University of Southern Denmark
IMADA
Ricardo Campello

Credits:

- Exercise 6.1 was adapted and extended from Arthur Zimek's original for DM870
- Exercise 6.5 is fully reproduced from Arthur Zimek's original for DM870

DM583: Data Mining

Exercise 6: Density Based Clustering and Outlier Detection

Exercise 6-1 Density-Based Clustering (Conceptual)

a) Consider a dataset in a D-dimensional Euclidean space, consisting of two clusters. Assume that these clusters are each contained within well-delineated spatial boundaries, specifically, each cluster is spatially restricted to the interior of its own D-dimensional "ball" centered at a particular location of the space. Assume these clusters both follow a *uniform distribution* within such boundaries. Initially, also assume that: (i) these boundaries do not overlap and are well separated; and (ii) an observation in the dataset is equally likely to come (i.e., to have been drawn) from either cluster. Finally, assume that the dataset (sample) size is large enough so a rough KNN density estimate as the one used by DBSCAN can reasonably capture the density profile of the dataset for a given choice of the neighborhood size, MinPts, allowing for both cluster detection and separability. Describe what you would expect in this case for different choices of the radius ε, including values that are too high or too low. Note: For simplicity, you can ignore DBSCAN's border points (i.e., you can think of DBSCAN*).

b) Repeat item (a), but now assuming that observations in the dataset are more likely to belong to one of the two clusters.

c) Repeat item (a), but now assuming that, within the same cluster boundaries as before, the distribution (shared by both equally likely clusters) is radial rather than uniform, i.e., density is the highest at the center and lowest at the borders.

d) If for each object (observation) in the dataset of size n we need to find, among all other n-1 objects, the ones within a distance ε, how come that the asymptotic computational complexity of DBSCAN can be made smaller than $O(n^2)$ in certain application scenarios?

e) Discuss the relation between the algorithms DBSCAN*, HDBSCAN* and Single-Linkage for MinPts = 1 or 2. Is there any difference between the cases where MinPts = 1 or 2?

f) Argue that border points in DBSCAN (which do not exist in DBSCAN*) conceptually violate Hartigan's classic model of density-contour clusters.

g) Discuss possible ways to solve ties for border points shared by core points from different clusters.

Exercise 6-2 DBSCAN Algorithm at Work

a) Consider the (symmetric) dissimilarity matrix $\boldsymbol{D}$ below, containing the pairwise distance between the data objects in a dataset. Manually apply DBSCAN to cluster this dataset with ε = 5 and MinPts = 3 (which includes the query object in question), describing the partial outcomes step-by-step, then indicating the final type of each object (core, border or noise) as well as the resulting clusters. Note: in case a border object is within reach from

core points from multiple clusters, you can just assign them to these clusters (overlapping assignment allowed).

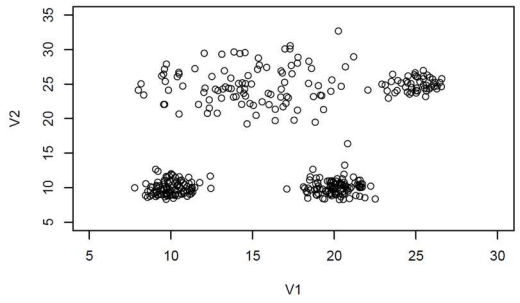
$$\mathbf{D} = \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{array} \begin{bmatrix} 0 & & & & & & \\ 1 & 0 & & & & & \\ 1 & 2 & 0 & & & & \\ 10 & 9 & 11 & 0 & & & \\ 11 & 10 & 12 & 2 & 0 & & \\ 21 & 20 & 22 & 18 & 19 & 0 & \\ 14 & 13 & 15 & 6 & 4 & 25 & 0 \end{bmatrix}$$

b) Repeat for other values of ε and MinPts.

Exercise 6-3 Practical DBSCAN and HDBSCAN* in R

a) Use trial-and-error to determine a suitable combination of values for minPts and ε that allows DBSCAN to reasonably recover the 4 clusters in the data set produced by the R code below (see figure). Plot your solution with the detected clusters in different colours and the noise points in black.

```
> set.seed(0)
> V11 <- rnorm(n = 100, mean = 10, sd = 1) # Cluster 1 (V1 coordinate)
> V21 <- rnorm(n = 100, mean = 10, sd = 1) # Cluster 1 (V2 coordinate)
> V12 <- rnorm(n = 100, mean = 20, sd = 1) # Cluster 2 (V1 coordinate)
> V22 <- rnorm(n = 100, mean = 10, sd = 1) # Cluster 2 (V2 coordinate)
> V13 <- rnorm(n = 100, mean = 15, sd = 3) # Cluster 3 (V1 coordinate)
> V23 <- rnorm(n = 100, mean = 25, sd = 3) # Cluster 3 (V2 coordinate)
> V14 <- rnorm(n = 50, mean = 25, sd = 1) # Cluster 4 (V1 coordinate)
> V24 <- rnorm(n = 50, mean = 25, sd = 1) # Cluster 4 (V2 coordinate)
> dat <- data.frame(V1 = c(V11,V12,V13,V14), V2 = c(V21,V22,V23,V24))
> plot(dat$V1, dat$V2, xlim = c(5,30), ylim = c(5,35), xlab = "V1", ylab = "V2")
```



b) Repeat for HDBSCAN*, displaying both the hierarchical solution (simplified cluster tree, which you should interpret) as well as the flat partitioning that can (optionally) be automatically extracted by the algorithm.

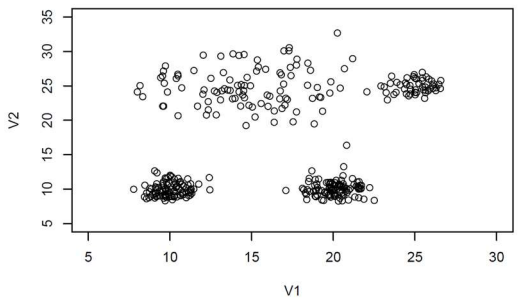
Disclaimer about the R implementation of HDBSCAN*: *The R implementation of HDBSCAN* from package* `dbscan()` *– function* `hdbscan()` *– is not memory efficient (as of writing of this document) and may produce a memory overflow error for datasets larger than a few tens of thousands of observations, depending on your hardware specs. Alternative implementations exist that are computationally much more efficient, e.g., in Python.*

c) Discuss the main differences between the experiment with DBSCAN in item (a) as contrasted to the experiment with HDBSCAN* in item (b).

Exercise 6-4 Practical KNN Outlier and LOF in R

a) Compute and display the top-20 KNN outliers for the dataset produced by the following code (see figure below), using a neighbourhood size $k = 4$. Use the standard (not weighted) KNN method.

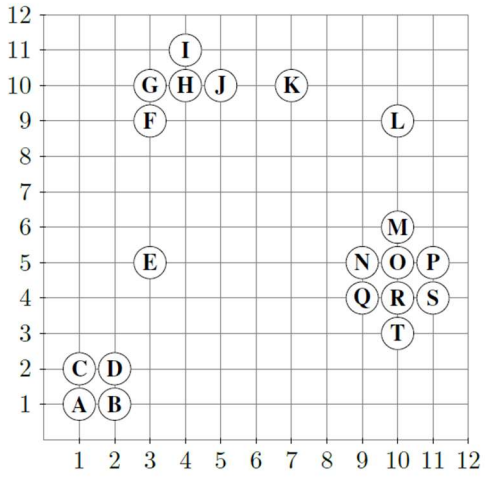
```
> set.seed(0)
> V11 <- rnorm(n = 100, mean = 10, sd = 1) # Cluster 1 (V1 coordinate)
> V21 <- rnorm(n = 100, mean = 10, sd = 1) # Cluster 1 (V2 coordinate)
> V12 <- rnorm(n = 100, mean = 20, sd = 1) # Cluster 2 (V1 coordinate)
> V22 <- rnorm(n = 100, mean = 10, sd = 1) # Cluster 2 (V2 coordinate)
> V13 <- rnorm(n = 100, mean = 15, sd = 3) # Cluster 3 (V1 coordinate)
> V23 <- rnorm(n = 100, mean = 25, sd = 3) # Cluster 3 (V2 coordinate)
> V14 <- rnorm(n = 50, mean = 25, sd = 1) # Cluster 4 (V1 coordinate)
> V24 <- rnorm(n = 50, mean = 25, sd = 1) # Cluster 4 (V2 coordinate)
> dat <- data.frame(V1 = c(V11,V12,V13,V14), V2 = c(V21,V22,V23,V24))
> plot(dat$V1, dat$V2, xlim = c(5,30), ylim = c(5,35), xlab = "V1", ylab = "V2")
```



b) Repeat for LOF.

Exercise 6-5 KNN Outlier, Weighted KNN Outlier, and LOF at Work

Consider the following dataset in 2 dimensions:



As distance function, use Manhattan distance $L_1(a, b) := |a_1 - b_1| + |a_2 - b_2|$.

Compute the following (*without including the query point when determining the kNN*):

- KNN Outlier using $k = 2$ and $k = 4$ for all points
- Weighted (aggregated) KNN Outlier for $k = 2$ and $k = 4$ for all points (aggregated kNN distance = averaged distances to all the kNNs)
- LOF using $k = 2$ for points E, K and O
- LOF using $k = 4$ for points E, K and O