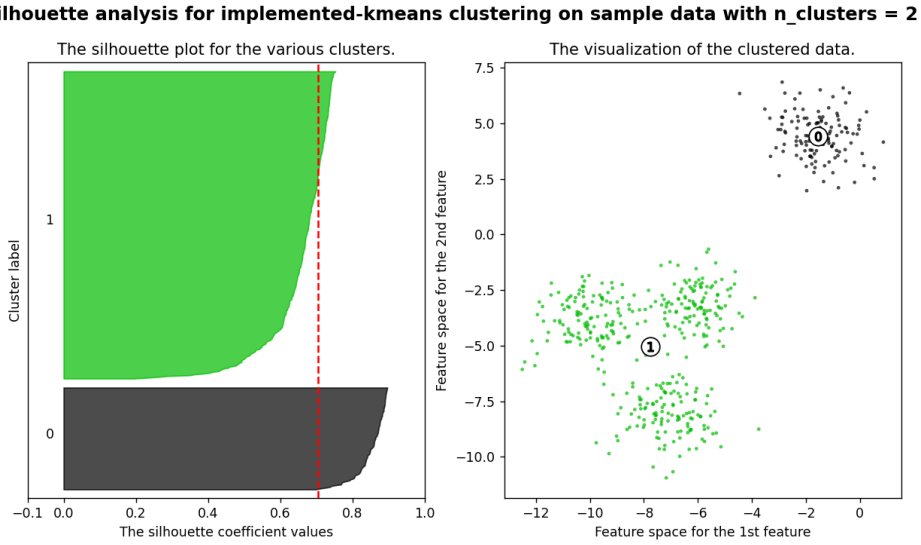
WORKSHOP II SOLUTION - POINT 5

What number of K got the best silhouette score? What can you say about the figures? Is this the expected result?

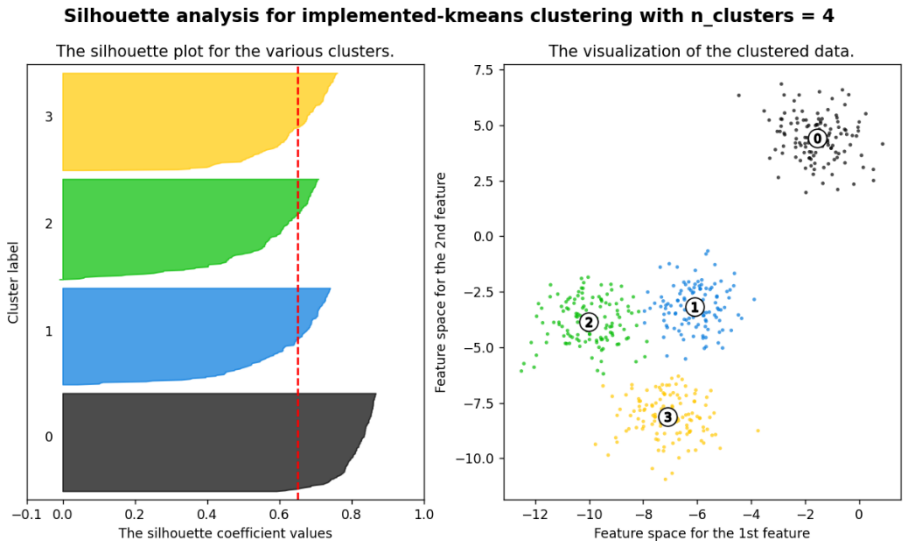
KMEANS

```
Silhouette analysis using implemeted algorithm KMeans
------------------------------------------------------------
For n_clusters = 2 The average silhouette_score is: 0.7049787496083262
For n_clusters = 3 The average silhouette_score is: 0.5882004012129721
For n_clusters = 4 The average silhouette_score is: 0.6505186632729437
For n_clusters = 5 The average silhouette_score is: 0.5804620679044765
```

The best silhouette score was obtained with k = 2, but analyzing the graphs, an unbalanced amplitude in the size of the silhouette graphs was identified:



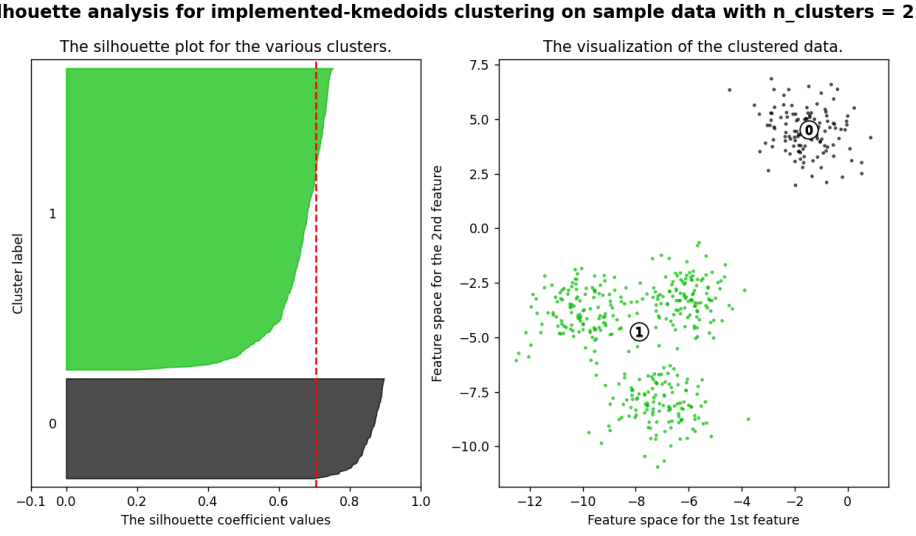
Discarding the k = 2 score, the next best score was obtained with k = 4. This appears to be an optimal selection because the silhouette score of each cluster is above the average silhouette scores. In addition, the fluctuation in the size of the clusters is similar and corresponds with the expected result:

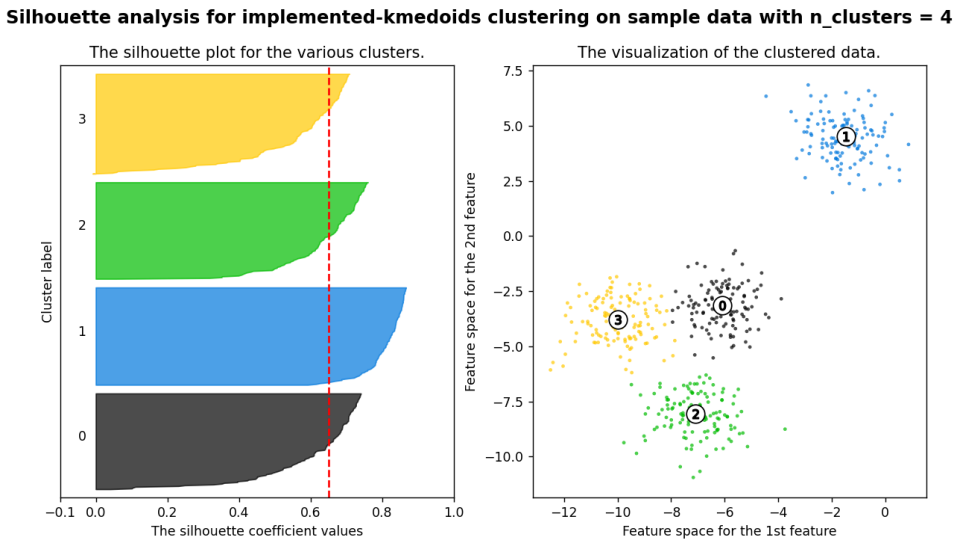


KMEDOIDS

The results obtained in the silhouette analysis using the implemented KMedoids algorithm are similar to the results obtained using the implemented KMeans algorithm. The results and graphs obtained are presented below:

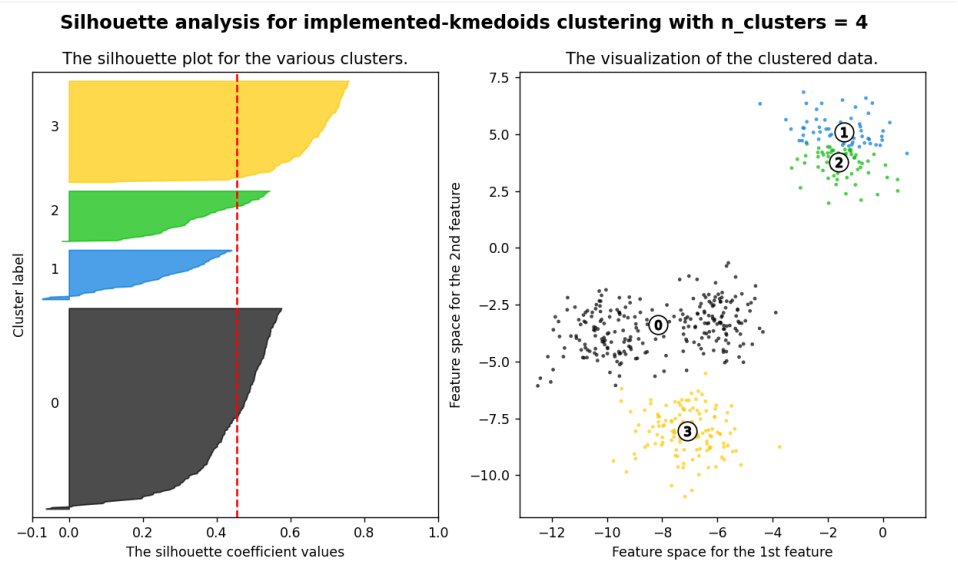
```
Silhouette analysis using implemeted algorithm KMedoids
-------------------------------------------------------------
For n_clusters = 2 The average silhouette_score is: 0.7049787496083262
For n_clusters = 3 The average silhouette_score is: 0.5873430979447513
For n_clusters = 4 The average silhouette_score is: 0.6505186632729437
For n_clusters = 5 The average silhouette_score is: 0.5707552126966944
```

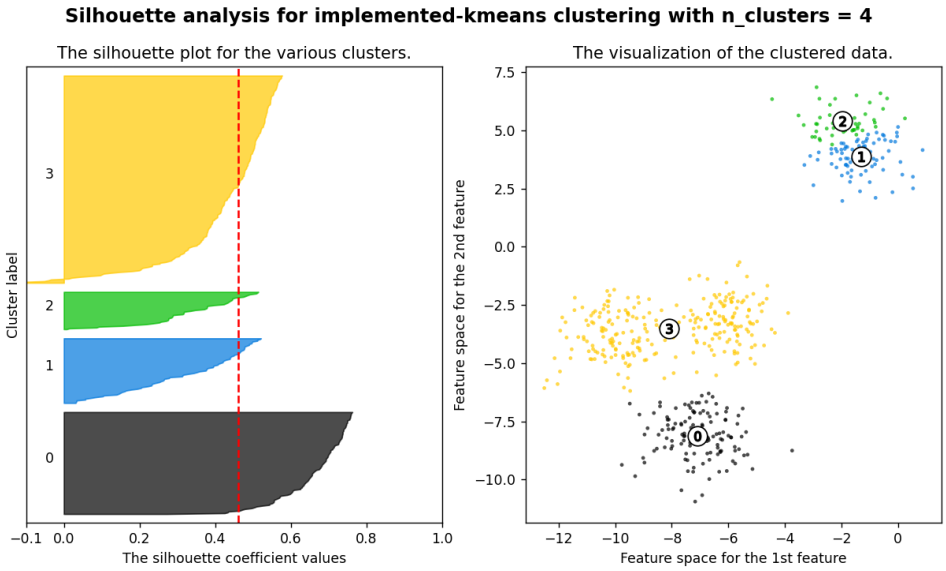




Remark

Running the script several times, it is observed that the results vary in each run due to the random initialization of the cluster centroids. In some runs, the implemented algorithms fail to cluster the data as expected. The result for k = 4 is as follows:





As shown in the graph on the right, the algorithms cluster two different but close clusters and separate one cluster into two groups.

To avoid this result, the algorithms were initialized to obtain the same results in each run using the *random_state* parameter.