CS5228 – Tutorial 2

Clustering: K-Means & DBSCAN

K-Means and DBSCAN are two very popular clustering methods. Since clustering is typically used to find patterns in unlabeled data, it is generally very difficult to reliably assess if a resulting clustering is ”good”. This makes it even more important to properly understand the underlying principles, as well as the pros and cons of the different methods to better interpret the results.

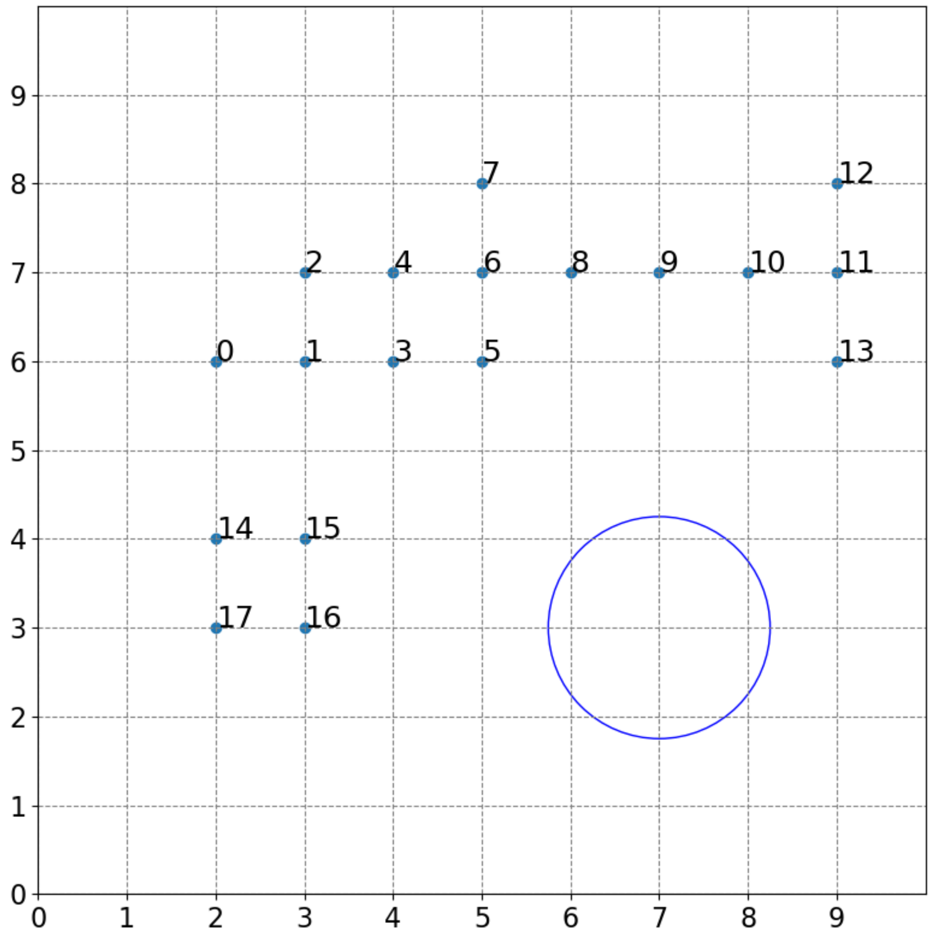

Figure 1: Toy dataset for ”manually” performing DBSCAN.

1. **Performing DBSCAN "by hand".** Figure 1 shows a toy dataset with 18 data points. Let's assume we run DBSCAN over this data with $\epsilon = 1.25$ and $MinPts = 4$.

   (a) What will be the result of DBSCAN? Describe the ouput by listing all

   - core points
   - border points
   - noise points

   Solution:

   - core points = [1, 3, 4, 6, 11]
   - border points = [0, 2, 5, 7, 8, 10, 12, 13]
   - noise points = [9, 14, 15, 16, 17]

   (b) How many clusters are there, and what are their data points?

   Solution:

   - Cluster 1 = [0, 1, 2, 3, 4, 5, 6, 7, 8]
   - Cluster 2 = [10, 11, 12, 13]

   (c) Can you add 2 data points such that the resulting clustering contains only 1 cluster and no noise? If so, give the coordinates for both points!

   Solution:

   - Example solution: (2.5, 5), (7, 6.8)
   - Note that this is not a unique solution

   The simplicity of the toy dataset should allow you to answer all three tasks by just looking at the plot in Figure 1. There should be no need to actually calculate any distances. The blue circle reflects the chosen radius of $\epsilon = 1.25$ to make it easier for you.

2. **K-Means.** For the following questions, assume that we now want to run K-Means over the toy dataset shown in Figure 1.

   (a) For $K = 3$, can you find locations for the initial centroids so that the resulting clustering will contain 0, 1, 2, or 3 non-empty clusters? You can answer this question qualitatively; there is no need to list any exact coordinates for the initial centroids.

   Solution:

- There will always be at least 1 non-empty cluster; so 0 non-empty clusters can never happen.
- 1 or 2 clusters means 2 or 1 empty clusters. For example, the get 2 empty clusters, we only need to place 1 centroids "within" the dataset, and the 2 other centroids just far away.
- It is easy to see that placing the 3 centroids at the locations 4, 11, and 16 (not the only initialization) will result in 3 non-empty clusters.

(b) Now we assume that K-Means++ initialization is used. For $K = 3$, what is the minimum and maximum number of clusters? (Comment: For this question you may need to consider arbitrary datasets and not just the toy dataset!)

Solution:

- The number of clusters C is guaranteed to be in $1 \leq C \leq K$.
- After an update setup – typically already after the first one – the centroids are generally no longer at the location of data points. This means that in the next assignment there might now be a centroid which is not the closest to any other data point, thus yielding an empty cluster.

3. **K-Means vs. DBSCAN**

(a) Apart from their implementation, what is the fundamental difference between K-Means and DBSCAN?

Solution:

- K-Means is defined as an optimization problem; hence there is a notion of local and global optimal solutions. The commonly used Lloyd's algorithm is a heuristic that does no guarantee to always find the global optimum
- K-Means considers relative similarities/distances
- K-Mean favors blob-like clusters
- DBSCAN is not defined as an optimization problem, so there's no notion of a local/global optimum. The DBSCAN algorithm is not a heuristic.
- DBSCAN considers absolute similarities/distances
- DBSCAN can handle non-blob-like clusters

(b) What are meaningful criteria to decide whether K-Means or DBSCAN is the preferable clustering method for a certain task?

Solution:

- DBSCAN has the notion of noise, which can be used for outlier detection
- DBSCAN better when clusters are decidedly not blobs
- DBSCAN can work well if the parameters for ϵ and $MinPts$ can be intuitively set
- K-Means when the value for k is predefined by application context
- since K-Means considers relative similarities/distances, it's arguably easier to use for an EDA to get a meso-view of the data

(c) Come up with 5 example tasks and discuss why K-Means or DBSCAN would be your method of choice!

Solution:

1. Traffic congestion along roads based on location of cars (DBSCAN)
2. Identifying locations of low coverage, e.g., distribution of Starbucks outlets across a city (DBSCAN)
3. Credit card fraud or intrusion detection to find outliers (DBSCAN)
4. Organizing conference papers into a given set of research areas (K-Means)
5. Clustering people based on biological data (e.g., height, weight). Such data is typically always normally distributed so clusters are more likely to be blobs (K-Means).

(d) Is there any example where fundamentally only K-Means is applicable but not DBSCAN, or vice versa?

Solution:

- No, both methods only require a well-defined similarity/distance measure to be applicable.

(e) Is it possible that both K-Means and DBSCAN return the same clustering for a dataset or will the clusterings always be different?

Solution:

- It is easy to create or image a dataset where both algorithms – assuming the appropriate parameter values – will return the same clustering.
- Straightforward example: a dataset with very well-separated (i.e., very obvious) clusters. With the right parameters, both K-Means and DB-SCAN will return the same set of those well-separated clusters.