Experiment No. 9

Title:

Clustering Data Using EM Algorithm and K-Means — Comparison and Analysis

Lab Objective:

- Apply **Expectation-Maximization (EM) algorithm** (using Gaussian Mixture Models) to cluster data from a CSV file.
- Apply **k-Means clustering** on the same dataset.
- Compare clustering results quantitatively and visually.
- Comment on the quality and differences of the clusters produced by both algorithms.

Python Setup

Tools Required:

- Python 3.x
- Libraries: `numpy`, `pandas`, `scikit-learn`, `matplotlib`, `seaborn`

Step 1: Import Libraries

```
import numpy as np
import pandas as pd
from sklearn.mixture import GaussianMixture
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
import matplotlib.pyplot as plt
import seaborn as sns
```

Step 2: Load Dataset

```
# Load your CSV file here (update file path accordingly)
data = pd.read_csv('your_dataset.csv')

# Preview the data
print(data.head())

# Optional: Select relevant columns/features if dataset has non-numeric columns
X = data.values  # Or select specific columns: data[['feature1', 'feature2']].values
```

Step 3: Apply EM Algorithm (Gaussian Mixture Model)

```
# Define number of clusters
n_clusters = 3  # Change based on your dataset knowledge or use methods like BIC/AIC for
selection

gmm = GaussianMixture(n_components=n_clusters, covariance_type='full',
random_state=42)
gmm_labels = gmm.fit_predict(X)
```

Step 4: Apply K-Means Algorithm

```
kmeans = KMeans(n_clusters=n_clusters, random_state=42)
kmeans_labels = kmeans.fit_predict(X)
```

Step 5: Evaluate and Compare Clustering

```
# Silhouette Score (higher is better, ranges from -1 to 1)
silhouette_gmm = silhouette_score(X, gmm_labels)
silhouette_kmeans = silhouette_score(X, kmeans_labels)

print(f'Silhouette Score for GMM (EM): {silhouette_gmm:.3f}')
print(f'Silhouette Score for K-Means: {silhouette_kmeans:.3f}')
```

Step 6: Visualize Clustering Results

For 2D or 3D data, visualize clusters:

```
def plot_clusters(X, labels, title):
    plt.figure(figsize=(8,6))
    sns.scatterplot(x=X[:, 0], y=X[:, 1], hue=labels, palette='viridis', s=50)
    plt.title(title)
    plt.show()

# For 2D data visualization
plot_clusters(X, gmm_labels, "GMM (EM) Clustering")
plot_clusters(X, kmeans_labels, "K-Means Clustering")
```

Step 7: Comment on the Results

- **Silhouette Scores**: Indicate cluster cohesion and separation.
- **Cluster Shapes**:
    - EM (GMM) can model **elliptical** clusters using covariance matrices.
    - K-Means assumes **spherical** clusters of similar size.
- **Interpretation**:
    - If silhouette score of GMM is higher → better separation.
    - Visual plots show how clusters differ in shape and overlap.

<u>Reflection Questions:</u>

- For high-dimensional data, consider PCA before clustering for visualization.
- Experiment with different numbers of clusters and select optimal using BIC/AIC (for GMM) or elbow method (for k-Means).