exp3

October 21, 2025

[1]:
```
# STEP 1: Upload dataset from local machine

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import silhouette_score

# Upload CSV file
#filename = "Mall_Customers.csv"

# Read the uploaded CSV
df = pd.read_csv("Mall_Customers.csv")
print("First 5 rows of dataset:")
print(df.head())
```

```
First 5 rows of dataset:
   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
```

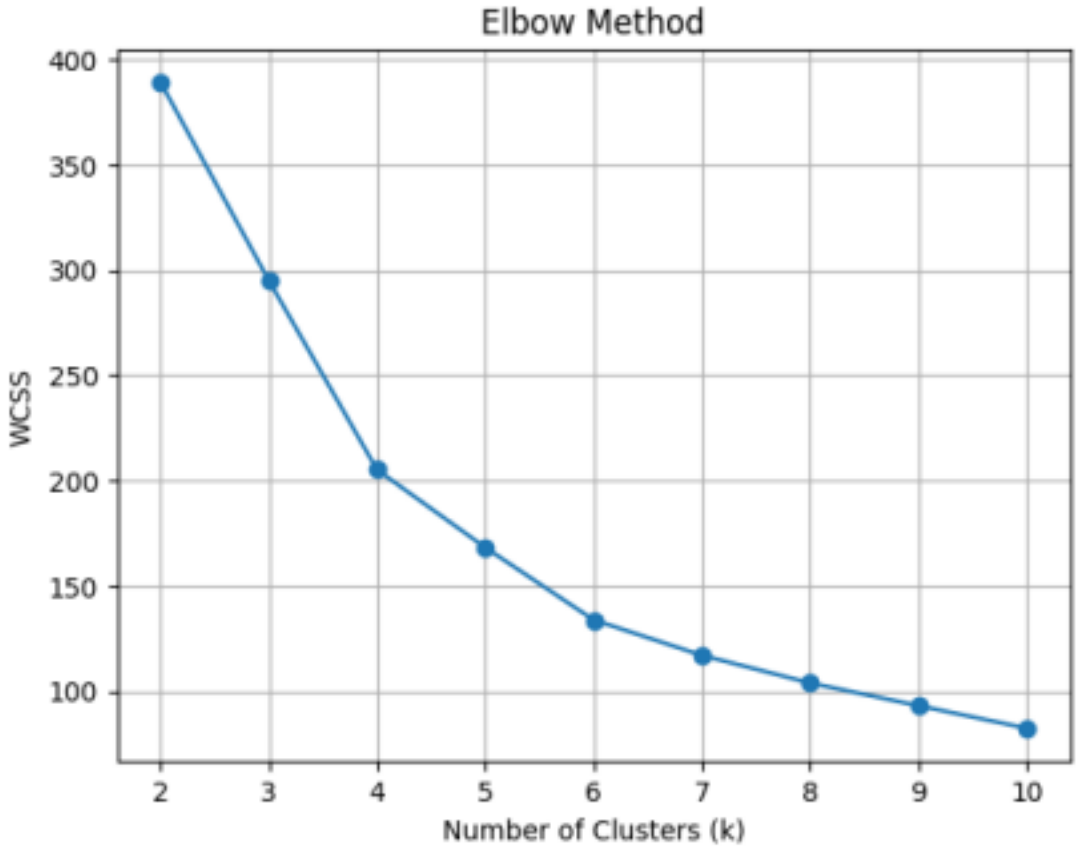
[2]:
```
# STEP 2: Select features for clustering

# Update column names here according to your dataset
features = df[['Age', 'Annual Income (k$)', 'Spending Score (1-100)']] # STEP 3:

Scale the data

scaler = StandardScaler()
scaled_features = scaler.fit_transform(features)
```

[3]:
```
# STEP 4: Determine optimal k (Elbow Method)

wcss = []
```



```
for k in range(2, 11):
    kmeans = KMeans(n_clusters=k, random_state=42, n_init=10)
    kmeans.fit(scaled_features)
    wcss.append(kmeans.inertia_)
```

```
# Plot the Elbow curve
plt.plot(range(2, 11), wcss, marker='o')
plt.xlabel('Number of Clusters (k)')
plt.ylabel('WCSS')
plt.title('Elbow Method')
plt.grid(True)
plt.show()
```



[4]:
```
# STEP 5: Check silhouette scores (optional)

for k in range(2, 11):
    kmeans = KMeans(n_clusters=k, random_state=42)
    labels = kmeans.fit_predict(scaled_features)
    score = silhouette_score(scaled_features, labels)

    print(f"k={k}, silhouette score={score:.3f}")
```



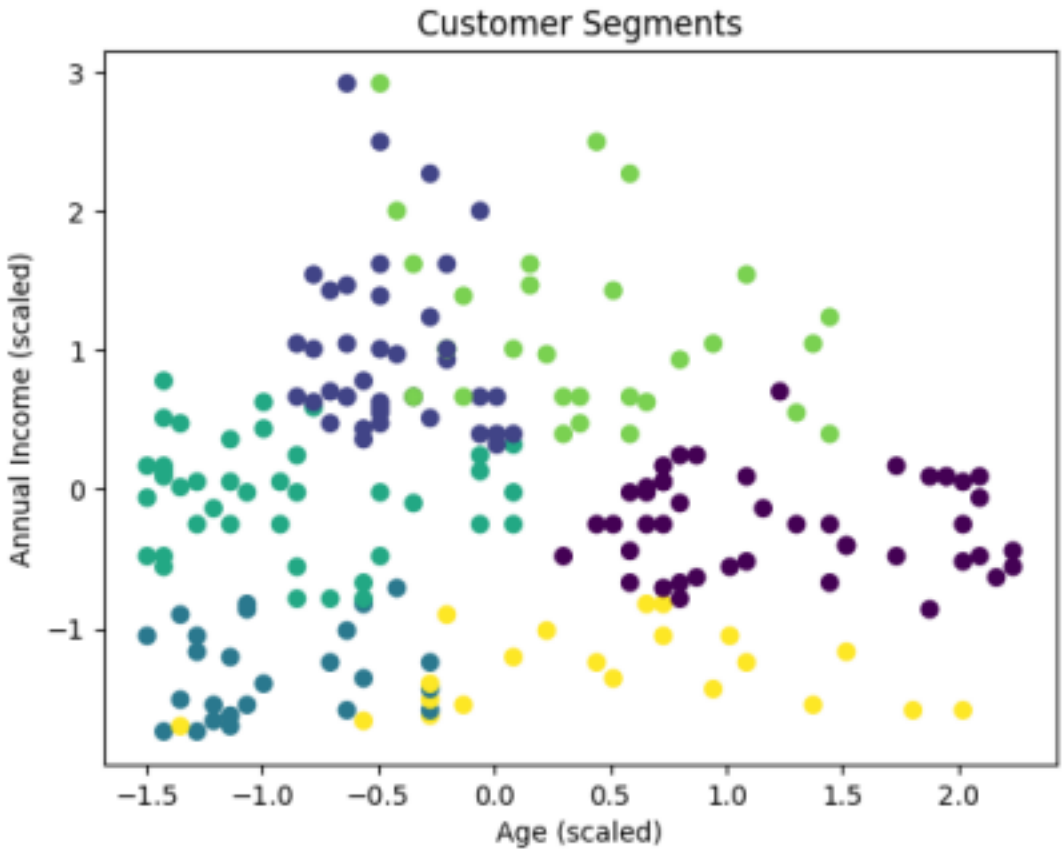
```
k=2, silhouette score=0.335
k=3, silhouette score=0.358
k=4, silhouette score=0.404
k=5, silhouette score=0.408
```

```
k=6, silhouette score=0.431
k=7, silhouette score=0.410
k=8, silhouette score=0.367
k=9, silhouette score=0.374
k=10, silhouette score=0.362
```

[5]:
```
# STEP 6: Fit K-Means with chosen k

optimal_k = 6 # Replace with value from elbow/silhouette analysis kmeans =
KMeans(n_clusters=optimal_k, random_state=42)
df['Cluster'] = kmeans.fit_predict(scaled_features)

# STEP 7: Analyze clusters

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

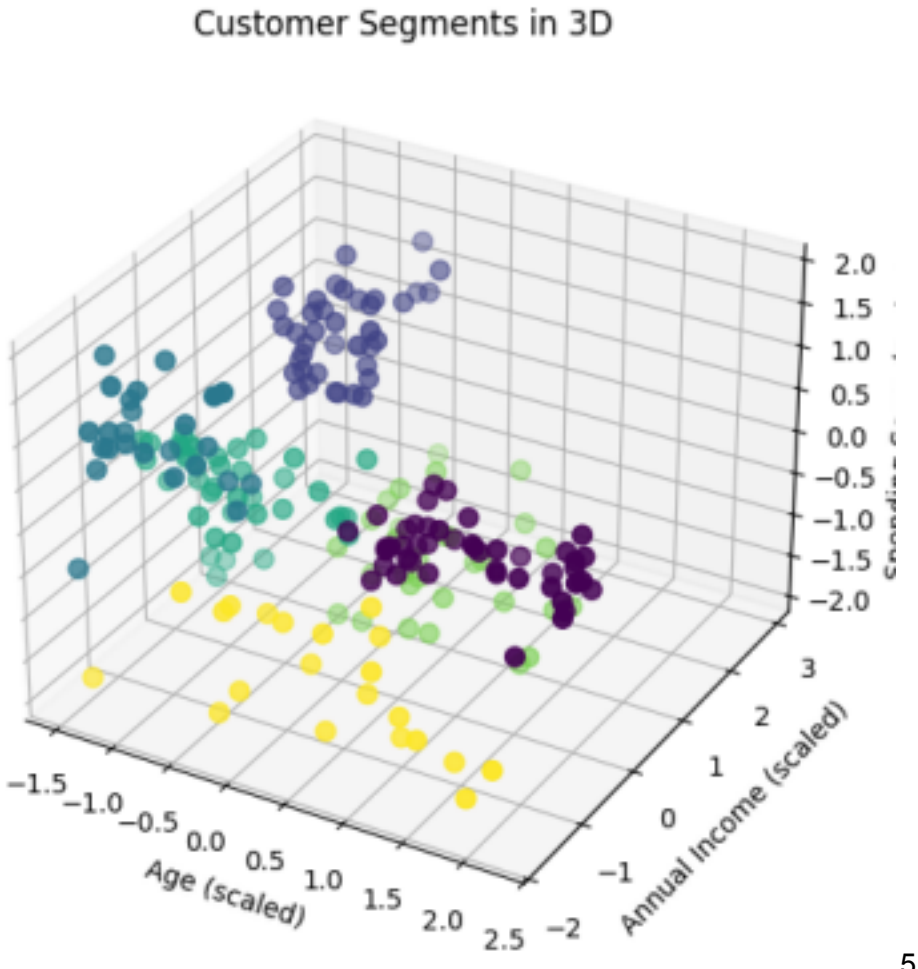
```
Cluster averages:
                Age Annual Income (k$) Spending Score (1-100)
Cluster
0 56.333333 54.266667 49.066667 1 32.692308 86.538462 82.128205 2
25.560000 26.480000 76.240000 3 26.125000 59.425000 44.450000 4
44.000000 90.133333 17.933333 5 45.523810 26.285714 19.380952
```

[6]:
```
# STEP 8: Visualize clusters (2D example)

plt.scatter(scaled_features[:, 0], scaled_features[:, 1], c=df['Cluster'],␣
↪cmap='viridis')
plt.xlabel('Age (scaled)')
plt.ylabel('Annual Income (scaled)')
plt.title('Customer Segments')
plt.show()
```



[7]:
```
# STEP 9: Optional 3D visualization
fig = plt.figure(figsize=(8, 6))
ax = fig.add_subplot(111, projection='3d')
  ax.scatter(scaled_features[:, 0], scaled_features[:, 1], scaled_features[:, 2], c=df['Cluster'],
                                        cmap='viridis', s=50)
ax.set_xlabel('Age (scaled)')
ax.set_ylabel('Annual Income (scaled)')
ax.set_zlabel('Spending Score (scaled)')
ax.set_title('Customer Segments in 3D')
plt.show()
```

[ ]: [ ]:


Customer Segments in 3D
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
3
2
1
0
−1
−2
Annual Income (scaled)
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
2.5
Age (scaled)