REPORT

Lab Assignment 8

Vrushali Pandit - B20BB047

Data:

Details about chemical composition of milk obtained from 25 different animals. Protein, water, fat, lactose and ash content. Similar animals such as terrestrial would have similar features.

Shape = (25,5)

Task 1:

Performing k means clustering for different values of k and plotting the clusters + cluster centres

Pre-processing:

1. We need to standardize the data. This is because (like K-nearest neighbours) K means is a distance based algorithm.
2. Called *StandardScaler()* from *sklearn.preprocessing* and fit it on the data.
3. This ensured that there is no bias amongst the features.

Applying k-means:

1. Called *KMeans* from *sklearn.clusters* python library
2. For values of k = {3,5,7} , applied K Means with random_State = 2021, to ensure reproducible results.
3. Plotting the first two features as it is not possible to visualize 5D space

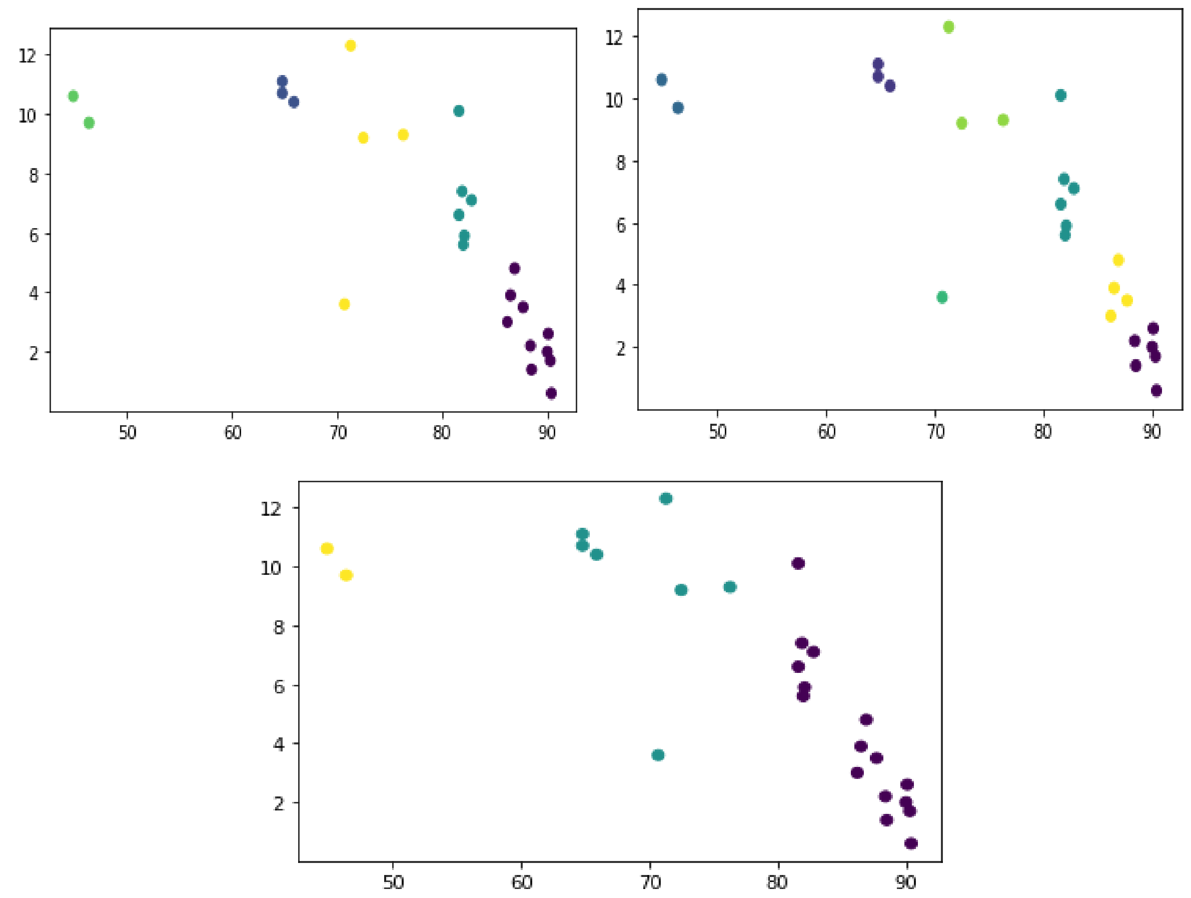
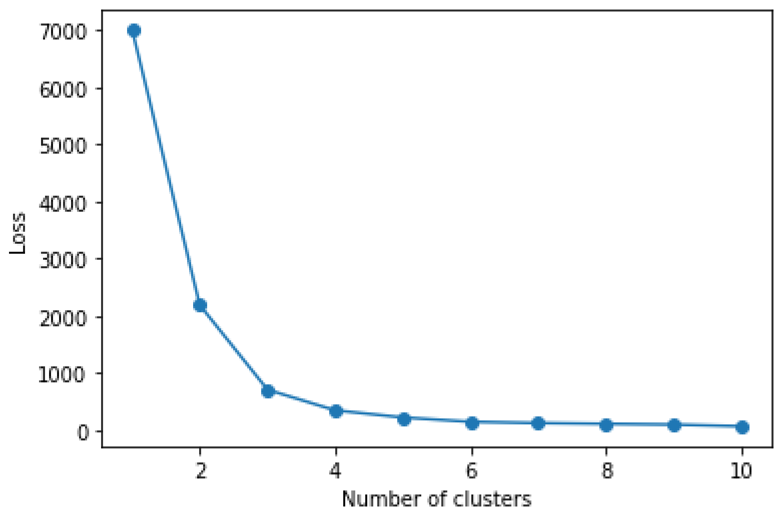


Figure 1: K Means with Inbuilt library with k = 3,5,7 respectively

By using inertia loss and elbow plot, we try to see which value of k will give the best clustering.

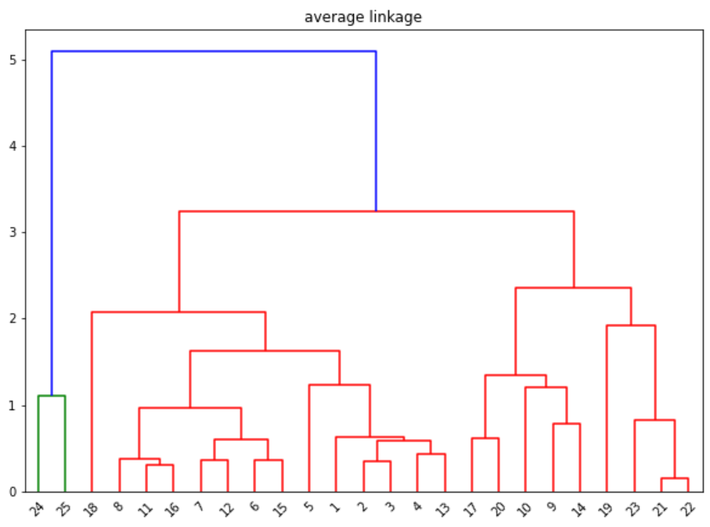


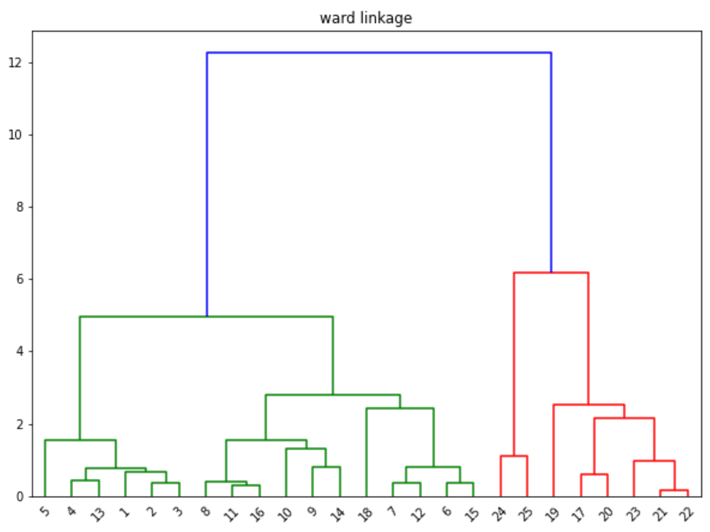
We see that there is an elbow point at k=3. Hence we conclude that k=3 provides the best clustering.

Task 2 part 1:

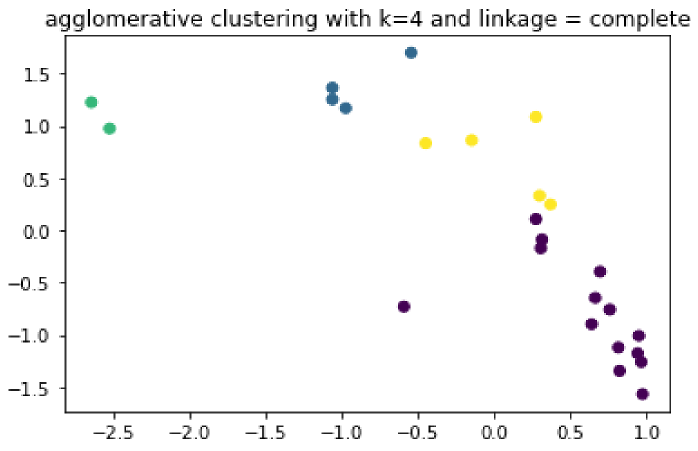
https://colab.research.google.com/drive/1wAzGSBoUi5_G-3ViNrqiOwDbMkOyJCbm?usp=sharing

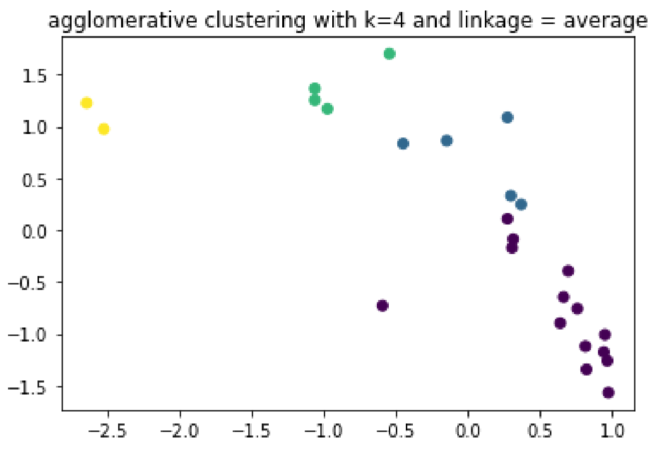
Dendrogram visualization:
To do this, we import dendrogram and linkage from scipy library. Then for different types of linkages, we cluster the data together and plot the dendrogram.

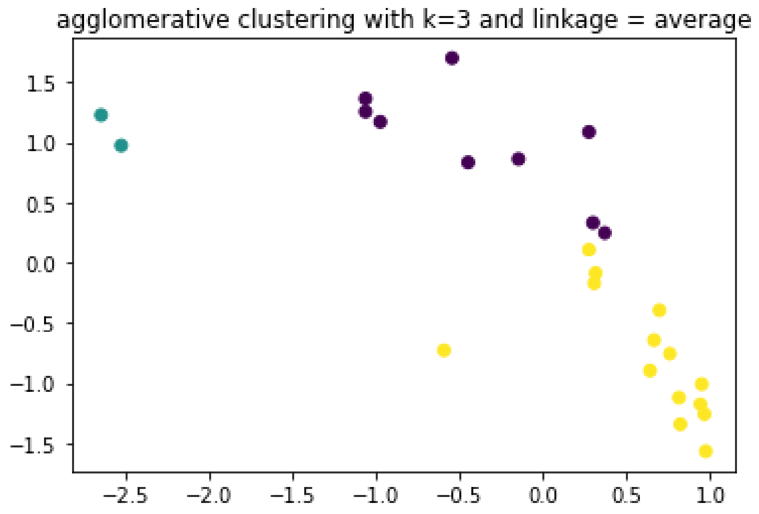




We see that average linkage doesn't work very well.







<u>Task 2 part 2 - KMeans from scratch</u>

Preprocessing:
As usual, we must standardize the data for k means to work without any bias.

Steps:

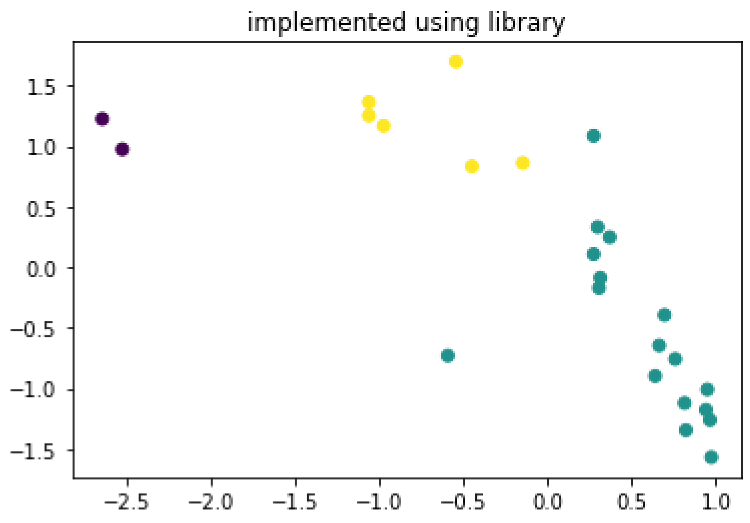
1. Initialize randomly selected k number of samples as cluster centres. This is achieve using *np.random.choice()* which returns random k integers which can be used as index
2. Then using function *euclidean()* assign all samples the clusters closest to them
3. Now over a while loop with condition that the distance previous clusters and the updated ones if not less than 0.005 :

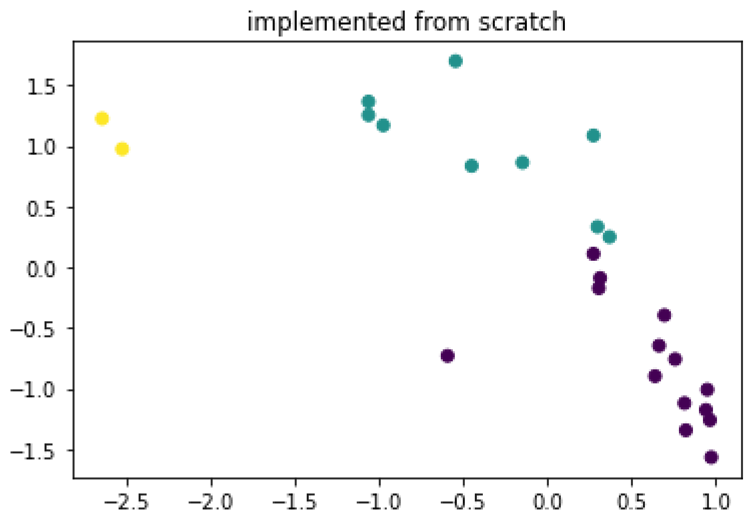
    a. Find the mean of the data, clusterwise
    b. This mean vector is the new cluster centre
    c. Cluster_centre ← mean_of_cluster
4. Assumption:

    Because K-means is an iterative algorithm, we can't be sure that it will EXACTLY converge where previous and updated cluster centres overlap exactly. This will eventually happen, however a close approximation is often good enough and can save computation. Hence I have used dist(previous cluster centres, updated cluster centres)==0.005 as the convergence criterion

Results obtained :

Here I am treating the labels predicted by the inbuilt library of K Means and comparing it with the labels obtained from my implementation to gauge its performance.





We can see that there isn't much of a difference when k=3 and plotting is against the first two features i.e : WATER and PROTEIN