import time

from sklearn.cluster import KMeans as SklearnKMeans

from sklearn\_extra.cluster import KMedoids as SklearnKMedoids

from sklearn.metrics import pairwise\_distances

# Function to measure execution time

def measure\_time\_sklearn(algorithm, X, n\_clusters):

start\_time = time.time()

model = algorithm(n\_clusters=n\_clusters, random\_state=42).fit(X)

end\_time = time.time()

return end\_time - start\_time, model

# Sklearn K-means

kmeans\_time\_sklearn, kmeans\_model\_sklearn = measure\_time\_sklearn(SklearnKMeans, X, 3)

kmeans\_labels\_sklearn = kmeans\_model\_sklearn.labels\_

kmeans\_sse\_sklearn = kmeans\_model\_sklearn.inertia\_

kmeans\_iterations\_sklearn = kmeans\_model\_sklearn.n\_iter\_

# Sklearn K-medoids

kmedoids\_time\_sklearn, kmedoids\_model\_sklearn = measure\_time\_sklearn(SklearnKMedoids, X, 3)

kmedoids\_labels\_sklearn = kmedoids\_model\_sklearn.labels\_

kmedoids\_sse\_sklearn = sum(np.min(pairwise\_distances(X, kmedoids\_model\_sklearn.cluster\_centers\_), axis=1) \*\* 2)

kmedoids\_iterations\_sklearn = kmedoids\_model\_sklearn.n\_iter\_

# Calculate cluster sizes for sklearn K-means

kmeans\_cluster\_info\_sklearn = cluster\_analysis(kmeans\_labels\_sklearn, kmeans\_model\_sklearn.cluster\_centers\_, X)

# Calculate cluster sizes for sklearn K-medoids

kmedoids\_cluster\_info\_sklearn = cluster\_analysis(kmedoids\_labels\_sklearn, kmedoids\_model\_sklearn.cluster\_centers\_, X)

# Print results for sklearn K-means

print("Sklearn K-means Clustering Results:")

print("Iterations:", kmeans\_iterations\_sklearn)

print("Cluster Info:", kmeans\_cluster\_info\_sklearn)

print("Overall SSE:", kmeans\_sse\_sklearn)

print("Time Complexity:", kmeans\_time\_sklearn, "seconds")

# Print results for sklearn K-medoids

print("\nSklearn K-medoids Clustering Results:")

print("Iterations:", kmedoids\_iterations\_sklearn)

print("Cluster Info:", kmedoids\_cluster\_info\_sklearn)

print("Overall SSE:", kmedoids\_sse\_sklearn)

print("Time Complexity:", kmedoids\_time\_sklearn, "seconds")

# Comparative Analysis

print("\nComparative Analysis:")

print("1. Iterations: The number of iterations may vary due to differences in implementation details.")

print("2. Cluster Sizes: Both custom and sklearn implementations should produce similar cluster sizes.")

print("3. SSE/WCSS Values: The SSE values should be comparable if the implementations are correct.")

print("4. Time Complexity: Sklearn implementations are generally optimized and may run faster.")

print("5. Alignment: If results align closely, it indicates correctness. Differences may arise from optimization techniques in sklearn.")

Output:



