Exercise 6 - 23011101105 January 29, 2025

GMM Clustering

Aim: To perform GMM clustering on two datasets

GMM Clustering:

Gaussian Mixture Model (GMM) clustering is an unsupervised learning technique that groups data points into clusters based on a mixture of Gaussian distributions. Each cluster is represented by a Gaussian distribution with its own mean and covariance. GMM clustering is done using the Expectation-Maximization (EM) Algorithm, which involves the following steps:

1. Initialization: Choose the number of clusters (k) and initialize the mean and covariance parameters.
2. Expectation (E) Step: Assign each data point to a cluster based on the current parameters and their probability of belonging to each cluster.
3. Maximization (M) Step: Update the parameters (mean and covariance) based on the current cluster assignments.
4. Repeat: Alternate between the E and M steps until the parameters converge.

Algorithm:

1. Load the Titanic dataset.
2. Select relevant features for clustering.
3. One-hot encode categorical variables (e.g., Sex, Embarked).
4. Impute missing numerical values (e.g., Age).
5. Standardize features using StandardScaler.
6. Apply GMM clustering with a specified number of clusters (e.g., 3).
7. Get cluster labels and Gaussian means (centroids).
8. Apply PCA to reduce data to 2D.
9. Visualize clusters with a scatter plot, colored by labels.
10. Mark cluster centers on the plot.

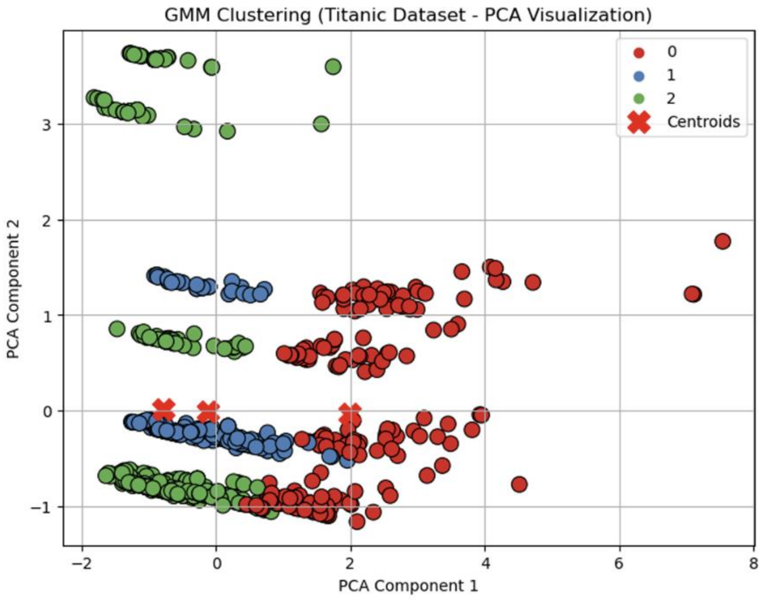
Code:

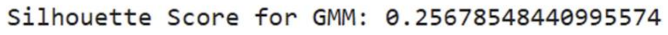
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.mixture import GaussianMixture
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler
from sklearn.impute import SimpleImputer
from sklearn.model_selection import train_test_split
import seaborn as sns
```

```
from sklearn.metrics import silhouette_score

# Load Titanic dataset
titanic_data = pd.read_csv('Titanic-Dataset.csv')
titanic_data.head()

# Select relevant features for clustering
data = titanic_data[['Pclass', 'Sex', 'Age', 'Fare',
'Embarked']]

# One-hot encode categorical variables
data = pd.get_dummies(data, drop_first=True)

# Handle missing values (imputation)
imputer = SimpleImputer(strategy='mean')  # Using mean for
numeric, and mode for categorical
data['Age'] = imputer.fit_transform(data[['Age']])
data['Embarked_Q'] = data['Embarked_Q'].fillna(0)  # Filling
missing Embarked data

# Standardize features
scaler = StandardScaler()
data_scaled = scaler.fit_transform(data)

# Apply GMM clustering
gmm = GaussianMixture(n_components=3, random_state=42)
gmm.fit(data_scaled)
labels = gmm.predict(data_scaled)
means = gmm.means_

# Apply PCA for 2D visualization
pca = PCA(n_components=2)
data_pca = pca.fit_transform(data_scaled)

# Visualize the clusters in 2D
plt.figure(figsize=(8, 6))
sns.scatterplot(x=data_pca[:, 0], y=data_pca[:, 1],
hue=labels, palette='Set1', s=100, marker='o',
edgecolor='black')

# Mark the cluster centers (centroids)
pca_centroids = pca.transform(means)
plt.scatter(pca_centroids[:, 0], pca_centroids[:, 1], s=200,
c='red', marker='X', label='Centroids')

# Add plot details
```

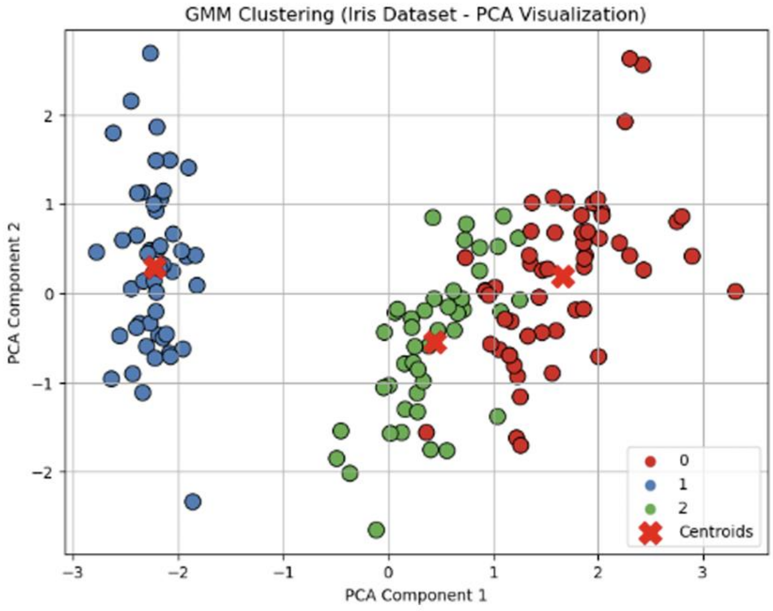
```
plt.title('GMM Clustering (Titanic Dataset - PCA
Visualization)')
plt.xlabel('PCA Component 1')
plt.ylabel('PCA Component 2')
plt.legend()
plt.grid(True)
plt.show()

# Calculate and print silhouette score
sil_score_gmm = silhouette_score(data_scaled, labels)
print(f"Silhouette Score for GMM: {sil_score_gmm}")
```

Output:





```
Silhouette Score for GMM: 0.25678548440995574
```


GMM Clustering (Iris Dataset - PCA Visualization)
PCA Component 1
PCA Component 2
● 0
● 1
● 2
✖ Centroids