TP ML: DBSCAN

MASTER 2019/2020

Réalisé par : Nour el houda MEHADJIBIA
Groupe : 2

Table des matières

Introduction: 2

Concept de base: 2

Choix des paramètres : 3

Algorithme : 4

Etude de cas : 4

Implémentation sur R : 5

Résultats : 7

Introduction:

Le DBSCAN (density-based spatial clustering of applications with noise) est un algorithme d'apprentissage non-supervisé basé sur la densité, il a été proposé en 1996 par Martin Ester, Hans-Peter Kriegel, Jörg Sander et Xiaowei.

L' DBSCAN algorithme considère les clusters comme des zones de haute densité séparées par des zones de faible densité. En raison de cette vue plutôt générique, les grappes trouvées par DBSCAN peuvent avoir n'importe quelle forme, par opposition à k-means qui suppose que les grappes ont une forme convexe.

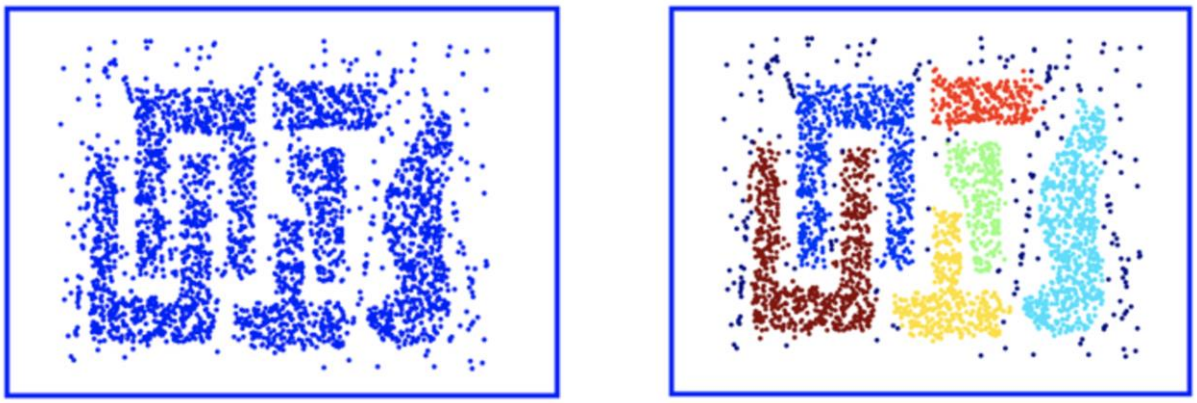


Figure 1: K-means vs DBSCAN

Concept de base:

Le composant central du DBSCAN est le concept d'échantillons de base, qui sont des échantillons situés dans des zones de forte densité. Une grappe est donc un ensemble d'échantillons de cœur, proches les uns des autres (mesurés par une mesure de distance) et un ensemble d'échantillons autres que de cœur qui sont proches d'un échantillon de base (mais ne sont pas eux-mêmes des échantillons de base). Il y a deux paramètres de l'algorithme, min_samples et eps qui définissent formellement ce que nous voulons dire quand nous disons dense. Plus haut min_samples ou plus bas eps indiquer une densité plus élevée nécessaire pour former un cluster.

Plus formellement, nous définissons un échantillon de base comme étant un échantillon dans l'ensemble de données, de sorte qu'il existe d' min_samples autres échantillons à une distance de eps, définis comme voisins de l'échantillon de base. Cela nous indique que la carotte est

dans une zone dense de l'espace vectoriel. Une grappe est un ensemble d'échantillons de base pouvant être générés en prenant un échantillon de base de manière récursive, en trouvant tous ses voisins qui sont des échantillons de base, en recherchant tous les voisins qui sont des échantillons de base, etc. Une grappe a également un ensemble d'échantillons non principaux, qui sont des échantillons voisins d'un échantillon principal de la grappe mais ne sont pas eux-mêmes des échantillons principaux. Intuitivement, ces échantillons sont en marge d'un cluster.

Tout échantillon de base fait partie d'un cluster, par définition. Tout échantillon qui n'est pas un échantillon principal et qui se trouve à au moins une epsdistance de tout échantillon principal est considéré comme une valeur aberrante par l'algorithme.

Choix des paramètres :

Le paramètre min_samples contrôle principalement la tolérance de l'algorithme au bruit (sur des ensembles de données volumineux et bruyants, il peut être souhaitable d'augmenter ce paramètre). Pour des données qui contient beaucoup de bruits il est recommandé de choisir **min_samples = 2 * dim,** d'autres approches recommandent d'utiliser

min_samples = dim + 1 , min_samples = log(n) « ou n est le nombre de points regroupés »

Le paramètre eps contrôle le voisinage local des points. Si choisi trop petit, la plupart des données ne seront pas du tout groupées (et étiquetées comme -1«bruit»). Lorsque la taille choisie est trop grande, les clusters proches sont fusionnés dans un cluster et, éventuellement, l'ensemble du jeu de données est renvoyé sous la forme d'un cluster unique. Certaines heuristiques pour choisir ce paramètre ont été discutées dans la littérature, par exemple sur la base d'un genou dans le tracé des distances les plus proches **(k-distances).**

Algorithme :

```
DBSCAN (DB, distFunc, eps, minPts) {
   C = 0                                                  / * Compteur de cluster * /
   pour chaque point P dans la base de données DB {
       si étiquette (P) indéfini, puis  continuer              / * précédemment traité dans la boucle
interne * /
      Voisins N = RangeQuery (DB, distFunc, P, eps)       / * Trouver des voisins * /
      if | N | <minPts then {                              / * Contrôle de la densité * /
         label (P) = Bruit                                 / * Label comme Bruit * /
         continue
      }
      C = C + 1                                           / * étiquette suivante du cluster * /
      étiquette (P) = C                                        / * étiquette point initial * /
      ensemble de semences S = N \ {P}                               / * Voisins à développer * /
      pour chaque point Q dans S {                             / * Traiter chaque point de départ * /
         si étiquette (Q) = Bruit puis étiquette (Q) = C            / * Modifier le bruit en point
frontière * /
         si étiquette (Q) ≠ non défini puis  continuer          / * déjà traité * /
         étiquette (Q) = C                                   / * Label voisin * /
         Voisins N = RangeQuery (DB, distFunc, Q, eps)    / * Recherche de voisins * /
         if | N | ≥ minPts then {                           / * Contrôle de la densité * /
            S = S U N                                      / * Ajouter de nouveaux voisins au jeu de
semences * /
         }
      }
   }
}
```


L'algorithme DBSCAN peut être résumé aux étapes suivantes:

1. Trouvez les points situés dans le voisinage ε (eps) de chaque point et identifiez les points centraux avec plus de voisins voisins.
2. Recherchez les composants connectés des points centraux sur le graphe voisin, en ignorant tous les points non centraux.
3. Attribuez chaque point non essentiel à un cluster proche si le cluster est un voisin ε (eps), sinon affectez-le au bruit.

Une mise en œuvre naïve de cette opération nécessite de stocker les voisinages à l'étape 1, nécessitant ainsi une mémoire importante. L'algorithme DBSCAN d'origine ne l'exige pas en effectuant ces étapes pour un point à la fois.

Etude de cas :

Base de données originale du Wisconsin sur le cancer du sein.

Informations d'attribut :

1. Numéro de code de l'échantillon: numéro d'identification

2. Épaisseur de touffe: 1 - 10

3. Uniformité de la taille de la cellule: 1 - 10

4. Uniformité de la forme de la cellule: 1 - 10

5. Adhérence marginale: 1 - 10

6. Taille de la cellule épithéliale unique : 1 - 10

7. Noyaux nus: 1 - 10

8. Chromatine nue: 1 - 10 9. Noyaux

9. normaux: 1 - 10

10. Mitoses: 1 - 10

11. Classe: (2 pour bénigne, 4 pour maligne)

Implémentation sur R :

Voici le script :

```
# chargement des données ----------------------


data_original <- read.table(file.choose(), header = TRUE, sep = ",")

data_original = data_original[,-1]

data_original = data

dim(data_original)

summary(data_original)

data = data_original[,-10]

# DBSCAN  avec 9 attributs ---------------------------------

data_test1 = data

kNNdistplot(data_test1, k = 10)

c1 = dbscan(data_test1,8,10)
```

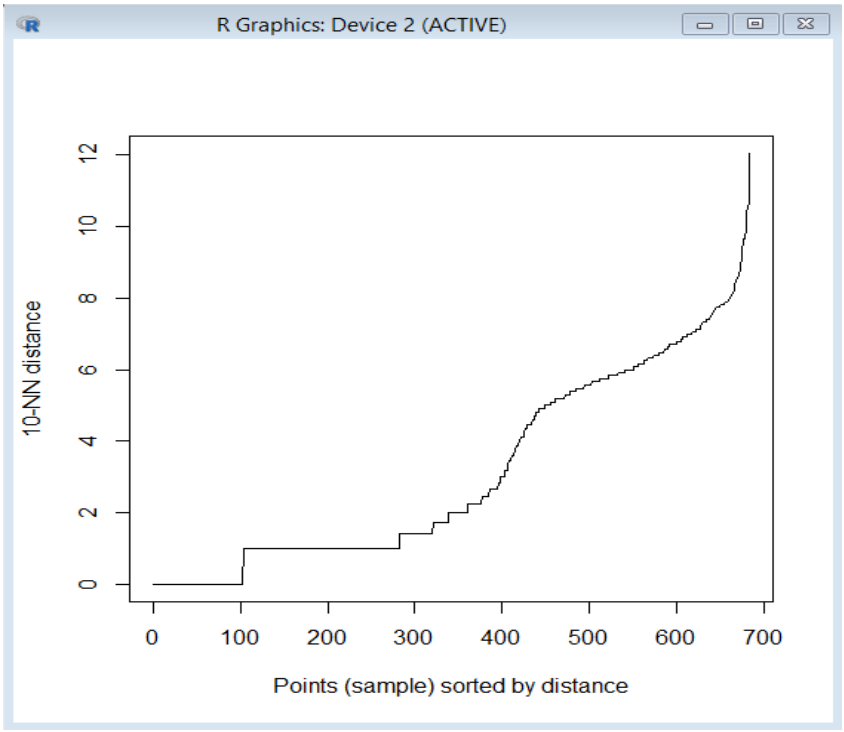
```
c1

clusplot(data_test1,c1$cluster,color= TRUE, labels = 3)

hullplot(data_test1 , c1)

# featrures selection ----------------------------------

set.seed(100)

dbscanMod <- train(Class~., data=data_original, method="rpart")

dbscanImp <- varImp(dbscanMod)

print(dbscanImp)

# DBSCAN aprés une feauters selection 6 attributs --------------------------------

data_test11 = data_test1

data_test2 = data_test11[,-1]

data_test2 = data_test2[,-8]

data_test2 = data_test2[,-7]

dim(data_test2)

kNNdistplot(data_test2, k = 7)

c2 = dbscan(data_test2,6,7)

c2

clusplot(data_test2,c2$cluster,color= TRUE, labels = 3)

# DBSCAN aprés une feauters selection 3 attributs --------------------------------

data_test22 = data_test2

data_test3 = data_test22[,-4]

data_test3 = data_test22[,-4]

data_test3 = data_test22[,1:3]

dim(data_test3)
```

```
kNNdistplot(data_test3, k = 4)

c3 = dbscan(data_test3,1.5,4)

c3

clusplot(data_test3,c3$cluster,color= TRUE, labels = 3)

hullplot(data_test3 , c3)
```

Résultats :

Sans Features selection :

1. K-distance :



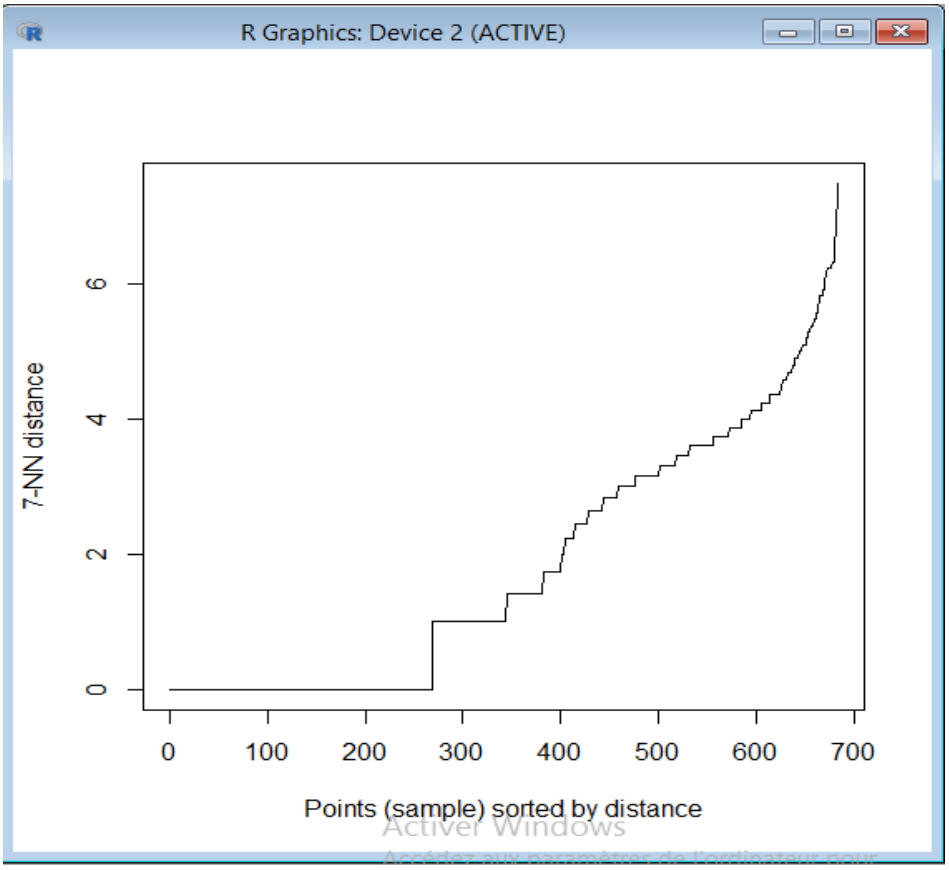
2. DBSCAN :

```
DBSCAN clustering for 683 objects.
Parameters: eps = 8, minPts = 10
The clustering contains 1 cluster(s) and 3 noise points.

  0   1
  3 680

Available fields: cluster, eps, minPts
```

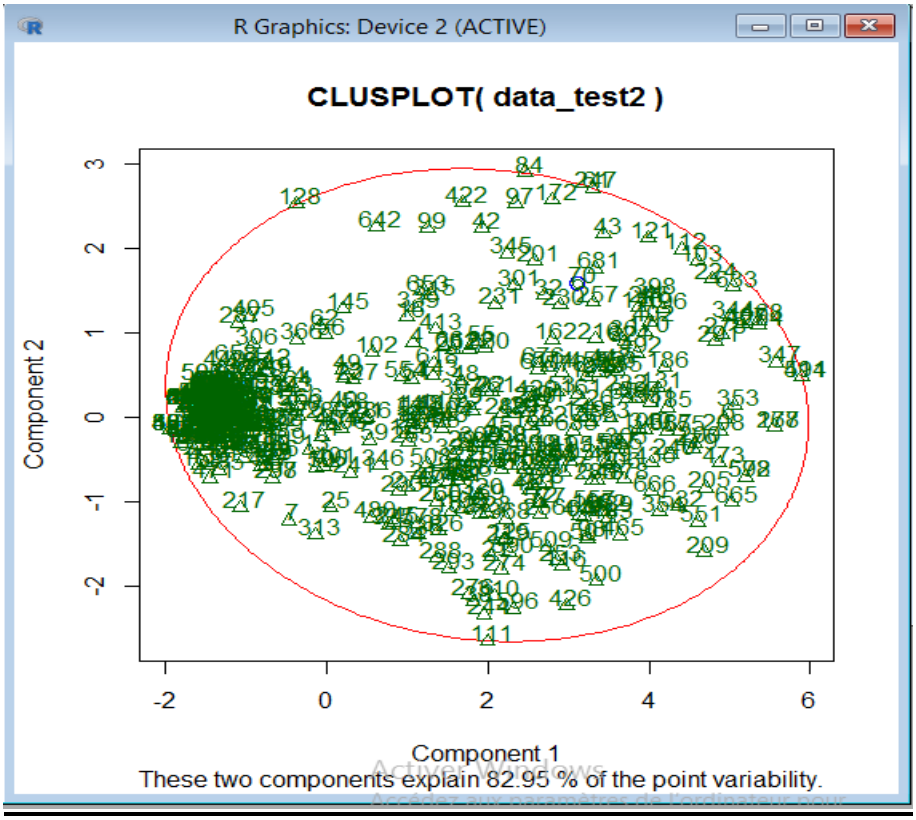
Avec features selection :

```
rpart variable importance

                             Overall
Uniformity.of.Cell.Shape      100.00
Uniformity.of.Cell.Size        98.38
Bare.Nuclei                    90.26
Bland.Chromatin                83.52
Single.Epithelial.Cell.Size    61.38
Marginal.Adhesion              16.95
Normal.Nucleoli                 0.00
Clump.Thickness                 0.00
Mitoses                         0.00
```

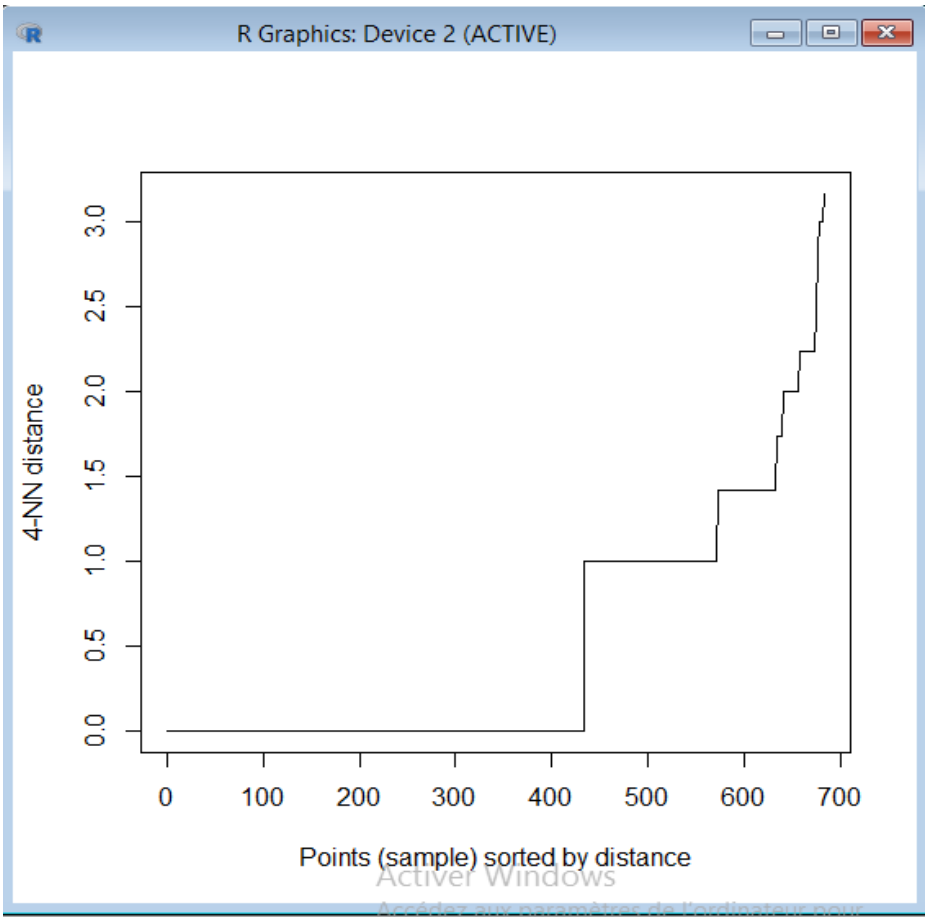
1. Avec 6 variables :




```
DBSCAN clustering for 683 objects.
Parameters: eps = 6, minPts = 7
The clustering contains 1 cluster(s) and 1 noise points.

  0   1
  1 682

Available fields: cluster, eps, minPts
```



2. Avec 3 variables :



```
DBSCAN clustering for 683 objects.
Parameters: eps = 1.5, minPts = 4
The clustering contains 3 cluster(s) and 29 noise points.

  0   1   2   3
 29 620  24  10

Available fields: cluster, eps, minPts
```