TU Dortmund
Artificial Intelligence / Data Science and Data Engineering
Prof. Dr. Erich Schubert
Prof. Dr. Emmanuel Müller
Daniel Wilmes

3.6.2022
Machine Learning for Complex Data

Summer 2022

Assignment 7: OPTICS and SUBCLU

Due: Thursday, 9.6.2022

Problem 7-1 Monotonicity in SUBCLU 7

Let D be a d-dimensional dataset, $\mathcal{A}$ be the set of all attributes, $S \subseteq \mathcal{A}$ be a subspace and $p \in D$. Prove that for arbitrary $\varepsilon \in \mathbb{R}^+$ and $\text{minPts} \in \mathbb{N}$ it holds:

$$\forall T \subseteq S : |\mathcal{N}_\varepsilon^S(p)| \geq \text{minPts} \Rightarrow |\mathcal{N}_\varepsilon^T(p)| \geq \text{minPts}$$

with $\mathcal{N}_\varepsilon^S(p) := \{x \in D | dist(\pi_S(p), \pi_S(x)) \leq \varepsilon\}$, where $dist$ is one of the L_p-norms and $\pi_S(o)$ is the projection of an object o into the subspace of S.

Problem 7-2 Implement OPTICS Clustering 15

In this assignment, we will implement a *simplified* version of OPTICS, that is deterministic except for permutation of the input data. We only use $\varepsilon = \infty$ and no heap, so the runtime will be $O(n^2)$.

a) Implement the function `reachability(X, i, minpts)`, which given a data set X, a row number i, and minPts returns the reachability distances $\text{ReachDist}(x_j \leftarrow x_i)$ for all rows $x_j \in X$. You may use a library function to do the distance computations.

b) Implement a function `find_min` that returns an index i, such that (i) object i is unprocessed, (ii) there exists no object $j \neq i$ which currently has a lower reachability *and* a lower index. (If multiple objects have the same current reachability, the lowest index *must* be returned.)

c) Implement OPTICS clustering, using above functions. Use `find_min` to find the next element, add a row to the cluster order, mark the element as processed, compute the `reachability`, and update ReachDists. Return the resulting cluster order.

d) Run your OPTICS implementation on the iris data set (without shuffling the data set), with minPts= 10 (use `sklearn.datasets.load_iris().data` to get the data set).
Extract the reachability column from the cluster order. Write a program that (for the plot *only*) replaces any infinite value with 1.1 times the maximum of all finite values.
Write generic code, that can process *any* reachability array.

Use a *step* plot to draw the reachability plot.
Make sure your plot shows $\text{ReachDist}(\lfloor x \rfloor)$.

e) Write a method `extract(order, height)` to cut a cluster order at the given height. For any point i with $\text{ReachDist}(i) \leq \text{height}$, assign the same cluster label to the point *and* its predecessor. Points in different valleys must be assigned different labels, and all remaining points must be assigned the cluster label -1.

f) Plot the cluster labeling you get when extracting clusters from the iris at height 0.6.

g) Run the sklearn DBSCAN algorithm with the same parameters, and plot the result.

h) Compare the two results using the ARI and NMI indexes, using the sklearn implementation. Discuss your result.