Density Based Methods

✧ Clustering based on density (local cluster criterion), such as density-connected points

✧ Major features:
  ✧ Discover clusters of arbitrary shapes
  ✧ Handle noise
  ✧ Need density parameters as termination condition
Density Based Methods

✧ Main Concepts:

✧ parameters:

✧ Eps: Maximum radius of the neighborhood

✧ MinPts: Minimum number of points in an Eps-neighbourhood of that point

✧ Sample q is directly density-reachable from sample p, if \( d(p, q) \leq \text{Eps} \) and p has MinPts points in its neighborhood.

✧ Sample q is density-reachable from a sample p if there is a chain of points \( p_1, \ldots, p_n \), \( p_1 = p \), \( p_n = q \) such that \( p_{i+1} \) is directly density-reachable from \( p_i \).

✧ Sample p is density-connected to sample q if there is a sample o such that both, p and q are density-reachable from o.
Density Based Methods: DBSCAN

- **DBSCAN** (Density Based Spatial Clustering of Applications with Noise)
  - Relies on a *density-based* notion of cluster: A *cluster* is defined as a maximal set of density-connected points
  - Discovers clusters of arbitrary shape in spatial data with noise

- **Algorithm DBSCAN** *(Eps, MinPts)*
  - Arbitrary select a sample *p*
  - Retrieve all samples density-reachable from *p* w.r.t. *Eps* and *MinPts*.
  - If *p* is a core sample (some samples are density-reachable from *p*), a cluster is formed.
  - If *p* is a border sample (no samples are density-reachable from *p*), DBSCAN visits the next sample of the database.
  - Continue the process until all of the samples have been processed.
Graph-based Clustering

- Represent data points as the vertices $V$ of a graph $G$.
- All pairs of vertices are connected by an edge $E$.
- Edges have weights $W$.
  - Large weights mean that the adjacent vertices are very similar; small weights imply dissimilarity.
Graph-based Clustering

- Clustering on a graph is equivalent to partitioning the vertices of the graph.
- A loss function for a partition of $V$ into sets $A$ and $B$
  \[
  \text{cut}(A, B) = \sum_{u \in A, v \in B} W_{u,v}
  \]

- In a good partition, vertices in different partitions will be dissimilar.
  - Mincut criterion: Find a partition $A, B$ that minimizes $\text{cut}(A, B)$
  - Mincut criterion ignores the size of the subgraphs formed
Graph-based Clustering

✧ Normalized cut criterion favors balanced partitions.

\[
N\text{cut}(A, B) = \frac{\text{cut}(A, B)}{\sum_{u \in A, v \in V} W_{u,v}} + \frac{\text{cut}(A, B)}{\sum_{u \in B, v \in V} W_{u,v}}
\]

✧ Minimizing the normalized cut criterion exactly is NP-hard.

✧ One way of approximately optimizing the normalized cut criterion leads to \textit{spectral clustering}. 
Spectral Clustering

* Spectral clustering
  * Looks for a new representation of the original data points, such that
    * Preserve the edge weights.
    * The convex clusters’ shapes in the new space represents non-convex ones in the original space.
  * Cluster the points in the new space using any clustering scheme (say k-means).

* We only describe the resulting algorithm here.
  * For more information about derivations, refer to *U. Luxburg, “A Tutorial on Spectral Clustering”*. 