DMML, 11 Feb 2020

Clustering – K Means algorithm

Randomly choose K centroids

for each item i

assign to nearest centroid

Recompute centroids

Till centroids stabilize

Efficiency ?

Incrementally adjust centroid with each new data item

If mean of n values is a, after new value, mean is $(n*a + v_{n+1}) / n+1$

What is a good choice for K?

Measuring quality of clustering

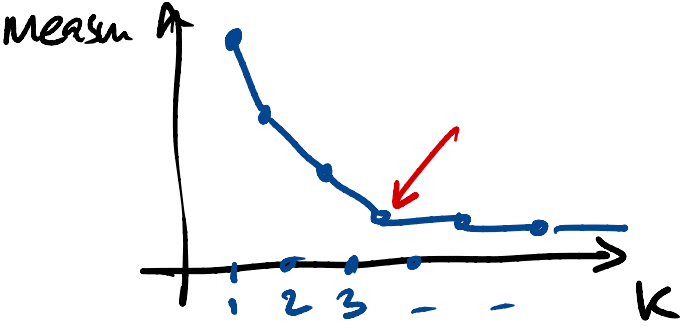
Obvious measure - dispersion of clusters

Mean radius wrt centroid

Variance

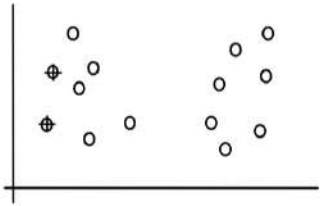
As K increases, any reasonable measure will improve

Typically, graph has "elbow" (or "knee")

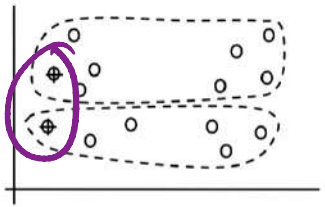


Problem of random starting centroids

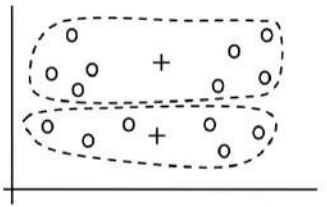
- Brute force – repeat the process & check for variation
- Systematically choose random points far apart



(A). Random selection of seeds (centroids)



(B). Iteration 1



(C). Iteration 2

Having chosen j centroids so far

- choose next point proportional to distance from nearest centroid

Claim: Cost of this offset by quality improvement

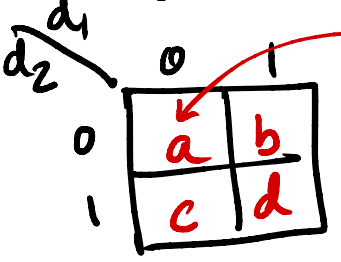
What about non-numeric attributes ?

Make everything boolean – 1-hot encoding

	Red	Blue	Green	Black
Recall our set of words	1	0	0	0
document model	0	0	1	0

Each document is over $\{0,1\}^{|V|}$

Similarity – no. of positions where documents agree



# of positions where $d_1 = d_2 = 0$

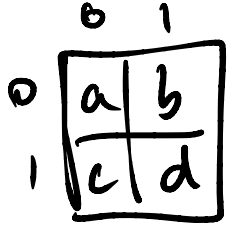
$$\frac{a+d}{a+b+c+d} \rightsquigarrow 1 \text{ if } d_1 = d_2$$

Problem is again asymmetry

English $|V| \sim 10^5$

Newspaper article ≤ 1000 words

Almost all entries are 0



a dominates by a large margin

Ignore a

Define similarity as $\frac{d}{b+c+d}$

Jaccard ~~distance~~ / similarity

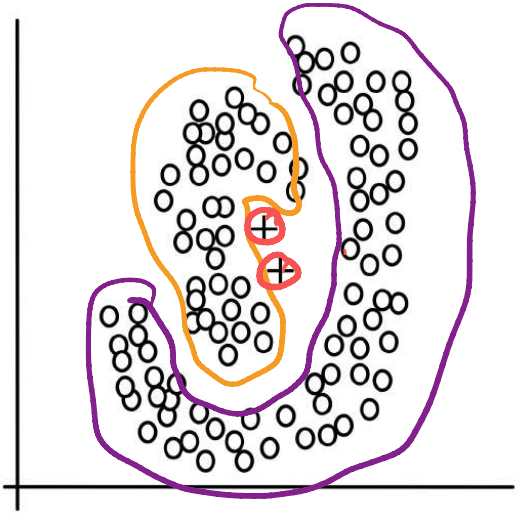
$$\sqrt{\Delta x_1^2 + \Delta x_2^2 + \cdots + \Delta x_n^2}$$

↓ wt in gm ↓ height in cm

Be careful about scales across attribute

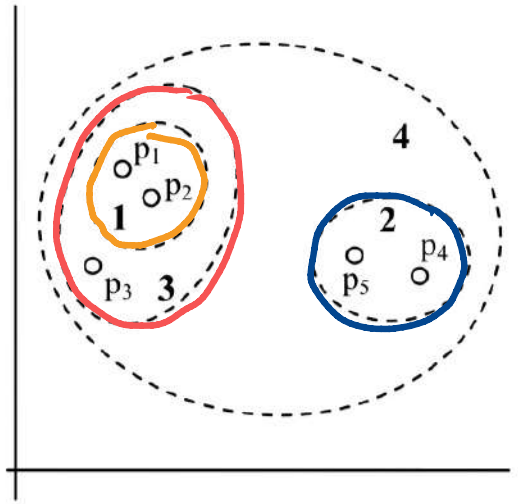
Solution – Normalize

K-Means can only discover ellipsoidal clusters

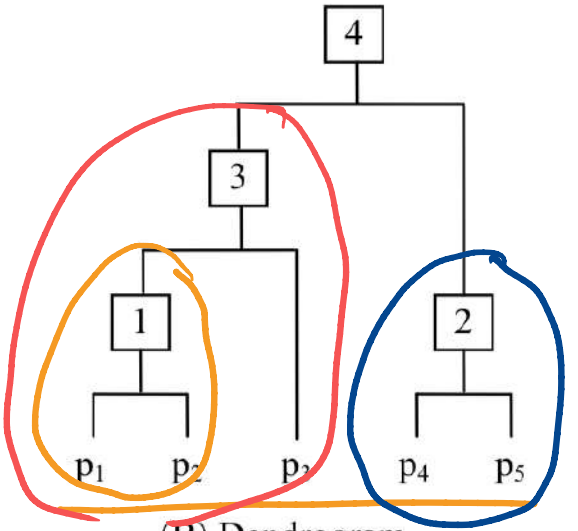


Bottom Up Clustering

Keeping combining nearest clusters to form a larger one



(A). Nested clusters

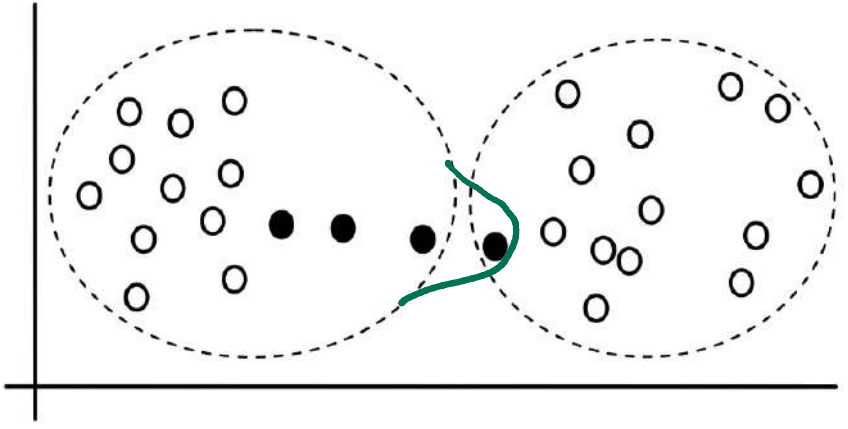


(B) Dendrogram

Choose the level that is best

Main challenge — how to define inter-cluster distance

complexity



All of these essentially involve a computation of $|S_1| \times |S_2|$

Two sets of points S_1, S_2

Single Link

- Nearest pair across S_1 & S_2

Complete Link

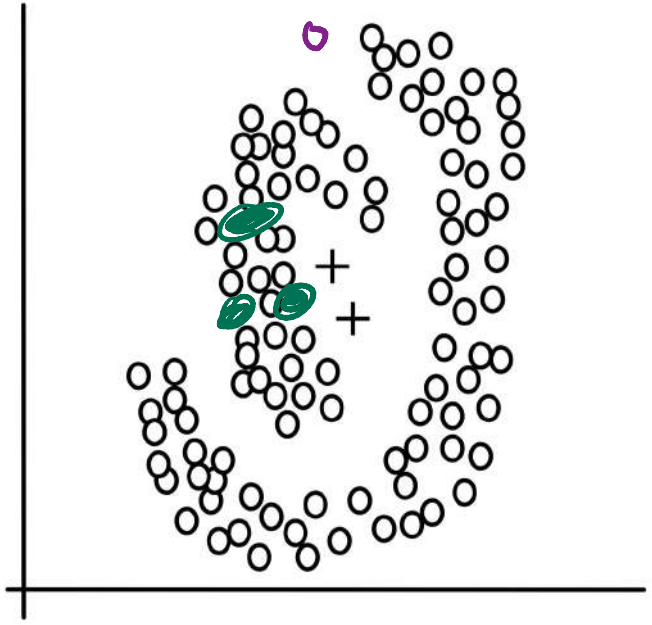
- Max pairwise distance

Average Link

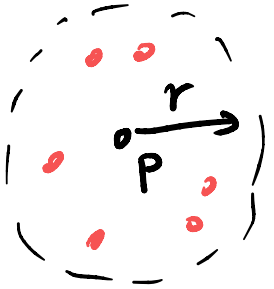
- Mean pairwise distance

Density based approach

Grow clusters based on similar dispersion



Define local density of a point



Fix r

Count nbrs in r-neighbourhood

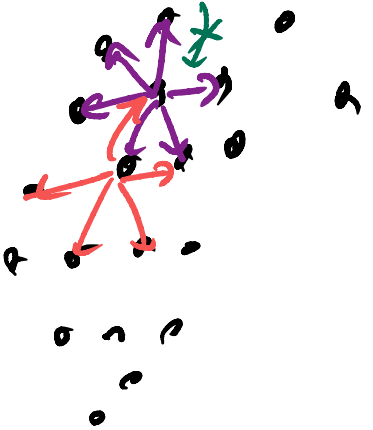
Define a threshold t

p is "dense" if nbd has $\geq t$ pts

"Core points"

Start with core point

Connect each core point with a directed edge to its r-nbd points



Throw away direction

Connected components are clusters