

## Goal: Customer Segmentation / Clustering 🧑🧑

The goal is to segment customers into distinct groups based on both their profile information and transaction behavior using clustering techniques. The result will help identify patterns in customer behavior, which can be useful for personalized marketing, targeting high-value customers, and improving customer retention.

---

### Customer Segmentation Steps:

#### 1. Data Preprocessing 📊:

- Merge customer and transaction data.
- Extract key features like total spending, purchase quantity, and average product price.
- Normalize numerical features and one-hot encode categorical ones (e.g., Region).

#### 2. PCA (Dimensionality Reduction) 🔍:

- Use PCA to reduce features to 2 components for easy visualization.

#### 3. Elbow Method 📈:

- Determine the optimal number of clusters by calculating inertia and plotting the elbow curve.

#### 4. K-Means Clustering 🏠:

- Apply K-Means clustering with the chosen number of clusters (e.g., 4).

#### 5. Evaluation 📊:

- Evaluate clusters using the Davies-Bouldin Index and Silhouette Score.

### ✓ Importing Libraries

```
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
from sklearn.metrics import davies_bouldin_score, silhouette_score
import matplotlib.pyplot as plt
import seaborn as sns
```

```
# Load the datasets
customers = pd.read_csv('/content/drive/MyDrive/Zeotap/Customers.csv')
transactions = pd.read_csv('/content/drive/MyDrive/Zeotap/Transactions.csv')
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
df = pd.merge(transactions, customers, on='CustomerID', how='inner')
```

```
df
```



	TransactionID	CustomerID	ProductID	TransactionDate	Quantity	TotalValue	Price
<b>0</b>	T00001	C0199	P067	2024-08-25 12:38:23	1	300.68	300.68
<b>1</b>	T00112	C0146	P067	2024-05-27 22:23:54	1	300.68	300.68
<b>2</b>	T00166	C0127	P067	2024-04-25 07:38:55	1	300.68	300.68
<b>3</b>	T00272	C0087	P067	2024-03-26 22:55:37	2	601.36	300.68
<b>4</b>	T00363	C0070	P067	2024-03-21 15:10:10	3	902.04	300.68
...	...	...	...	...	...	...	...
<b>995</b>	T00496	C0118	P037	2024-10-24 08:30:27	1	459.86	459.86
<b>996</b>	T00759	C0059	P037	2024-06-04 02:15:24	3	1379.58	459.86

Next steps: [Generate code with df](#) [View recommended plots](#) [New interactive sheet](#)

## Feature Engineering

```
customer_features = df.groupby('CustomerID').agg({
    'TotalValue': 'sum',
    'Quantity': 'sum',
    'Price': 'mean'
}).reset_index()
```

```
customer_features = pd.merge(customer_features, customers[['CustomerID', 'Region']], on='Cus
```

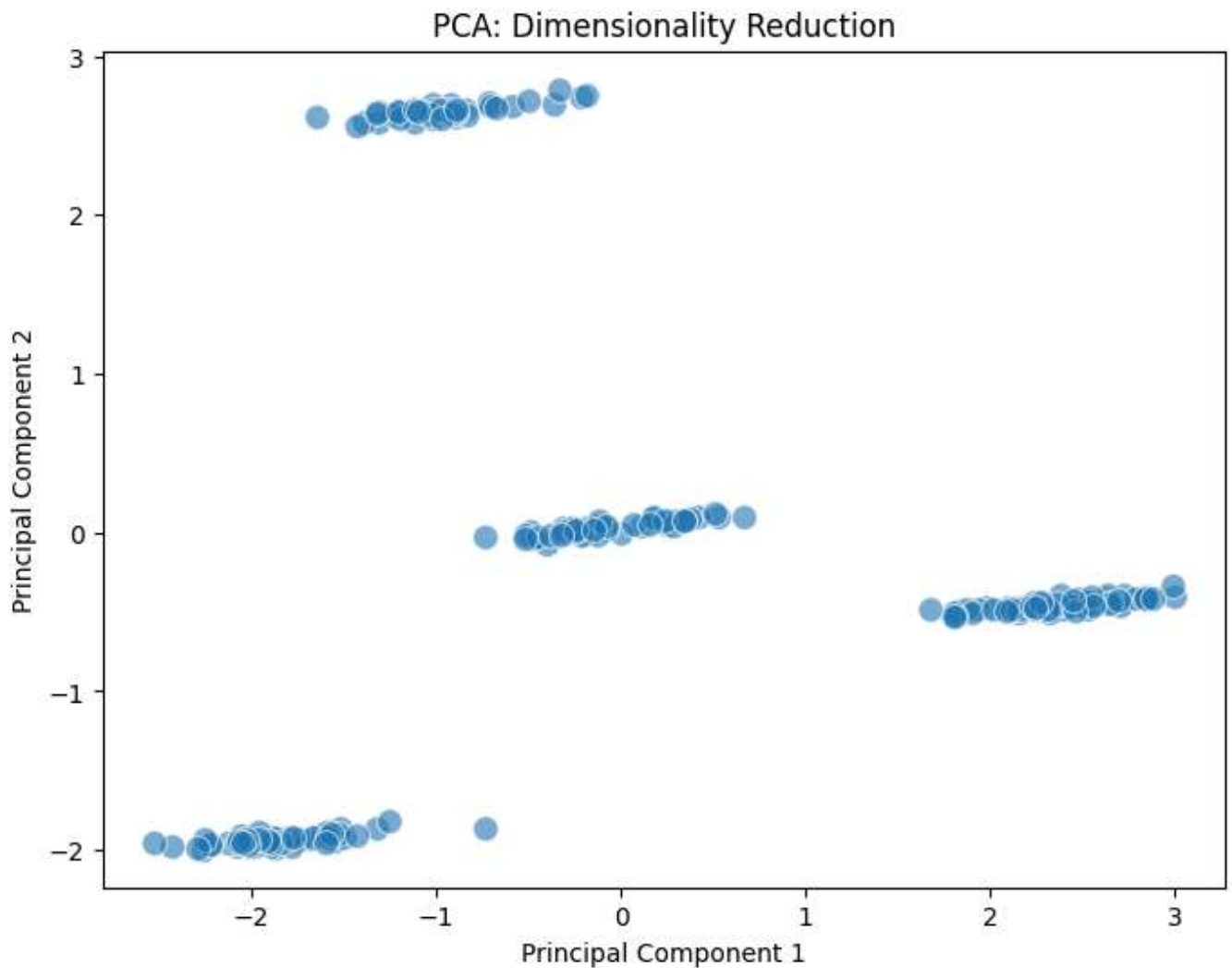
```
# Region using one-hot encoding
customer_features = pd.get_dummies(customer_features, columns=['Region'], drop_first=True)

scaler = StandardScaler()
scaled_features = scaler.fit_transform(customer_features.drop('CustomerID', axis=1))
```

## ✓ PCA for Dimensionality Reduction

```
pca = PCA(n_components=2)
pca_result = pca.fit_transform(scaled_features)

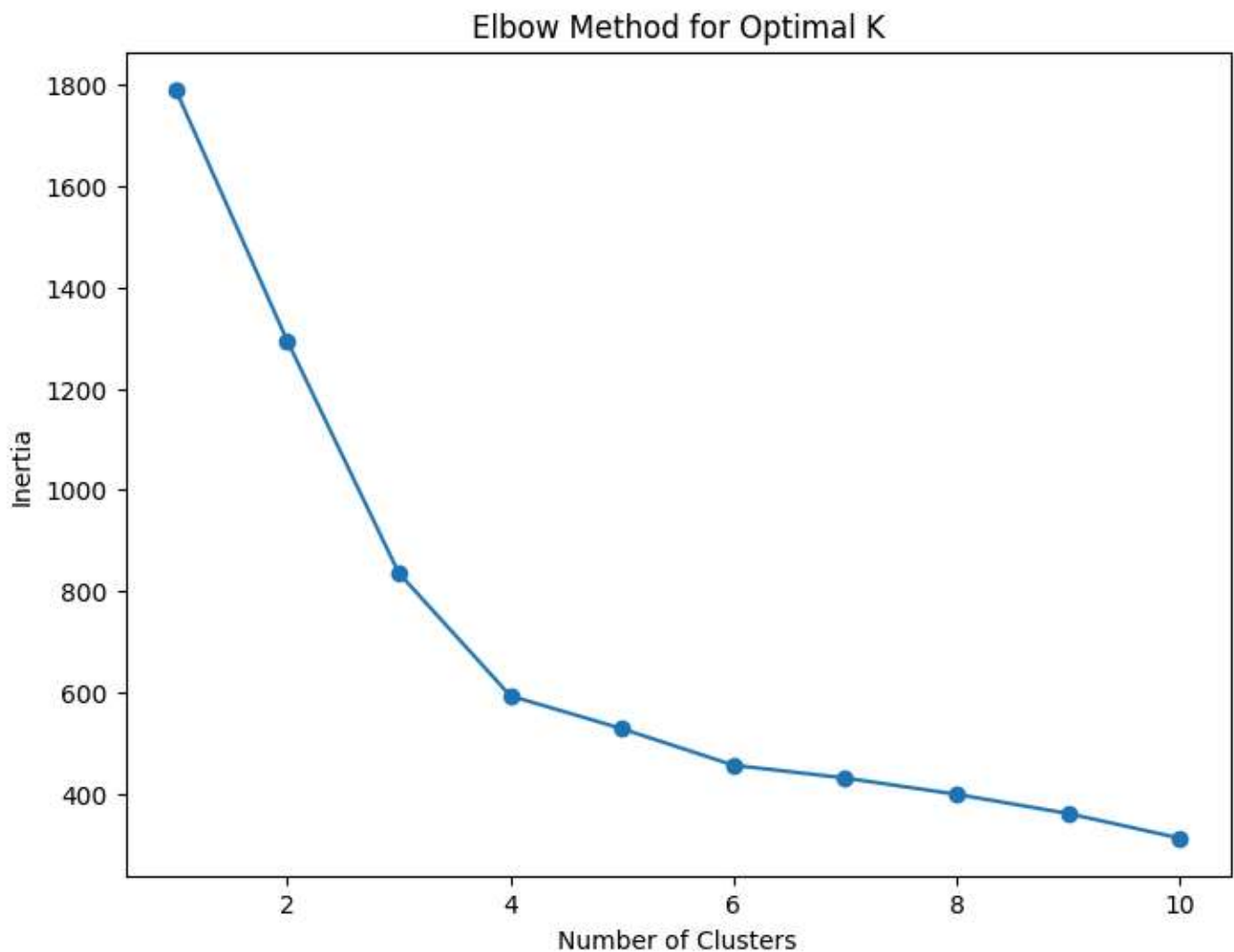
plt.figure(figsize=(8, 6))
sns.scatterplot(x=pca_result[:, 0], y=pca_result[:, 1], s=100, alpha=0.6)
plt.title('PCA: Dimensionality Reduction')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.show()
```



## ✓ Elbow Method to Determine the Optimal Number of Clusters

```
inertia = []  
range_n_clusters = list(range(1, 11))  
  
for n_clusters in range_n_clusters:  
    kmeans = KMeans(n_clusters=n_clusters, random_state=42)  
    kmeans.fit(scaled_features)  
    inertia.append(kmeans.inertia_)
```

```
plt.figure(figsize=(8, 6))  
plt.plot(range_n_clusters, inertia, marker='o')  
plt.title('Elbow Method for Optimal K')  
plt.xlabel('Number of Clusters')  
plt.ylabel('Inertia')  
plt.show()
```

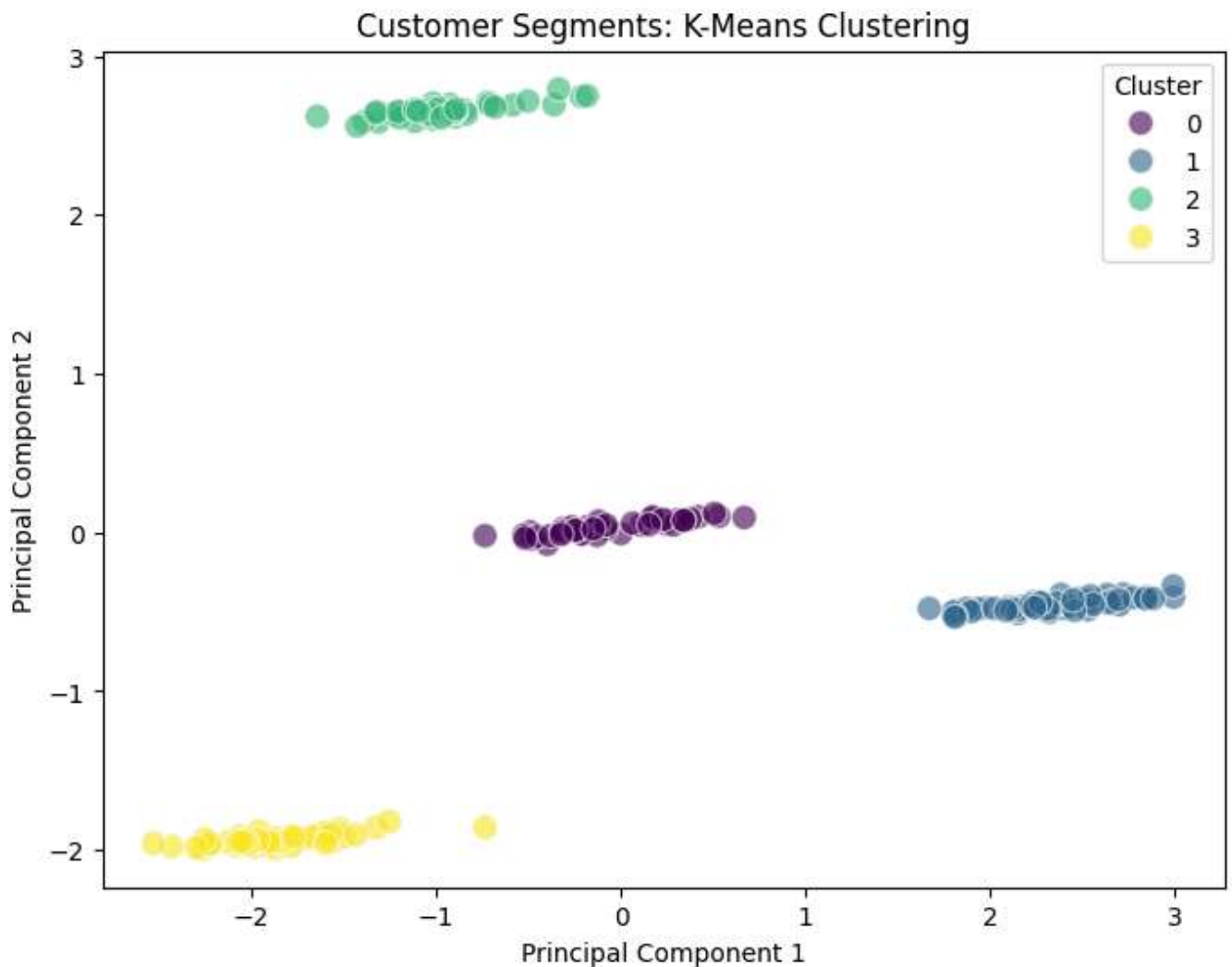


## ✓ K-Means Clustering

```
kmeans = KMeans(n_clusters=4, random_state=42)
customer_features['Cluster'] = kmeans.fit_predict(scaled_features)


pca_result = pca.fit_transform(scaled_features)

plt.figure(figsize=(8, 6))
sns.scatterplot(x=pca_result[:, 0], y=pca_result[:, 1], hue=customer_features['Cluster'], palette='magma')
plt.title('Customer Segments: K-Means Clustering')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.show()
```



## ✓ Clustering Evaluation

```
db_index = davies_bouldin_score(scaled_features, customer_features['Cluster'])  
print(f'Davies-Bouldin Index: {db_index}')
```

 Davies-Bouldin Index: 0.9568549999422535

Start coding or [generate](#) with AI.