Summarization of DBSCAN algorithm

For DBSCAN, two parameters have to be decided: eps and $minPts$ ($eps \geq 0$, $minPts \geq 1$).

Consider each data point in the hyperspace as a vertex in a graph, and denote $dist(u, v)$ as the distance between two data points u and v. A vertex u has a directed edge to another vertex v if (1) $dist(u, v) \leq eps$ and (2) there are at least $minPts$ vertices t (including u) where $dist(u, t) \leq eps$. Based on this, we build a directed graph $G = (V, E)$ where V is the set of all data points.

A vertex u is called a "core" if (1) $minPts = 1$ or (2) there is an edge from u to another vertex.

The DBSCAN algorithm runs as follow:

Algorithm 1 DBSCAN algorithm

1: **procedure** DBSCAN(*data*, *eps*, *minPts*)
2: Build the graph G from *data*, *eps* and *minPts*.
3: **for** all core vertices u in G **do**
4: **if** u is not assigned to any cluster yet **then**
5: Assign all vertices v where v is not assigned to any cluster and there is a path from u to v in G, to be in the same cluster as u.

In theory, this algorithm runs in $O(n^2)$ where n is the number of data points, because the assignment process in line 5 is a graph traversal algorithm such as DFS or BFS which takes $O(n)$ time. However, by choosing appropriate values for *eps* and *minPts*, the overall running time can be reduced to $O(n \log(n))$ with the help of a spatial access method such as R-tree, since the running time depends directly on the size of a vertex's neighborhood. If the size of a vertex's neighborhood is always not greater than a constant, the complexity for such a DBSCAN algorithm will be exactly $O(n \log(n))$.

In the paper, the authors suggest that *eps* can be chosen from the k-dist line chart of the initial data where $k = minPts$, and *minPts* should be 4 for 2-dimensional data. The k-dist line chart is constructed as follows:

- Build a map $D : V \to \mathbb{R}$, which maps any vertex u to the distance between u and its k-th nearest neighbor (excluding u).
- Sort all vertices u by decreasing value of $D(u)$.
- Draw a line chart where horizontal axis is the sorted vertices and vertical axis is the value of D applies to a vertex.

Then *eps* is set as the first value of D where the line in the chart starts to decrease abruptly slower (in other words, the first "valley", as called by the authors).

Advantages / disadvantages of DBSCAN algorithm

- Advantages:
  - No need to pre-define the number k of clusters.
  - Not very sensitive to outliers.
  - Can find a cluster of any shape.
  - Quick running time ($O(n \log(n))$) if parameters are chosen appropriately.
- Disadvantages:
  - In rare cases, running the algorithm twice will not produce the same output.
  - Depends largely on the distance function. In many cases, Euclidean distance is not suitable due to the "curse of dimensionality".
  - Data sets with clusters having large differences in density are not handled well, since the parameters cannot be chosen appropriately for all clusters.
  - Need a domain expert who knows the data well to decide good values for eps and $minPts$.