Task 3 :

For **Task 3: Customer Segmentation**, the evaluation criteria are **Clustering Logic and Metrics** and **Visual Representation of Clusters**. Below, I'll break down how to address these criteria and ensure your clustering model meets the expectations.

---

1. Clustering Logic and Metrics

What is Being Evaluated?

- The logic behind the clustering algorithm and the choice of features.
- The quality of the clusters, measured using clustering metrics such as the **Davies-Bouldin Index (DB Index)**.

How to Ensure Good Clustering Logic and Metrics:

1. **Feature Engineering**:
    - Use meaningful features derived from customer profiles and transaction history. For example:
        - **Total Spending**: Total amount spent by the customer.
        - **Average Order Value**: Average amount spent per transaction.
        - **Favorite Category**: The most frequently purchased product category.
        - **Region**: Geographic location of the customer.
        - **Signup Date**: How long the customer has been active.
    - Ensure that the features are normalized or standardized to avoid bias due to different scales.
2. **Clustering Algorithm**:
    - Use a clustering algorithm such as **K-Means**, **DBSCAN**, or **Agglomerative Clustering**.
    - **K-Means** is a good choice for this task because it is simple, scalable, and works well with numerical data.
    - Experiment with different numbers of clusters (between 2 and 10) and choose the optimal number based on clustering metrics.
3. **Clustering Metrics**:
    - Use the **Davies-Bouldin Index (DB Index)** to evaluate the quality of the clusters. The DB Index measures the average similarity ratio of each cluster with the cluster that is most similar to it. A lower DB Index indicates better clustering.
    - Other metrics you can use include:

        - **Silhouette Score**: Measures how similar an object is to its own cluster compared to other clusters.
        - **Calinski-Harabasz Index**: Measures the ratio of between-cluster dispersion to within-cluster dispersion.

4. **Optimal Number of Clusters**:
    - Use the **Elbow Method** or **Silhouette Analysis** to determine the optimal number of clusters.
    - The Elbow Method involves plotting the within-cluster sum of squares (WCSS) against the number of clusters and looking for an "elbow" point where the rate of decrease slows down.

---

2. Visual Representation of Clusters

What is Being Evaluated?

- The ability to visualize the clusters in a meaningful way.
- The clarity and interpretability of the visualizations.

How to Ensure Good Visual Representation:

1. **Dimensionality Reduction**:
    - Use **Principal Component Analysis (PCA)** or **t-SNE** to reduce the dimensionality of the data for visualization.
    - PCA is a linear technique that projects the data onto a lower-dimensional space while preserving as much variance as possible.
    - t-SNE is a non-linear technique that is particularly good for visualizing high-dimensional data in 2D or 3D.
2. **Cluster Visualization**:
    - Plot the clusters in 2D or 3D space using the reduced dimensions.
    - Use different colors to represent different clusters.
    - Add labels and legends to make the plot easy to interpret.
3. **Cluster Profiling**:
    - Create a profile for each cluster by analyzing the mean or median values of the features within each cluster.
    - This will help you understand the characteristics of each cluster and provide actionable insights.

[20]:
```python
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
from sklearn.metrics import davies_bouldin_score
# Use the same features as in Task 2
X = scaled_features
# Perform K-Means clustering
kmeans = KMeans(n_clusters=4, random_state=42)
customer_features['Cluster'] = kmeans.fit_predict(X)
print(f'K-Means:{kmeans}')
```

```
C:\Users\badam\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:
1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
C:\Users\badam\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:
there are less chunks than available threads. You can avoid it by sett
  warnings.warn(
K-Means:KMeans(n_clusters=4, random_state=42)
```

[21]:
```python
# Calculate DB Index
db_index = davies_bouldin_score(X, customer_features['Cluster'])
print(f'Davies-Bouldin Index: {db_index}')
```

```
Davies-Bouldin Index: 1.6838213239629722
```

[22]:

```python
# Visualize clusters using PCA
pca = PCA(n_components=2)
pca_result = pca.fit_transform(X)
customer_features['PCA1'] = pca_result[:, 0]
customer_features['PCA2'] = pca_result[:, 1]
plt.figure(figsize=(10, 6))
sns.scatterplot(x='PCA1', y='PCA2', hue='Cluster', data=customer_features, palette='viridis')
plt.title('Customer Segmentation using K-Means Clustering')
plt.show()

# Save clustering results
customer_features.to_csv('Clustering_Results.csv', index=False)
```