Lecture 24

Name: Harsh Sanjay Roniyar

Roll Number: 22B3942

`A4 Released - Due after EndSem`

---

k-means initialization

```
Initialize c_1 to c_k randomly
Loop
{
        y_i <- arg min_k ||x_i - c_k||**2, for all i (Cluster Assignments)
        c_k <- (sum_i 1{y_i = k}*x_i)/(sum_i 1{y_i = k}), for all k
(Calculate Centroids)
}
```

`1{}` is the indicator function.

Initialization of cluster centers

1. Pick c_1 randomly from X itself
2. Pick c_2 from X itself that is furthest from c_1
3. Pick c_3 that is furthest from c_1 and c_2.
4. $\dots$

Fuzzy c-means

In k-means, X_i and c_j's are mapped by hard partitioning, that is x_i can belong to only one class. i.e. $w_{ij} \in \{0, 1\}$

$$w_{ij} = \mathbb{1}\{d(x_i, c_j)\} = \min_k [d(x_i, c_j)]$$

In fuzzy c-means, x_i's can belong to every class with some probability, hence soft partitioning, i.e. $w_{ij} \in [0, 1]$

$$w_{ij} = \frac{1}{\sum_{k=1}^{c} \left[\frac{\|x_i - c_j\|}{\|x_i - c_k\|}\right]^{\frac{2}{m-1}}}$$

$\lim m \to 1$ will make the weights approach k-means.
Higher the hyperparameter value of m, higher the fuzziness.

In fuzzy c-means, the cluster centers will be a weighted mean.

$$c_j = \frac{\sum_i w_{ij} x_i}{\sum_i w_{ij}}$$

Problem

With both k-means and fuzzy c-means, both depend on Euclidean distance from cluster centers. That means, both prefer hyper-spherical clusters (isotropic) of equal radii

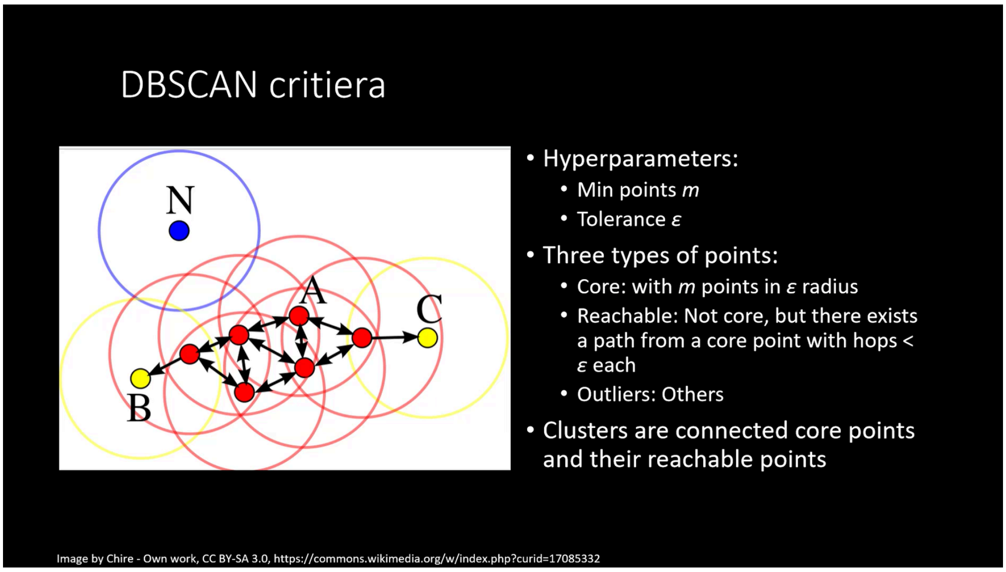
Density-based clustering

An example of which is `DB-SCAN`.

DB-SCAN

3 types of data points, clustered into

- High-density -> cores of arbitrary shapes
- Medium-density -> peripheries
- Low-density -> outlier regions



ϵ radius (`tolerance`) and m minimum points

Steps:

1. Form a graph with ϵ neighbor hard (`tolerance region`)
2. Identify core points (number of neighbors $\geq$ m)
3. Identify periphery/reachable points (non-core points, but at least one core-point neighbor, i.e. within ϵ reach)
4. Other points become the outliers

So, now we have two hyper-parameters in this problem

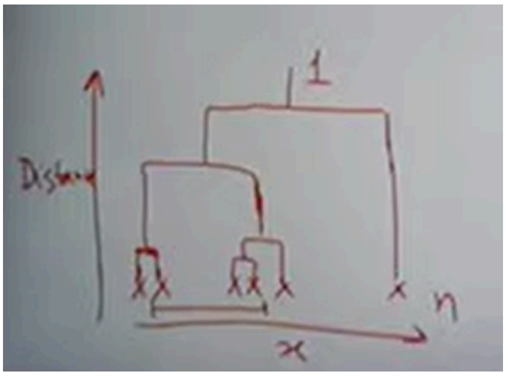
DB-SCAN Algorithm

- For each sample x_i
  - For each other sample x_j
    - Mark j as neighbor of i, if $d_{ij} < \epsilon$
    - Increment number of neighbors n_i of x_i
- For each sample x_i
  - If neighbors $\geq m$ then mark as **core**
  - If neighbors > 0 then mark as **reachable**
  - If neighbors $= 0$, then mark as **outlier**
- For each sample x_i
  - If core and un-clustered, then mark all connected samples with this cluster

Hierarchical Clustering

- Start with each sample as its singleton cluster
- For each merge iteration
  - For each cluster
    - For each other cluster
      - Compute inter-cluster distance
  - Merge two closest clusters

We get a resulting dendo-gram similar to the following



Now, what is inter-cluster distance?
Choices:

1. Distance between nearest points from the clusters: `Single-Linkage`
2. Distance between furthest: `Complete-Linkage`
3. Distance between centroids: `Average-Linkage`

Clustering Metrics

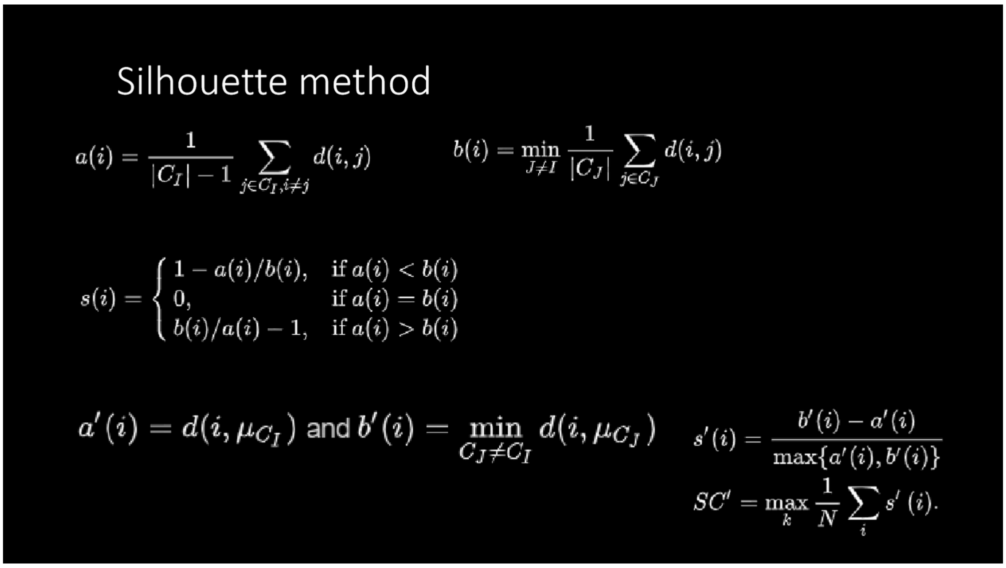
Variation explained
Low intra-cluster variation
High inter-cluster variation

There are multiple methods for such analyses, e.g.

- Elbow method (Variation explained vs No. of clusters) - We observe an elbow at 10% from 100% variation explained, and pertaining to that we get a good number of clusters from the method.
- Silhouette method

How would we measure the variation:

- Avg. distance from centroid
- Avg. distance from all other points
- Fit an isotropic gaussian
  These methods won't work for DB-SCAN

- **Silhouette** method:
  

  $a(i)$ is the intra-cluster parameter
  $b(i)$ is the inter-cluster parameter
  $S = \underset{i}{\text{mean}}\ s(i)$
- **Davies-Bouldin** index:
  minimize ratio of intra-cluster versus inter-cluster variation

$$\underset{i}{\text{mean}}\ \max_j \frac{S_i + S_j}{M_{ij}}$$

S_i is variation in cluster i and M_{ij} is the variation in cluster i and j combined.