# ITI8730 - Data Mining
# Home Assignment 1

**LEBARON Maëlie**
Student code : 214322IV
maleba@ttu.ee

## 1 Feature Selection

### 1.1 Entropy

To compute the entropy, we need to discretize the data. We separate each dimension in $\phi$ sections. We take $value_{min} - 1\%$ and $value_{max} + 1\%$ as the extremum of the grid for each dimension. The 1% margin allow us to be sure to not delete a point.

We compute the number of point on each section of the map, and can then use the vector with the number of point by section to compute the entropy following the formula.

### 1.2 Hopkins statistics

Hopkins statistics require to have a synthetic set. Using the method runif(), we can generate a synthetic dataset. We then take a representative sample of our original dataset, selecting randomly 70% of the points in the dataset.

We will use the two dataset from figure 1 to compare the value of Hopkins statistics depending on the way the data is distributed.

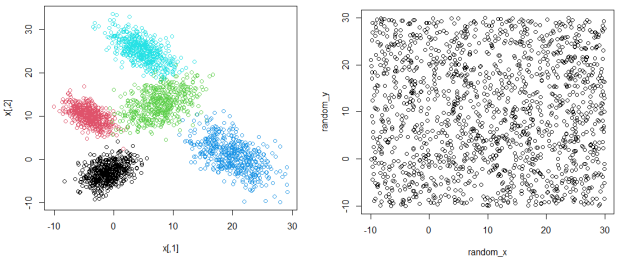


Figure 1: Clustered Dataset and randomly generated dataset

For the clustered dataset, the Hopkins statistics computed is 0.9099794, when the randomly generated dataset return the value 0.4900289. The clustered dataset has an higher Hopkins statistics, because the data is distribued in groups, therefore the feature is interesting to study.

## 2 K-means

Using the implementation from the practice, we add a converging criteria.

To decide when to stop the iteration, I used the position of the centroids. When the distance between the "old" centroids and the "new" centroid is almost the same, that mean that the algorithm is not improving more. We need to check for all the centroids : if one centroid stay at the same place, the other can still be processing.

Using this criteria, we get the figure 2 as a result of our algorithm. In red, we can see the positions of the centroids.

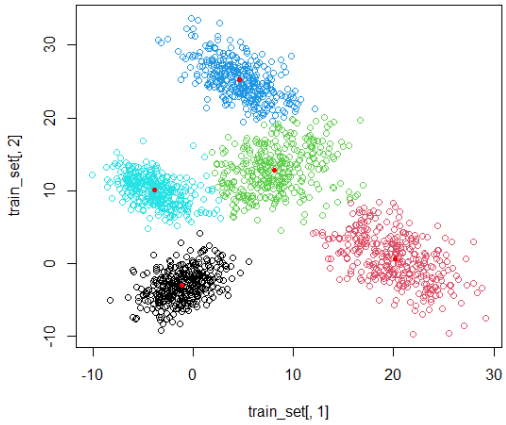


Figure 2: K-means clustering on a dataset of 5 clusters

This dataset is the same a the clustered dataset from Figure 1. If we look closely, we can see that some of the points are labelled differently. It may be due to the choice of our metric, but also simply to the algorithm, as the cluster are really close from each other, which result on some points being closer to the centroid of another cluster that their own.

## 3 EM Algorithm

To adapt the EM Algorithm to the 3D-case, we need to convert the ellipses to ellipsoids. We add a coordinate to centers of our ellipses, $\mu$, and the $\sigma$ matrix become $3 * 3$ matrix, and we take care to keep this matrix symmetrical, even if randomly generated.

During the iteration, we updates all the coordinates of $\mu$ centers and all components of $\sigma$.

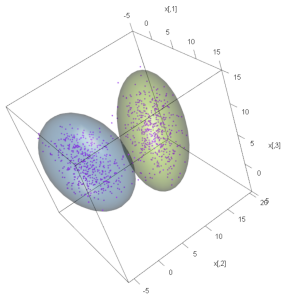

Figure 3: Example of clustering of a 3D dataset by EM Algorithm

## 4 Outlier Analysis

The implementation of the Local Outlier Factor of X is done by finding the $knearestneighbors$, the maximum distance for those neighbors, and then compute $AR_k$ depending on those values to finally calculate the local outlier factor for k neighbors. This factor can be used as the scale to represent points that are more likely to be outliers.

Using a gradient in the color, we can see graphicly the point that have the highest outlier score, as shown in figure 4.

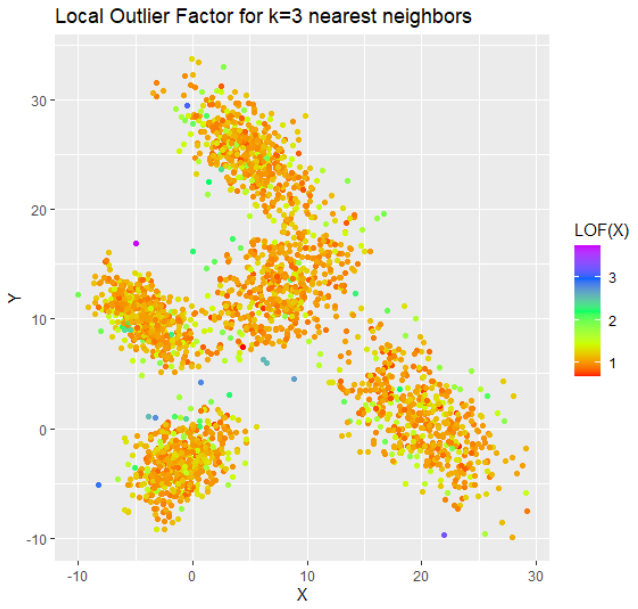


Figure 4: Outlier analysis of a dataset (Local Outlier Factor)

## 5 DBSCAN

To compute DBSCAN, we duplicate our dataset, and with the copy of this dataset, we will study each point. We will use a function to find the neighbors and the neighbors of the neighbors, etc.

We start we choosing a random point from the dataset. If it is a core point, we find the neighbors. Otherwise, we remove it from the copy of the dataset. The idea of this copy is that this point could still be part of a cluster but we will not start a cluster from it, and we already studied it.

Each time we have a core point and the list of point from the same cluster, we deleted those point from the copy of the original dataset, because they are now labelled.

Unfortunately, my implementation of DBSCAN don't seem to work as expected, but I described the idea I tried to implement.