① K-means. clustering
② hierarchical clustering (computational intensive for large datasets)

NOTICE: Data Normalization maybe Required if attributes ranges very significantly.

Steps for K-means:

① define # of K.
② randomly assign obs to cluster centers $C_1, C_2, \cdots C_K$.
③ calculate the centroids (mean) of cluster centers $\mu_1, \mu_2, \cdots \mu_K$.
④ calculate distance between each observation $x_1, x_2, \cdots x_n$ and cluster centroids $\mu_1, \mu_2, \cdots \mu_K$.

$$d(x_i, \mu_j) = (x_1 - x_2)^2 + (y_1 - y_2)^2 + \cdots \quad \text{or} \quad \sqrt{\cdots}$$

(to reduce computation without sqrt)

(sum of squared error)
⑤ calculate $SSE_T = SSE_1 + SSE_2 + \cdots + SSE_K$

⑥ iterate (repeat) above until SSE_T dose not decrease.

(choosing randomly initial points (centroids) are crucial for K-means)

Steps for Hierarchical clustering.

① calculate point-wise distances. (Euclidan / manhattan)
squared euclidian (maybe this in exam)
euclidian.
generate proximity Matrix.

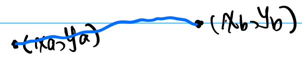
② fuse (merge) the closet. two.

③ calculate distance again and update matrix. (either point-to-point ← euclidant/manhattan
or point-to-clusster ← linkage
cluster-to-cluster ← linkage

④ fuse the closet two:

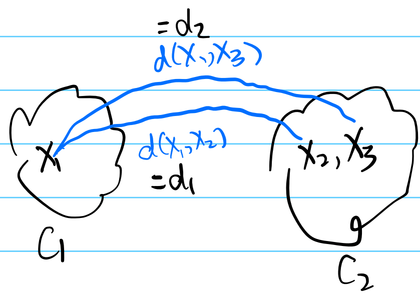
⑤ repeat until we reach the root of dendrogram.

how to calculate distances



- squared euc : $(x_a - x_b)^2 + (y_a - y_b)^2$
- euc : $\sqrt{(x_a - x_b)^2 + (y_a - y_b)^2}$
- manhattan : $|x_a - x_b| + |y_a - y_b|$

how to calculate point/cluster-to-cluster linkage?:



$d(C_1, C_2) =$

- single : $\min(d_1, d_2)$
- complete : $\max(d_1, d_2)$
- average : $\frac{d_1 + d_2}{2}$
- centroid : $d(\mu_1, \mu_2)$

(Single)

min
- △ : can handle non-elliptical shapes
- ▽ : sensitive to noise and outliers

(Complete)

max.
- △ : less sensitive to noise and outliers ①
- ▽ :
  - tends to break large clusters
  - biased towards globular clusters ②

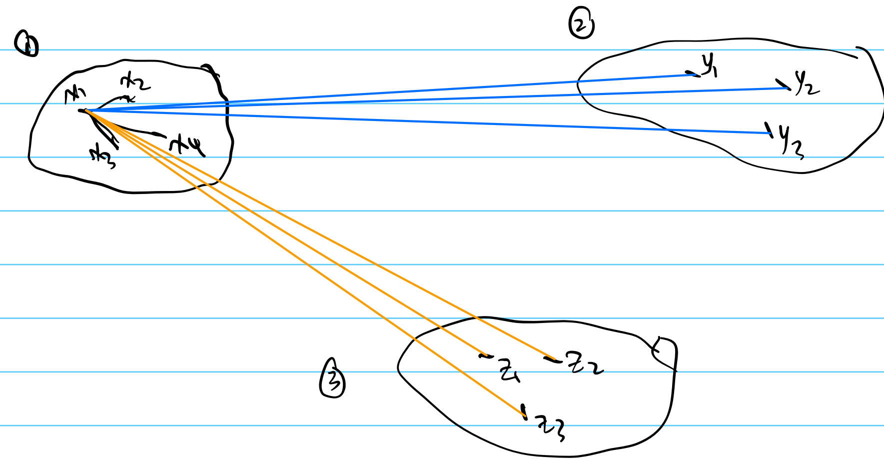
average
- △ : same as ① (noise-outliers)
- ▽ : same as ② (biased)

k-means vs hierarchical ?

k-means :
- △ fast.
- ▽ because it uses mean. easy to be affected by the outliers

How we determind the best cluster?

(by using cluster index — silhouette index)



(a-value)

a1 = average distance from point 1 to other points at the same cluster.

$$a_1 = avg.(d(x_1, x_2), d(x_1, x_3), d(x_1, x_4))$$

(b-value)

b1 = minimum of average distance from point 1 to other clussters

① calc avg distance from x_1 to other clusters.

② took the minimum.

$$d(x_1, ②) = \frac{d(x_1, y_1) + d(x_1, y_2) + d(x_1, y_3)}{3}$$

$$d(x_1, ③) = \frac{d(x_1, z_1) + d(x_1, z_2) + d(x_1, z_3)}{3}$$

$$b_1 = \min(d(x_1, ②), d(x_1, ③))$$

(s-value)

$$S_1 = \frac{(b_1 - a_1)}{\max(b_1, a_1)} \text{ . generally } b_1 > a_1$$

$$\therefore \rightarrow S_1 = \frac{b_1 - a_1}{b_1}$$

if within cluster distance is very small $\rightarrow a_1 \sim 0$
then $S_1 \rightarrow 1$. (very good!)

if within cluster distance is almost same as inter-cluster $\rightarrow a_1 \approx b_1$
then $S_1 \rightarrow 0$. (very bad, means that data point actually belongs to another cluster)

if a value larger than b, then $\max(b_1, a_1) \rightarrow a_1$
in extreme, $b_1 - a_1 \rightarrow -a_1$.
$\therefore S_1 \rightarrow -1$.

$\therefore$ in general: $-1 \leq S_{value} \leq 1$

① we calculate s-value for each point

② we calculate s-value for each cluster $S_K = \frac{1}{n_K} \sum S_i$ — n item in k.

③ we calculate s-value for the clustering model $I = \frac{1}{K} \sum S_K$ — total # of clusters

(sihoutte Index)

close to '1' is better.