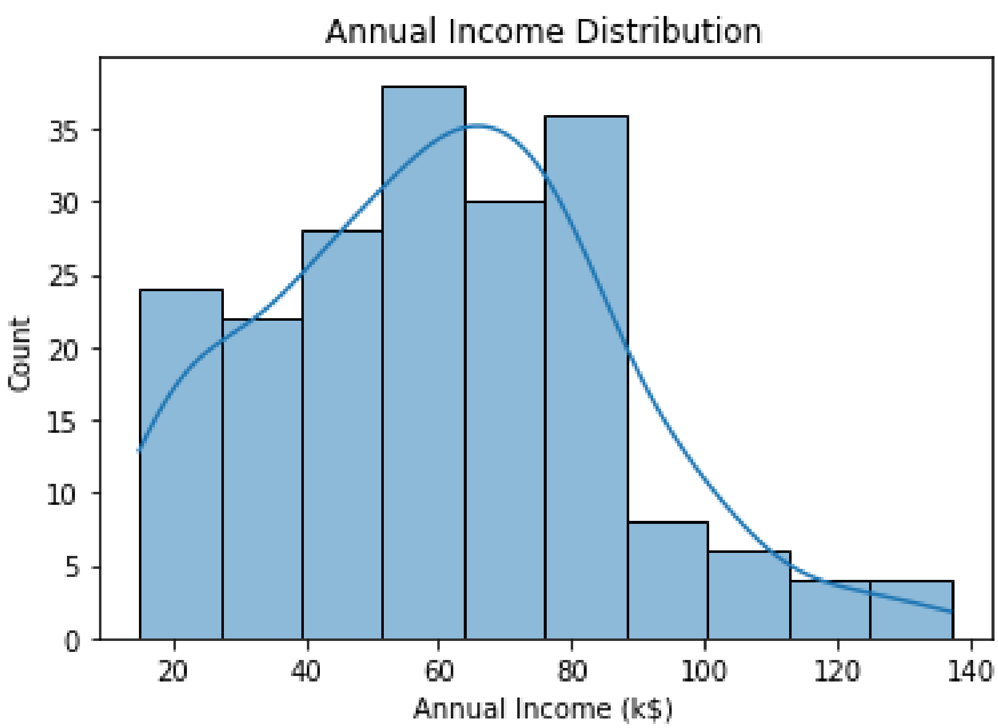
[45]:
```
CustomerID                  int64
Gender                     object
Age                         int64
Annual Income (k$)          int64
Spending Score (1-100)      int64
dtype: object
```

[46]:
```
df.describe()

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

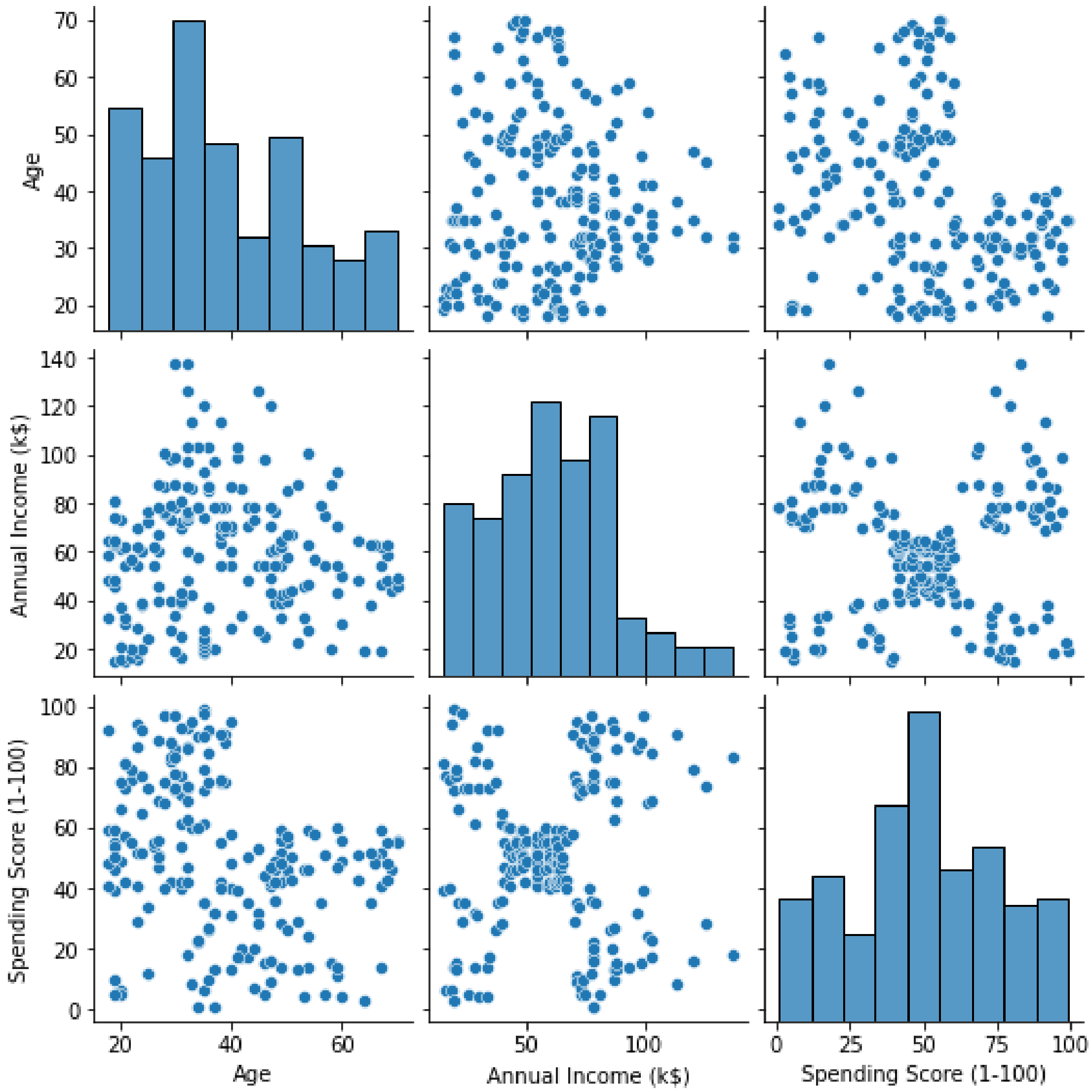
[46]:
```
Female    112
Male       88
Name: Gender, dtype: int64
```

[47]:
```
import seaborn as sns
import matplotlib.pyplot as plt

# Distribution plots for numerical columns
sns.histplot(df['Age'], kde=True)
plt.title('Age Distribution')
plt.show()

sns.histplot(df['Annual Income (k$)'], kde=True)
plt.title('Annual Income Distribution')
plt.show()

sns.histplot(df['Spending Score (1-100)'], kde=True)
plt.title('Spending Score Distribution')
plt.show()

# Pairplot to visualize relationships between features
sns.pairplot(df[['Age', 'Annual Income (k$)', 'Spending Score (1-100)']])
plt.show()
```


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


Annual Income Distribution
Count
Annual Income (k$)
0
5
10
15
20
25
30
35
20
40
60
80
100
120
140

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



[48]:
```
df.isnull().sum()


df = df.drop_duplicates()

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

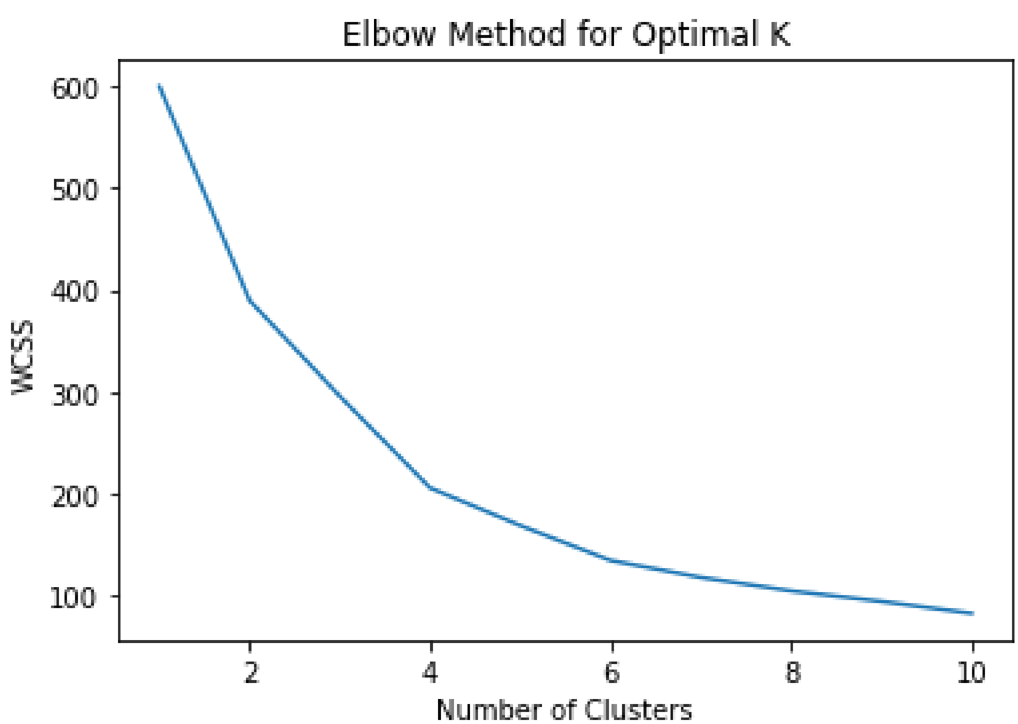
[49]:
```
X = df[['Age', 'Annual Income (k$)', 'Spending Score (1-100)']]
```

[50]:
```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

[51]:

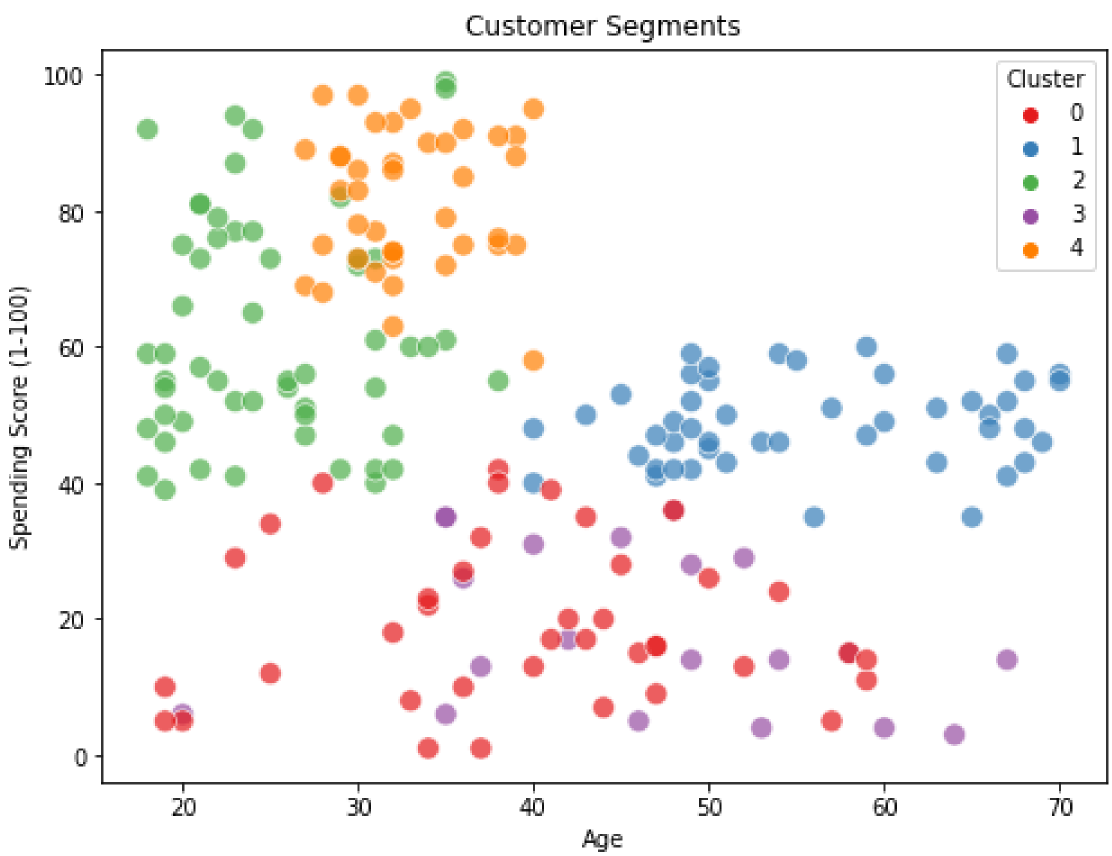
```
from sklearn.cluster import KMeans
wcss = []

for i in range(1, 11):
    kmeans = KMeans(n_clusters=i, init='k-means++', max_iter=300, n_init=10,␣
 ↪random_state=0)
    kmeans.fit(X_scaled)
    wcss.append(kmeans.inertia_)


plt.plot(range(1, 11), wcss)
plt.title('Elbow Method for Optimal K')
plt.xlabel('Number of Clusters')
plt.ylabel('WCSS')
plt.show()
```

```
C:\Users\divaa\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:1036:
UserWarning: KMeans is known to have a memory leak on Windows with MKL, when
there are less chunks than available threads. You can avoid it by setting the
environment variable OMP_NUM_THREADS=1.
  warnings.warn(
```



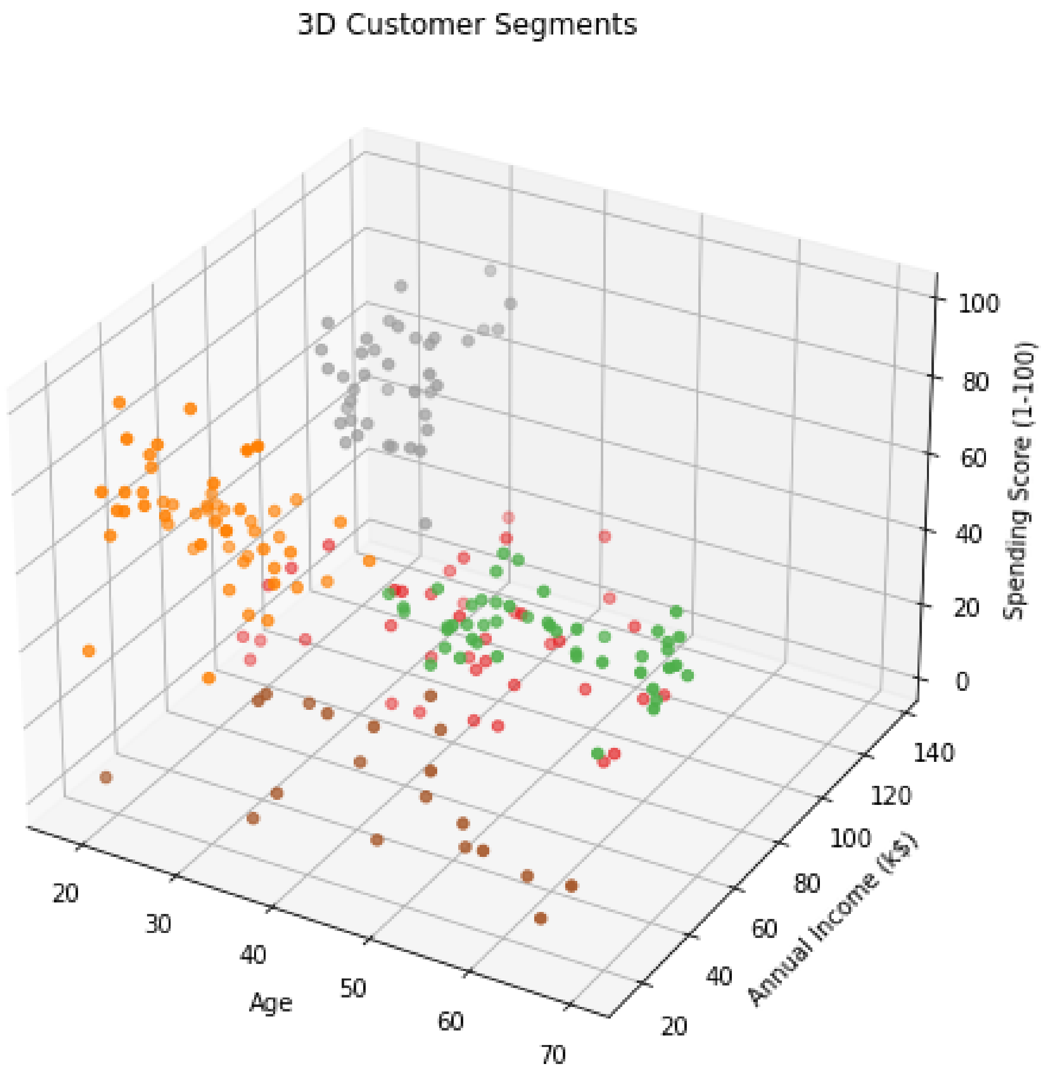
[52]:
```
kmeans = KMeans(n_clusters=5, init='k-means++', max_iter=300, n_init=10,␣
↪random_state=0)
df['Cluster'] = kmeans.fit_predict(X_scaled)
```

[53]:
```
plt.figure(figsize=(8, 6))
sns.scatterplot(x=df['Age'], y=df['Spending Score (1-100)'], hue=df['Cluster'],␣
↪palette='Set1', s=100, alpha=0.7)
plt.title('Customer Segments')
plt.xlabel('Age')
plt.ylabel('Spending Score (1-100)')
plt.show()
```



[54]:
```
from mpl_toolkits.mplot3d import Axes3D

fig = plt.figure(figsize=(10, 8))
ax = fig.add_subplot(111, projection='3d')
scatter = ax.scatter(df['Age'], df['Annual Income (k$)'], df['Spending Score␣
↪(1-100)'], c=df['Cluster'], cmap='Set1')
ax.set_xlabel('Age')
ax.set_ylabel('Annual Income (k$)')
```

```
ax.set_zlabel('Spending Score (1-100)')
plt.title('3D Customer Segments')
plt.show()
```



[55]:
```
df.groupby('Cluster').mean()
```

[55]:
```
          CustomerID    Gender        Age  Annual Income (k$)  \
Cluster
0         159.743590  0.487179  39.871795           86.102564
1          83.872340  0.574468  55.638298           54.382979
2          55.648148  0.592593  25.185185           41.092593
3          24.100000  0.600000  46.250000           26.750000
4         161.025000  0.550000  32.875000           86.100000
```

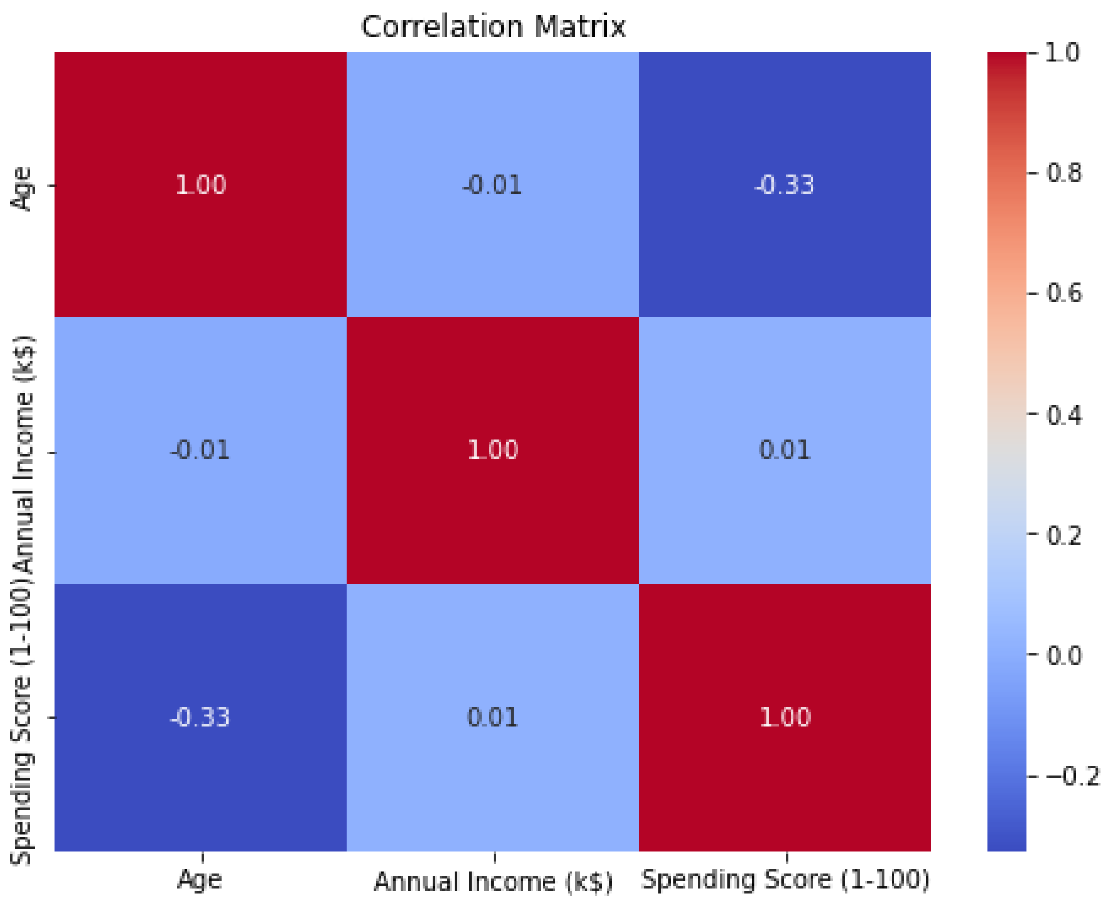
```
         Spending Score (1-100)
Cluster
0                     19.358974
1                     48.851064
2                     62.240741
3                     18.350000
4                     81.525000
```

Conclusion Cluster 1: Gender: Predominantly female (0.574) Age: Average age of 55.64 Annual Income: $54.38k Spending Score: 48.85 (Moderate spending behavior) Cluster 2: Gender: Predominantly female (0.593) Age: Average age of 25.19 Annual Income$:41.09k Spending Score: 62.24 (Moderate spending behavior)Cluster 3: Gender: Predominantly female (0.6) Age: Average age of 46.25 Annual Income: $26.75k (Lower income group) Spending Score: 18.35 (Low spending behavior) Cluster 4: Gender: Predominantly female (0.55) Age: Average age of 32.88 Annual Income$:86.10k (Higher income group) Spending Score: 81.53 (High spending behavior)Overall Conclusion: The customer segmentation shows diverse groups based on income and spending behavior:

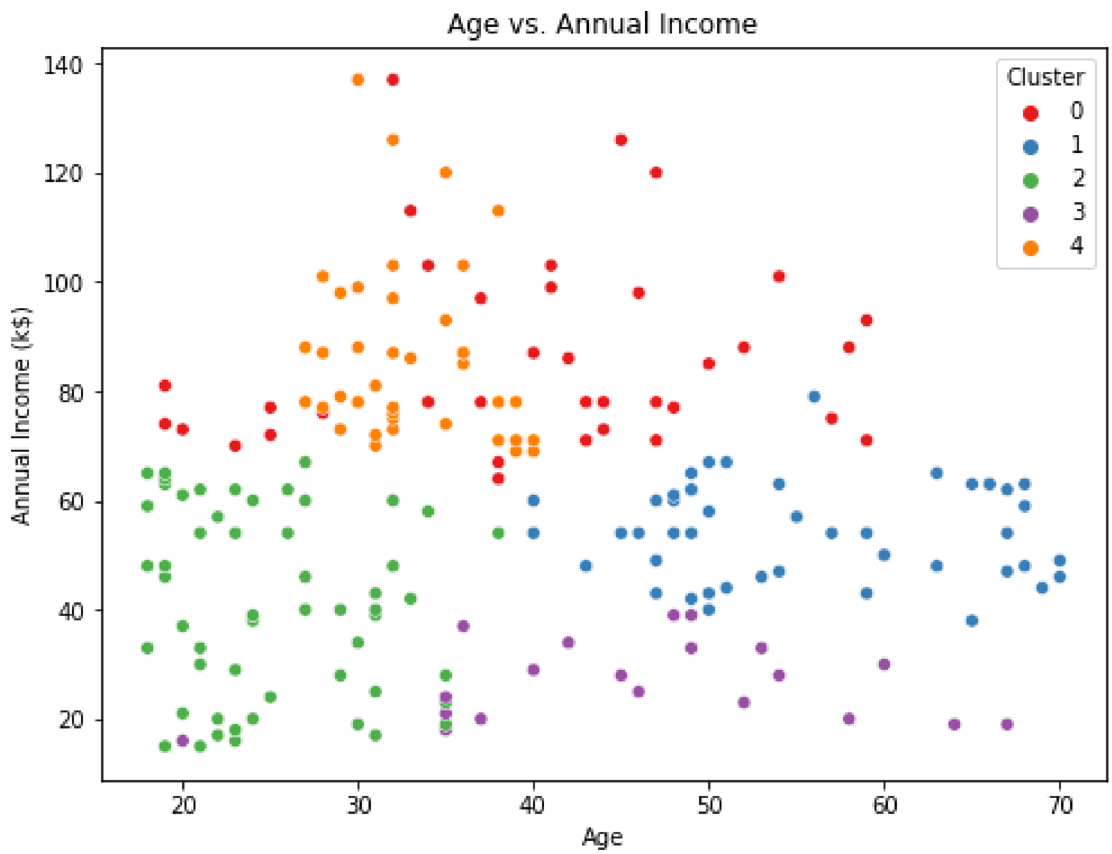
High-Income, Low-Spending: Cluster 0 Middle-Aged, Moderate-Spending: Cluster 1 Young, Moderate-Spending: Cluster 2 Older, Low-Spending, Low-Income: Cluster 3 Affluent, High-Spending: Cluster 4

[56]:

```python
# Correlation matrix
correlation_matrix = df[['Age', 'Annual Income (k$)', 'Spending Score␣
 ↪(1-100)']].corr()


plt.figure(figsize=(8, 6))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt='.2f',␣
 ↪cbar=True)
plt.title('Correlation Matrix')
plt.show()
```



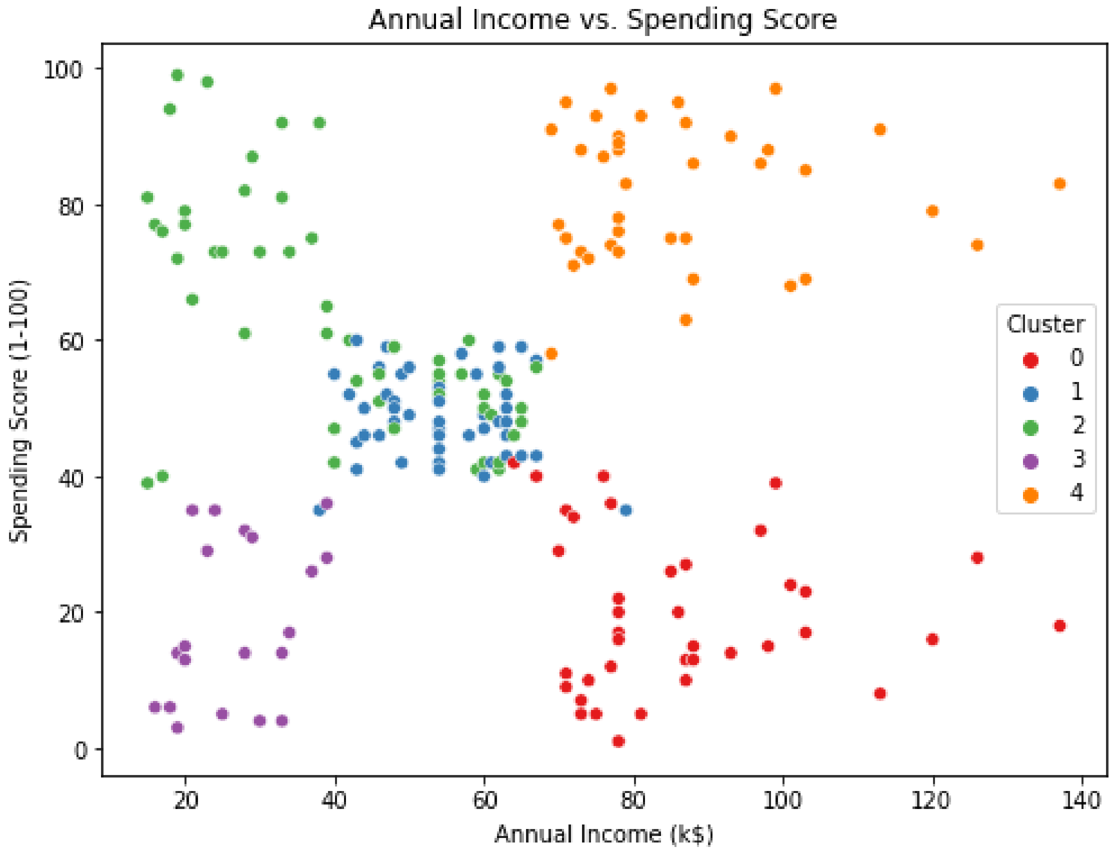
[57]:
```
plt.figure(figsize=(8, 6))
sns.scatterplot(x='Age', y='Annual Income (k$)', data=df, hue='Cluster',␣
↪palette='Set1')
plt.title('Age vs. Annual Income')
plt.show()
```



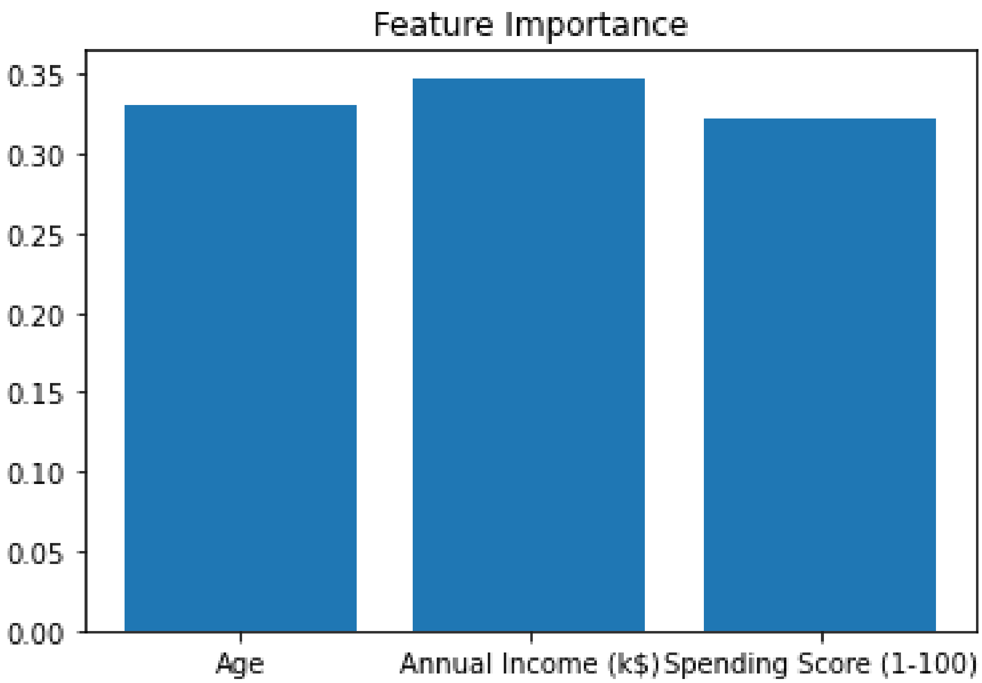
[58]:

```
plt.figure(figsize=(8, 6))
sns.scatterplot(x='Age', y='Spending Score (1-100)', data=df, hue='Cluster',␣
  ↪palette='Set1')
plt.title('Age vs. Spending Score')
plt.show()
```

[59]:
```
plt.figure(figsize=(8, 6))
sns.scatterplot(x='Annual Income (k$)', y='Spending Score (1-100)', data=df,␣
  ↪hue='Cluster', palette='Set1')
plt.title('Annual Income vs. Spending Score')
plt.show()
```



[60]:
```
from sklearn.ensemble import RandomForestClassifier

'
X = df[['Age', 'Annual Income (k$)', 'Spending Score (1-100)']]
y = df['Cluster']

# Fit Random Forest
rf = RandomForestClassifier(n_estimators=100, random_state=42)
rf.fit(X, y)

# Get feature importances
importances = rf.feature_importances_

# Plot feature importances
plt.bar(X.columns, importances)
plt.title('Feature Importance')
plt.show()
```



[61]:

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

[62]:
```
from sklearn.linear_model import LinearRegression

# Prepare data (features and target variable)
X = df[['Age', 'Annual Income (k$)']]  # Independent variables
y = df['Spending Score (1-100)']  # Target variable

# Fit the model
regressor = LinearRegression()
regressor.fit(X, y)

# Output coefficients and intercept
print(f"Coefficient for Age: {regressor.coef_[0]}")
print(f"Coefficient for Income: {regressor.coef_[1]}")
print(f"Intercept: {regressor.intercept_}")
```

```
Coefficient for Age: -0.6047872578754504
Coefficient for Income: 0.005748559223865298
Intercept: 73.34785222186397
```

Age is the more significant factor influencing Spending Score in this dataset. Younger people tend to spend more than older individuals. Income also affects spending, but its influence is very small compared to Age. Increasing income leads to a very modest increase in spending.

```
[ ]:
```