UNSUPERVISED LEARNING ASSIGNMENT

Machine Learning

COMPUTATIONAL BIOLOGY MASTER'S DEGREE 2021-2022
Sandra Alonso Paz

Data preparation

After storing given data into a data frame and change the index from numerical to the column 'gene' I thought to standardize the data in order to obtain better results. For this task I used StandarScaler function. However, results I obtained were worse and were less sensitive, so for this project I have worked with non-standardized data.

In any case, there is a chunk in the Jupyter notebook which can be untoggled and run with the standardized data. Everything is prepared for so.

Partitional unsupervised learning models

First step was to perform some k-means clustering by varying the number of clusters. For this task, I have created a loop for k between 1 and 15 which represents the number of clusters used to separate data values.

Inside the loop there is an instance of the Kmeans class which contains 5 different parameters:

- **Init**: method for initialization (Kmeans++, which selects initial cluster centers for k-mean clustering in a smart way to speed up convergence or random, which choose n_clusters observations (rows) at random from data for the initial centroids). As I wanted a complete random process I chose random init.
- **N_clusters (k):** The number of clusters to form as well as the number of centroids to generate. This parameter will change in every loop iteration.
- **N_init:** number of times the k-means algorithm will be run with different centroid seeds. The final results will be the best output of n_init consecutive runs in terms of inertia.
- **Max_iter:** Maximum number of iterations of the k-means algorithm for a single run.
- **Random_state:** Determines random number generation for centroid initialization. I have used 125 to make the randomness deterministic

After several combination attempts, the best one I have found is: **Init**: random, **N_init:** 10, **Max_iter:** 30 and **Random_state: 125**

Afterwards, I train the model with provided data and added the lowest value of the sum of squared errors to an array which will be used in next step to develop elbow plot. Finally, in order to obtain a visual result of this process, I have plotted by a Principal Components Analysis or PCA the distribution of the data separated in k clusters. The evolution of this distribution is shown in the Jupyter Notebook.

In order to evaluate the appropriate number of clusters I have used two different methods:

1. The elbow method:

The elbow method runs k-means clustering on the dataset for a range of values for k (from 0-14) and then for each value of k computes an average score for all clusters. By default, the distortion score is computed, the sum of square distances from each point to its assigned center.

In my case, I have used the *sse* array and Figure 1 is the result of its plot. Focusing on which point will be considered the knee point it seems to be 4 or 5.

For being sure, have found a function called *KneeLocator()* which returns the mentioned point. In this case the result was 5.

Figure 1: elbow method.

2. Silhouette coefficient

Silhouette Coefficient or silhouette score is a metric used to calculate the goodness of a clustering technique. Its value ranges from -1 to 1:

- 1: Means clusters are well apart from each other and clearly distinguished.
- 0: Means clusters are indifferent, or we can say that the distance between clusters is not significant.
- -1: Means clusters are assigned in the wrong way.

In this work I have obtained a vector with the silhouette coefficients for every value of k. All of them are over 0 so this means that although they cannot be clearly distinguished, they are not assigned in the wrong way. Best value is obtained for k = 4 with a silhouette score of 0.1240.

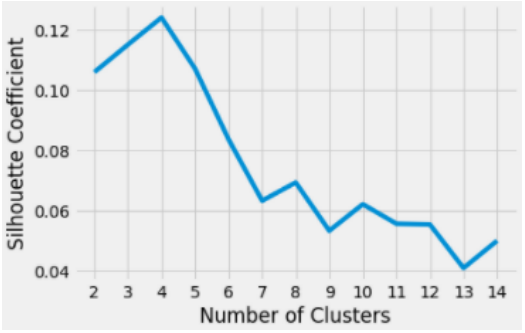

Figure 2: silhouette coefficients.

Results analysis.

As it has been clarified in last sections, in order to obtain the optimum number of clusters, it has been used elbow method through which the result acquired was k = 5 and the silhouette method which indicate that best k is 4. Let's discuss which k best fits the data.

Focusing on the distribution of the data in k clusters it is appreciable that for k = 4, dots are much ordered and separated from other groups than for k = 5. This follows the result obtained in silhouette method which shows that has higher values for k = 4 than for k = 5.

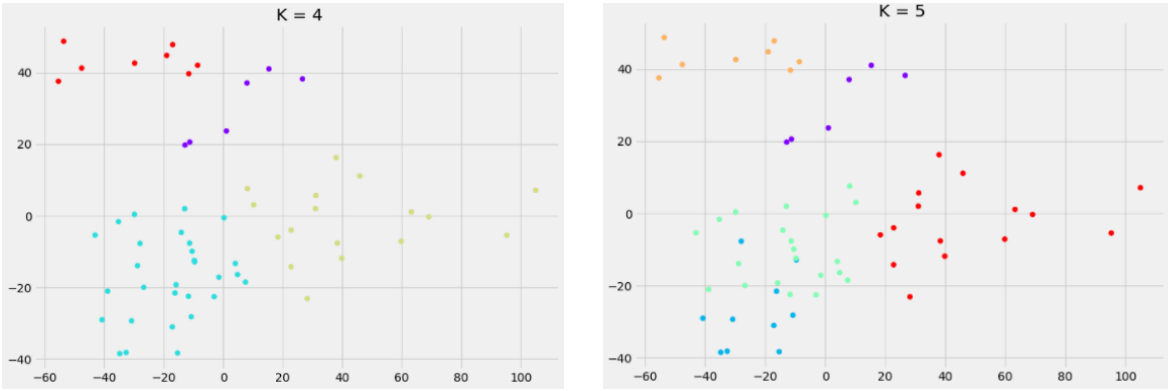

Figure 3:data distribution in 4 and 5 clusters.

In addition, I have ploted the number of iterations needed for each model to converge or reach an stable state. In case both k (4 and 5) has the same accuracy, it is better to chose the one with lowst number of iteration to avoid running time waste. As we can see in Figure 5, for 4 clusters we will need less iterarions to converge so it is a better option.

Figure 4: number of iterations need to converge per k clusters.

Moreover, I have plotted the silhouette diagram for k = 4 and 5 in order to see if they are suitable. I have obtained the results shown below in Figure 5. For a cluster distribution to be valid must have all its clusters silhouette coefficient over the mean threshold. In addition, it will be a better result if they are all balanced and their shapes are similar.
In this case, the only cluster distribution valid is k = 4 due all its silhouette coefficients are higher than the mean. However, they are not balanced at all (cluster 3 is much lagger than 1) and their shapes are not similar (cluster 1 is much bigger than 0).

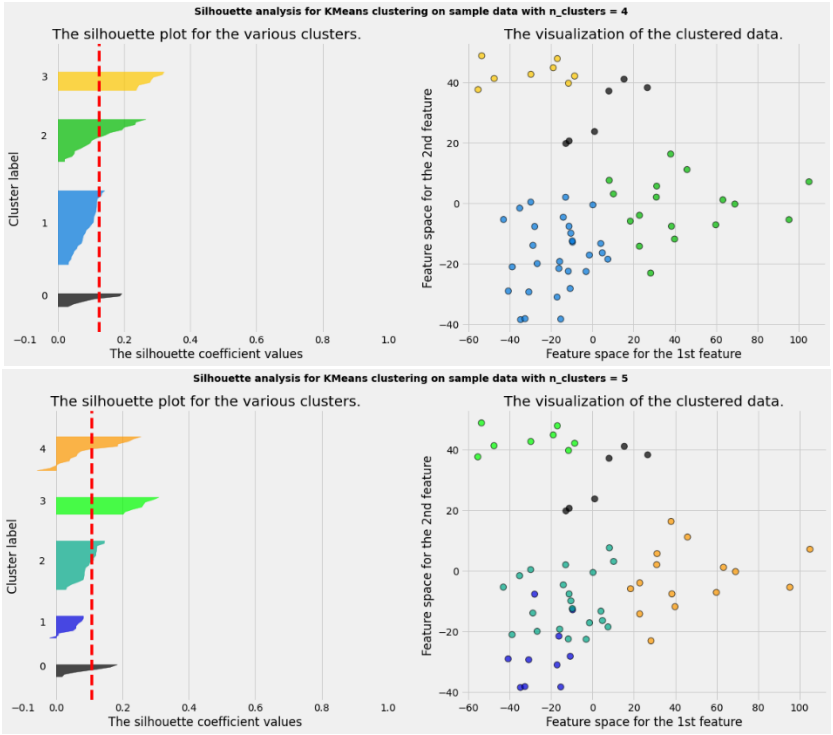

Figure 5: silhouette plot for 4 and 5 clusters.

To conclude, dividing the data in 4 clusters is the best option. This means that tissues of origin which were 9, do not match cluster assignment which are 4.
Using *predict()* function I have been able to stablish which rows of the initial data are included in each cluster and therefore their characteristics, are considered similar to the ones included in the same cluster and different to other ones.

```
Cluster  0  has  6  elems.
This are:  [6, 13, 19, 28, 40, 52]
Cluster  1  has  29  elems.
This are:  [0, 1, 2, 4, 5, 9, 11, 15, 18, 25, 29, 30, 31, 32, 35, 36, 37, 38, 39, 41, 42, 43, 44, 48, 51, 53, 54, 55, 59]
Cluster  2  has  17  elems.
This are:  [3, 7, 8, 10, 12, 14, 16, 17, 20, 21, 24, 27, 33, 34, 49, 50, 56]
Cluster  3  has  8  elems.
This are:  [22, 23, 26, 45, 46, 47, 57, 58]
```

Figure 6: elements included in each cluster.

Hierarchical unsupervised learning models

In order to develop a hierarchical model study, I have developed several models mixing different types of metrics (Euclidean, Manhattan, Chebyshev and Minkowski) and different linkage methods (Single, Ward, Complete and Average).

For each possible combination I have plotted its dendrogram, separated all the data in k clusters and done a PCA in order to see the distribution of every “dot” regarding the rest of the data.

For not taking too long I will show one example which, from my point of view, best fits the given data values.

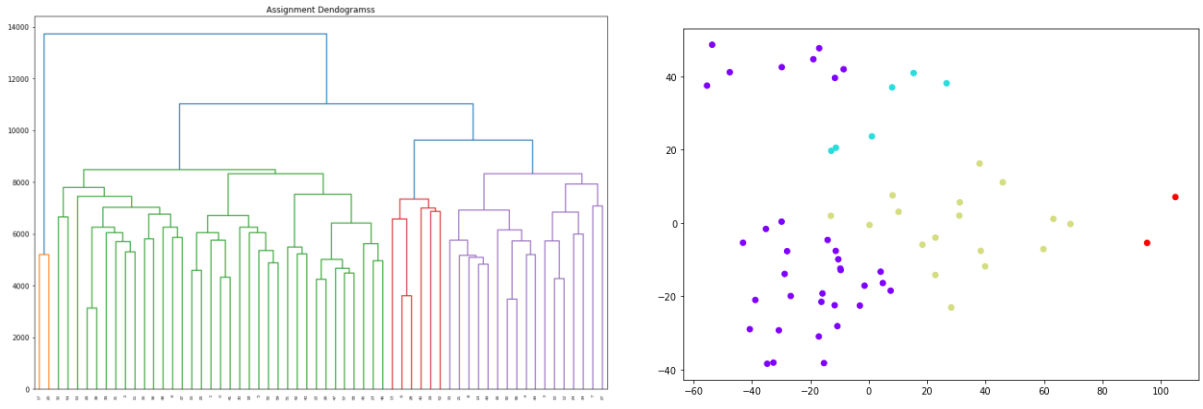

Figure 7: dendrogram and distribution of the data for complete linkage method and Manhattan metric.

Figure explanation: as colors in both plots do not match, through the number of elements in each cluster I have related them by colors: dendrogram (green, orange, red and purple) are (purple, red, blue and yellow) in the scatter plot.

Let's do some observations:

- The result of the dendrogram suggest to divide the data in 4 different cluster, this is because at the very bottom of the diagram there are 4 colors.
- In cluster 0 there are 35 elements (green in dendrogram), in cluster 1 there are 6 elements (red in dendrogram), in cluster 2 there are 17 elements (purple in dendrogram) and in cluster 3 there are just 2 elements (orange in dendrogram). This means that elements which were grouped in the same cluster, are near and therefore has similar characteristics.
- Focusing on the dendrograms levels, it needs 3 levels to separate in the final clusters (k=4).
- Having in mind the dendrogram, we can interpret that the first cluster which was separated at level 1, from the others was cluster 3 (orange in dendrogram and red in scatter diagram). Then in level 2, the remaining cluster was the divided in cluster 0 (green in dendrogram and purple in scatter diagram) and clusters 1 and 2. Finally in level 3 they were separated cluster 1 (red in dendrogram and blue in scatter diagram) and 2 (purple in dendrogram and yellow in scatter diagram). This means that cluster 1 and 2 are the closest ones so it has taken longer to separate them. Besides cluster 3 which is the furthest one an took less time to separate from others.

Linkage methods and distance metrics makes the difference when dividing the data in clusters. As sorter is the minimum distance between cluster, more clusters will be created. In the case shown below, I have chosen the complete linkage method and the Manhattan distance. This is the combination of the greatest distance between the groups (complete) and the longest path to join both (Manhattan distance).

For instance, in case of choosing the same linkage method (complete) but with the shortest path metric, which is Chebyshev, data is divided in 15 clusters (see in Figure 8). As we see in the scatter plot, higher number of clusters does not mean better distribution of the data.

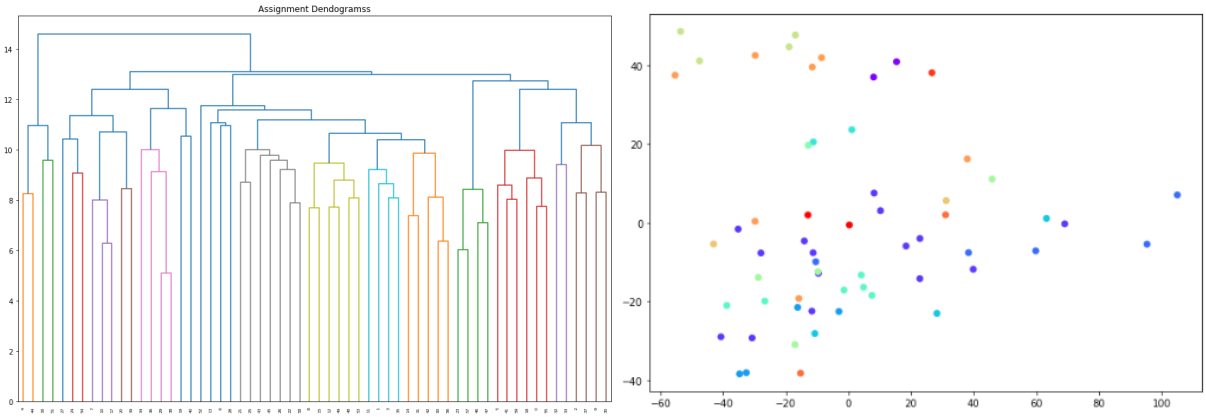

Figure 8: dendrogram and scatter plot of the data for complete linkage method and Chebyshev metric.