Clustering algorithms: Perspective and first example

Unsupervised Machine Learning

Alex Rodriguez.
arodrigu@ictp.it
ICTP, Leonardo Building
Office 265
+39 040 2240 369

→ Real world datasets have very diff. features

→ For proteins diff. sequences lead to diff. features for each data point

→ Clustering puts together similar data

→ Clustering ≠ Classification

Clustering → only groups

Classification → groups + labels

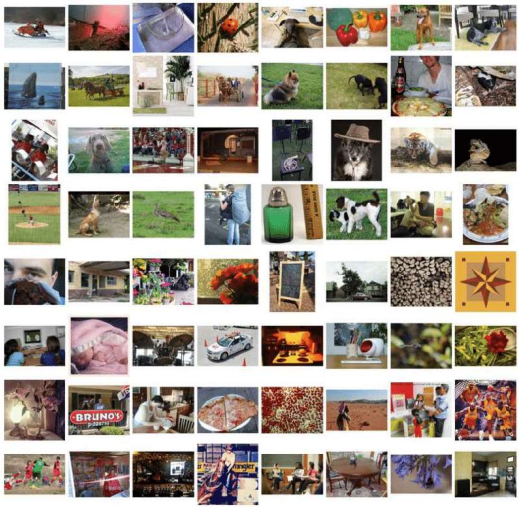
NO labels

→ Feature selection → Dim. reduction

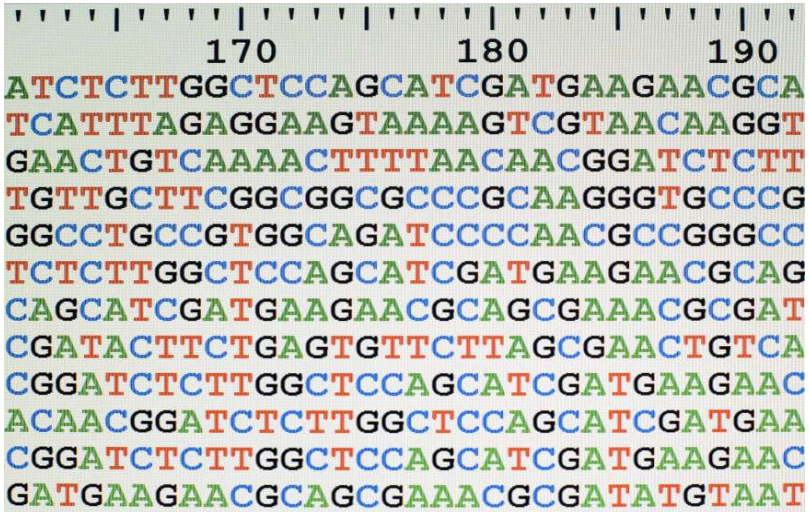
Try to select relevant features that are good for reproducing ground truth.

TWO WORDS ABOUT REAL WORLD DATA AND CLUSTERING

Some «real world» data sets



ImageNet



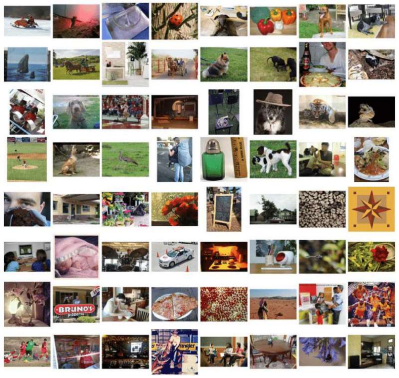
Genes

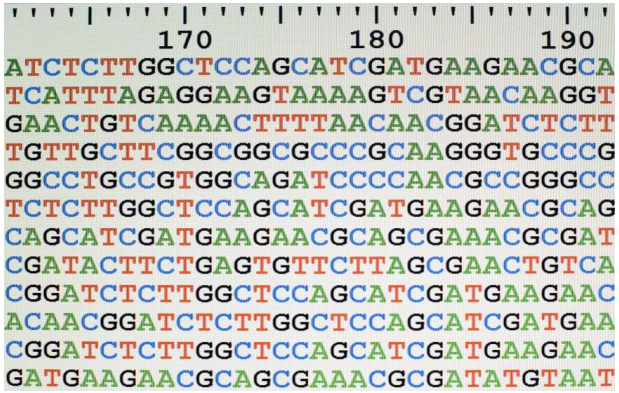
Feature Name	Feature ID
Insured Sex	0
Insured Occupation	1
Insured Hobbies	2
Capital Gains	3
Capital Loss	4
Incident Type	5
Collision Type	6
Incident Severity	7
Authorities Contacted	8
Incident Hour of the Day	9
Number of Vehicles Involved	10
Witnesses	11
Total Claim Amount	12
Age Group	13
Months as Customer	14
Annual Premium	15

TABLE 1: FEATURE LIST

Insurance

Some «real world» data sets



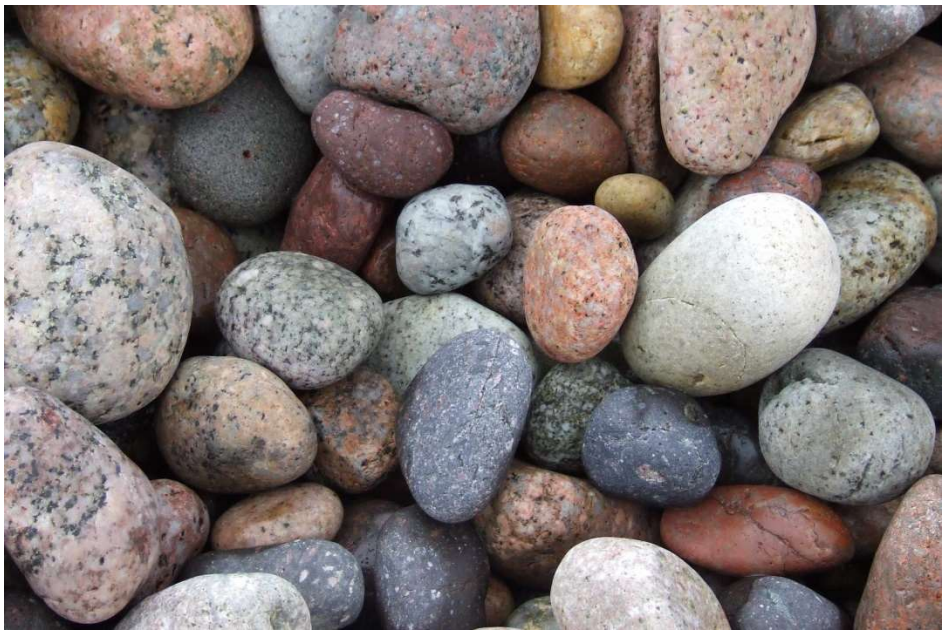


Feature Name	Feature ID
Insured Sex	0
Insured Occupation	1
Insured Hobbies	2
Capital Gains	3
Capital Loss	4
Incident Type	5
Collision Type	6
Incident Severity	7
Authorities Contacted	8
Incident Hour of the Day	9
Number of Vehicles Involved	10
Witnesses	11
Total Claim Amount	12
Age Group	13
Months as Customer	14
Annual Premium	15

TABLE 1: FEATURE LIST

- Complex to encode
- Not always the same number of features at each data point
- Different kinds of features

My stone collection



- Weight
- Light wavelength
- Shape
- Volume
- Rugosity
- …

Characteristics of the data sample

- Raw characteristics:
  - Number of features (Dimension)
  - Number of samples (Cardinality)
  - Type of features (Reals, integers, binary, qualitative)
- Learned characteristics:
  - Statistics (Variances, covariances, averages…)
  - Intrinsic dimension.
  - Probability density.
  - Clustering…

My stone collection

	Integer		Categorical		Real
Number	Weight (g)	Wavelenght (nm	Shape	Volume	Rugosity
1	20	450	spherical	0,2	1,04
2	33	698	cube	0,4	1,3
3	12	543	cube	0,2	0,8
4	70	691	spherical	1,1	1,9
5	120	465	cube	0,3	0,2
6	45	486	cube	0,8	0,3
7	136	504	cube	1,35	0,5
8	20	504	spherical	0,2	0,5
9	54	623	spherical	0,5	1
10	93	430	spherical	0,7	1
....					

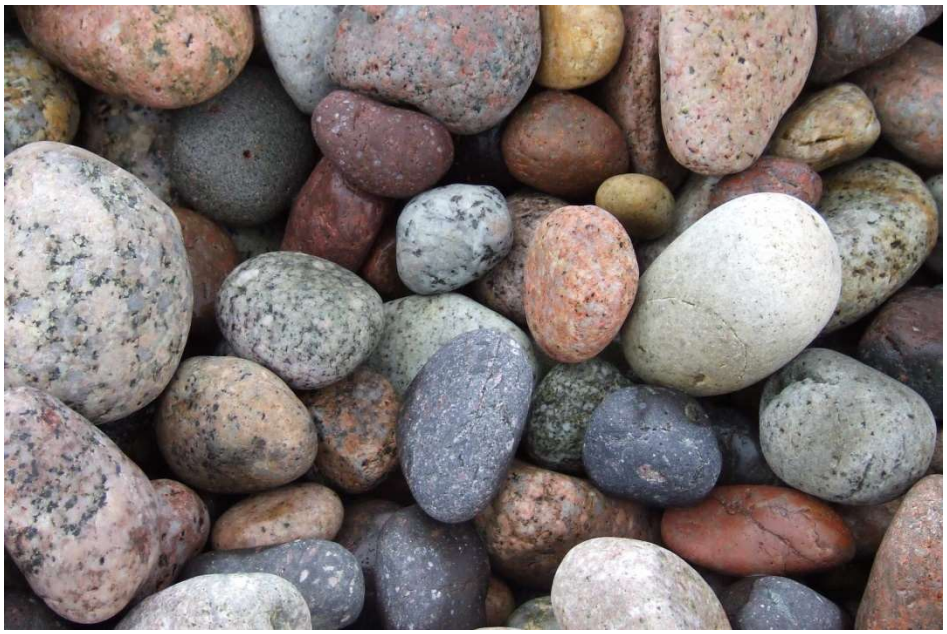
Why Clustering?

"Understanding our world requires conceptualizing the similarities and differences between the entities that compose it" (Tyron and Bailey, 1970).

- Clustering ≠ Classification
  - Clustering generate groups
  - Classification generate groups <u>& labels</u>

My stone collection (2)

Cluster by light wavelength



Are they nice or ugly?
Color classification?

Feature Selection

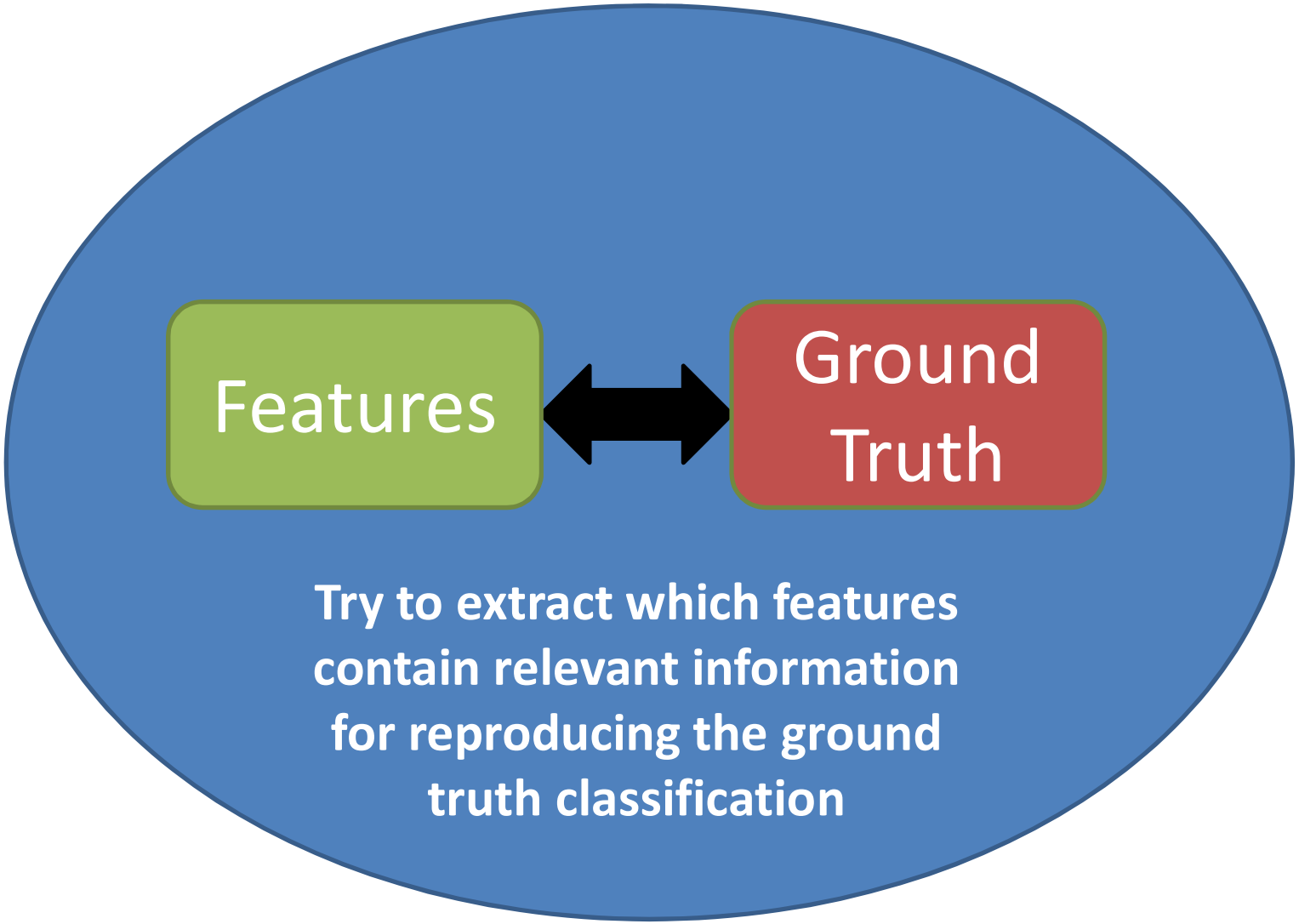
Supervised

Unsupervised Machine Learning

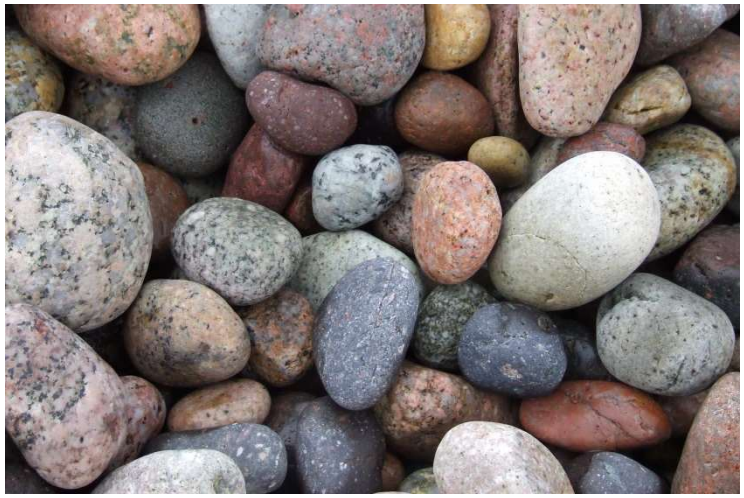
Feature Selection & Dimensionality reduction

- We often need to reduce the number of features due to the curse of dimensionality.
- Highly related with the problem that we want to solve.
- It may need feedback from the whole process.
- Feature selection depends on labeled data while dimensionality reduction does not.
- Feature selection can be based on expertise…

Feature selection



My stone collection (3)



- Weight
- Light wavelength
- Shape
- Volume
- Rugosity
- …

Of course, if you are an expert, you already know which are the relevant features… but, I´m quite dummy…

Techniques for Feature Selection (1) Variable Ranking

- Which variables explain better the labels? Quantify it.

Techniques for Feature Selection (1) Variable Ranking

- Which variables explain better the labels? Quantify it.
  - Linear correlation coefficient (R).

Techniques for Feature Selection (1)
Linear Correlation coefficient

$$R(i) = \frac{\sum_{k=1}^{m}\left(x_{k,i} - \overline{x_i}\right)(y_k - \bar{y})}{\sqrt{\sum_{k=1}^{m}\left(x_{k,i} - \overline{x_i}\right)^2 \sum_{k=1}^{m}(y_k - \bar{y})^2}}$$

- For continuous variables and outputs.
- Goodness of linear fit
- Easy to extend to linear fit of functions of variables (i.e. take log of x).

Doesn't work for non-continuous func's.

→ see if x & y are correlated linearly with output.

Techniques for Feature Selection (1) Variable Ranking

- Which variables explain better the labels? Quantify it.
  - Linear correlation coefficient (R).
  - Single variable classifier. Jaccard index, F-score, etc.

based on a confusion matrix (next page)

Techniques for Feature Selection (1)
Single variable classifier

→ val. of features

Confusion matrix

R ↓ C→	Forest	Indust.	Urban	Water	Total
Forest	**68**	7	3	0	**78**
Indust.	12	**112**	15	10	**149**
Urban	3	9	**89**	0	**101**
Water	0	2	5	**56**	**63**
Total	**83**	**130**	**112**	**66**	**391**

val. of ground truths

Columns → values of features

Rows → " " ground truths.

- Based in the correspondence between the ground truth classification and the ones that comes from the single variable
- In some cases, requires labeling (assign the variable to a class).
- The confusion matrix can summarize some of them

→ obtain vars. that explain the ground truth (last lec).

→ you compare joint distrib^n btw^n 2 vars & ground truth.

Techniques for Feature Selection (1) Variable Ranking

- Which variables explain better the labels? Quantify it.
  - Linear correlation coefficient (R).
  - Single variable classifier. Jaccard index, F-score, etc.
  - Mutual information between variable and the target.

Techniques for Feature Selection (1) Information Theoretic Ranking

$$I(i) = \int_{x_i} \int_{y} p(x_i, y) \log \frac{p(x_i, y)}{p(x_i)p(y)}$$

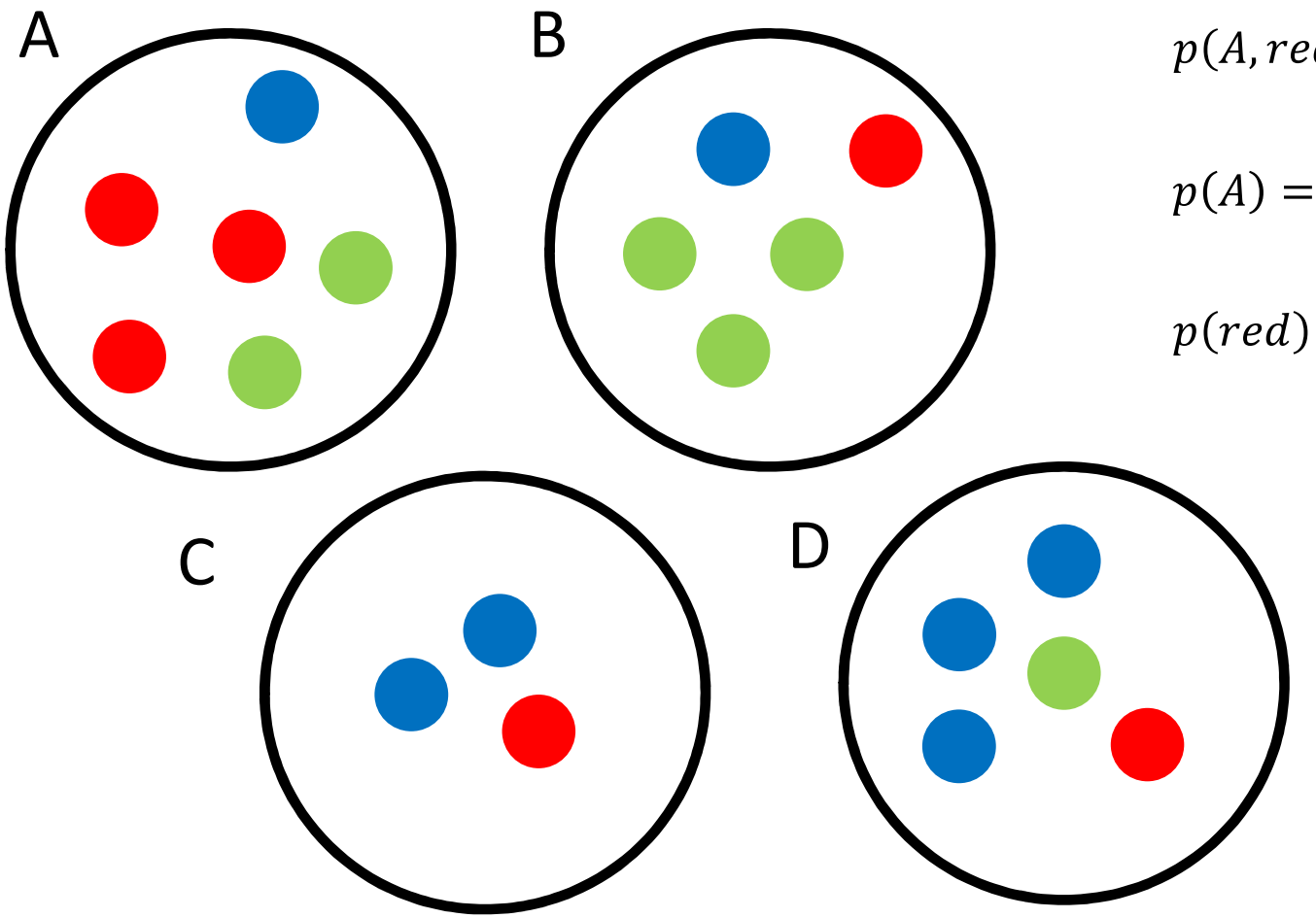
- A kind of single variable classifier.
- In non-continuous variables, the integrals become sums
- Extensible to continuous variables by non parametric density estimation
- Using a Gaussian distribution for estimating the density will lead to a similar criteria to the correlation coefficient.
- Is a formalization of the intuition that the higher the joint distribution, the higher the mutual information, i.e. the higher should it be in the rank.

→ Compute Mutual Inf. for a discrete case

Techniques for Feature Selection (1) Information Theoretic Ranking

- The probabilities, in a discrete case are estimated from frequency counts.
- Imagine a three class problem (red, green, blue) with a discrete variable that can take 4 values (A,B,C,D).
  - P(y) are 3 frequency counts.
  - P(x) are 4 frequency counts.
  - P(x,y) are 12 frequency counts.

Techniques for Feature Selection (1)
Information Theoretic Ranking



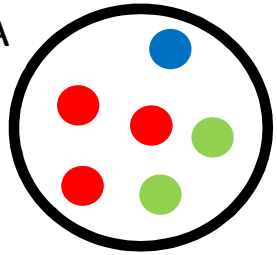
$$p(A, red) = \frac{3}{19}$$

$$p(A) = \frac{6}{19}$$

$$p(red) = \frac{6}{19}$$

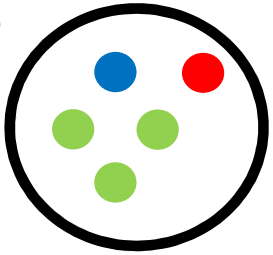
Techniques for Feature Selection (1)
Information Theoretic Ranking

A



$$I = \frac{3}{19}\log\left(\frac{{}^{3}/_{19}}{{}^{6}/_{19}{}^{6}/_{19}}\right) + \frac{1}{19}\log\left(\frac{{}^{1}/_{19}}{{}^{6}/_{19}{}^{7}/_{19}}\right)$$

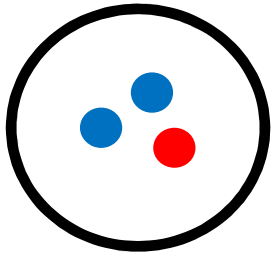
B



$$+ \frac{2}{19}\log\left(\frac{{}^{2}/_{19}}{{}^{6}/_{19}{}^{6}/_{19}}\right) + \frac{1}{19}\log\left(\frac{{}^{1}/_{19}}{{}^{5}/_{19}{}^{6}/_{19}}\right) +$$

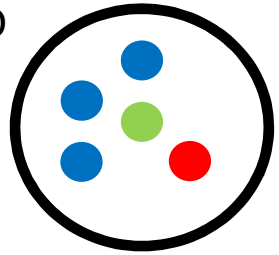
$$\frac{1}{19}\log\left(\frac{{}^{1}/_{19}}{{}^{5}/_{19}{}^{7}/_{19}}\right) + \frac{3}{19}\log\left(\frac{{}^{3}/_{19}}{{}^{5}/_{19}{}^{6}/_{19}}\right) +$$

C



$$\frac{1}{19}\log\left(\frac{{}^{1}/_{19}}{{}^{3}/_{19}{}^{6}/_{19}}\right) + \frac{0}{19}\log\left(\frac{{}^{0}/_{19}}{{}^{3}/_{19}{}^{7}/_{19}}\right) +$$

D



$$\frac{2}{19}\log\left(\frac{{}^{2}/_{19}}{{}^{3}/_{19}{}^{6}/_{19}}\right) + \frac{1}{19}\log\left(\frac{{}^{1}/_{19}}{{}^{5}/_{19}{}^{6}/_{19}}\right) +$$

$$\frac{1}{19}\log\left(\frac{{}^{1}/_{19}}{{}^{5}/_{19}{}^{7}/_{19}}\right) + \frac{3}{19}\log\left(\frac{{}^{3}/_{19}}{{}^{5}/_{19}{}^{6}/_{19}}\right) \approx 0.17$$

Techniques for Feature Selection (1)

Some questions about Variable Ranking

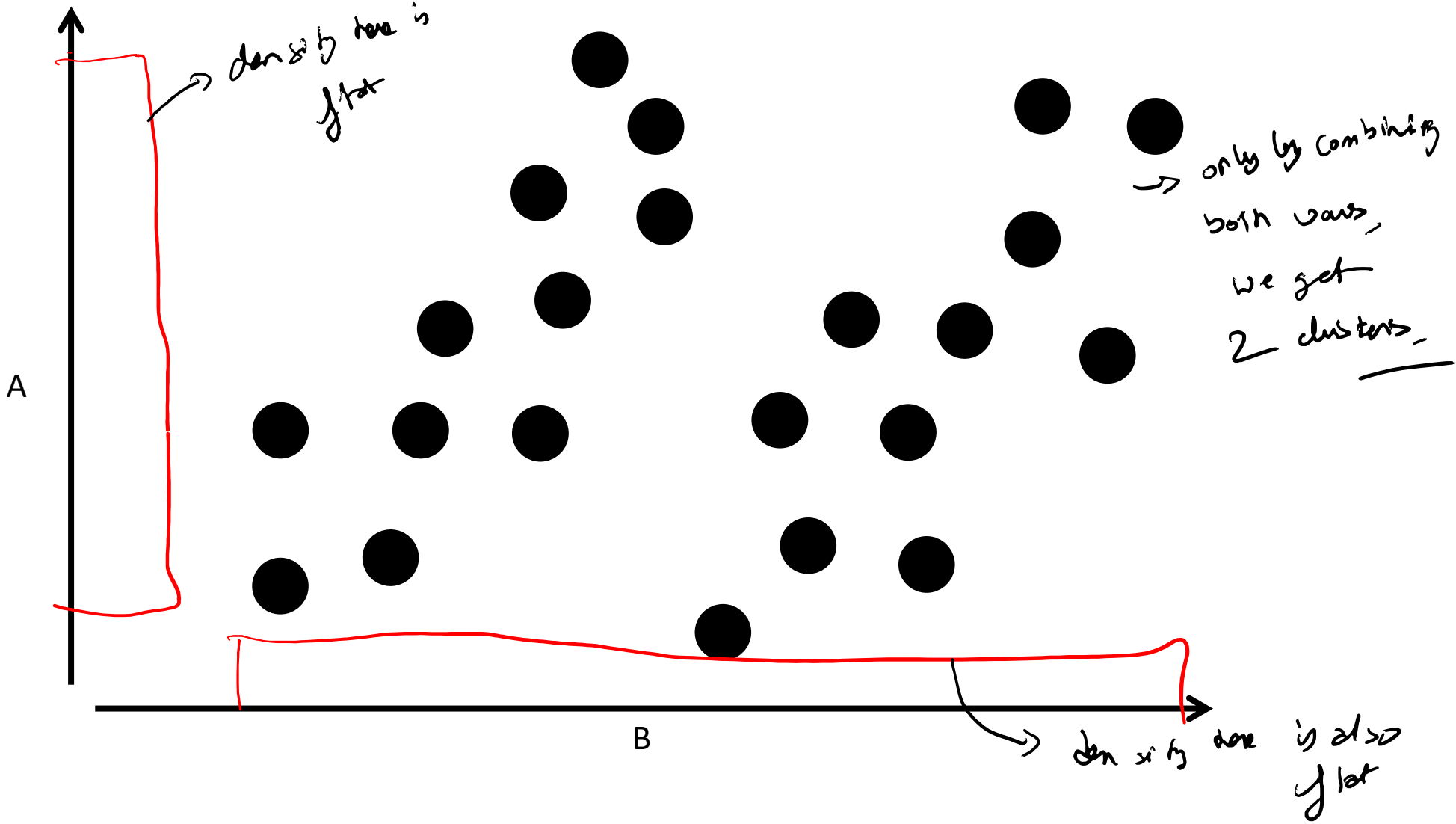
- How to treat redundant variables?
  - Redundant variables get the same information but its combination can lead to noise reduction.
  - Correlation is a measure of redudancy
- A variable useless by itself can be useful together with others → eg on next page

→ It can happen that 2 vars. are redundant & are subject to independent noise, & their combination can lead to reduction in noise of the data. So, sometimes maintaining redundant vars. is imp.

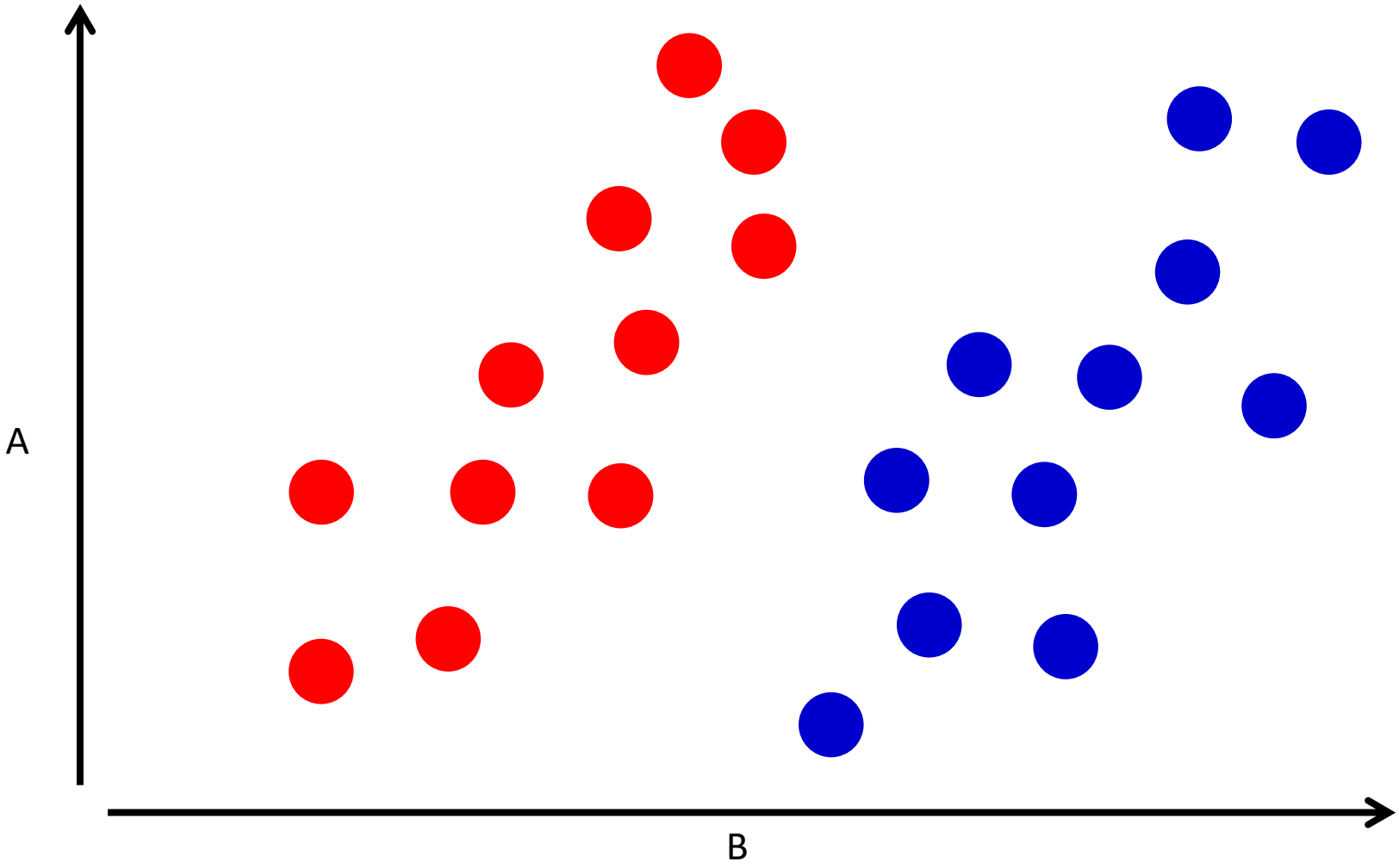
M.I. btw: A & A (ground truth) ~ low
" " B " B (" ") ~ "
→ combine A+B
high M.I.
2 clusters

Features useful only in combination

If I look only to one of the variables I would not be able to separate this data.



Features useful only in combination: which combination of A and B will be useful for separate the two clusters?



My stone collection (4)

- Weight
- Light wavelength
- Shape
- Volume
- Rugosity
- …

Will it recover the density as an important feature for mineral recognition?

Density=Weight/Volume

Techniques for Feature Selection (2)

Subset selection

- Wrappers: Use the predicting power of a given learning machine to assess the usefulness of a given subset

  ① – How to search the space? (Brute force is NP hard.)

  ② – How to assess the performance of the prediction.

  ③ – Which learning machine to use.

Similarities and distances

Unsupervised Machine Learning

Why working with distances?

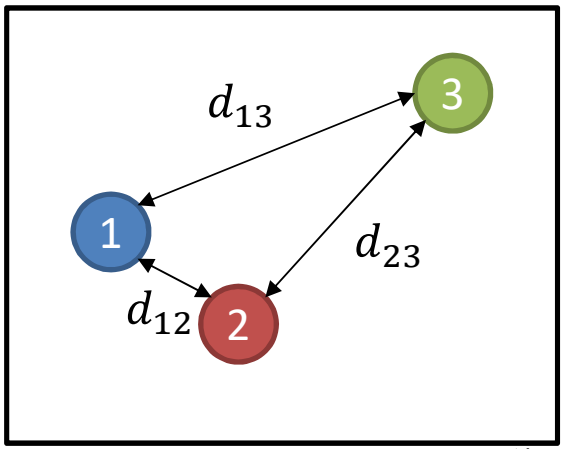
My data:

1 AACDPGGGAD

2 CCDPGGADA

3 CCCCCCCCCC

?



Look at them pairwise

1 AACDPGGGAD*

2 *CCDP*GGADA

$S_{12} = \frac{7}{11}$

2 CCDPGGADA*

3 CCCCCCCCCC

$S_{23} = \frac{2}{10}$

1 AACDPGGGAD

3 CCCCCCCCCC*

$S_{13} = \frac{1}{10}$

$d_{ij} = 1 - S_{ij}$

- Similarities and distances are much more flexible than simple feature representation.
- Each type of data has its own optimal distance.

Similarity and Distances

- Clustering tries to separate data "*naturally*", in such a way that *similar* elements lay in the same cluster while *dissimilar* elements belong to a different one
- *Similarity* (S_{ij}) is a pairwise function of the features of the elements i and j.
- In terms of space, it can be thought that similar elements are near while dissimilar are far. So many times it is useful to talk about "distances between elements" (D_{ij})
- Its definition depends on the nature of the features

Similarity and Distances almost the same but…

A (metric) distance must accomplish: (Properties of a metric distance)

1. Symmetry: $d(x, y) = d(y, x)$
2. Non-negativity: $d(x, y) \geq 0$
3. Identity of indiscernibles: $d(x, y) = 0 \Leftrightarrow x = y$
4. Triangle inequality: $d(x, z) + d(z, y) \geq d(x, y)$

We have to take this into account for some clustering algorithms

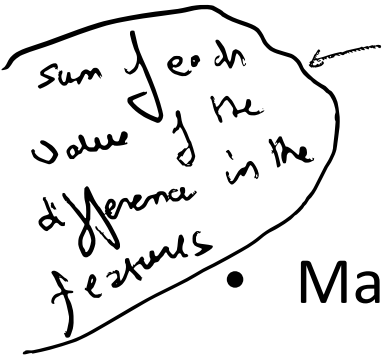
→ if ∆ inequality fails, distances MAY or MAY NOT be applicable. Care is needed

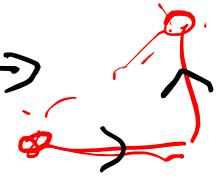
Quantitative Features: *Metric Distances*

- Minkowski distance:

$$d_{ij} = \left(\sum_{l=1}^{d} |x_{il} - x_{jl}|^p\right)^{1/p}$$

  - Special cases are:
    - Euclidean (p=2)
    - City-block (p=1)
    - Sup (p→∞) . Eqv to $d_{ij} = \max_l |x_{il} - x_{jl}|$





(superior distance)

- Mahalanobis distance

$$d_{ij} = (x_i - x_j)^T \mathbb{C}^{-1} (x_i - x_j)$$

(inverse of cov mat. is metric of our distance)

Invariant with respect to any non-singular linear transformation of the coordinates. $\mathbb{C}$ is the covariance matrix.

$\mathbb{M} \rightarrow \mathbb{C}$

"M" (metric) → "C" (cov. mat)

then, Mahalanobis Dist.

Quantitative Features: *Not metric distances*

- Pearson correlation:

$$d_{ij} = \frac{1-r_{ij}}{2}; r_{ij} = \frac{\sum_{k=1}^{m}(x_{k,i}-\bar{x_i})(x_{k,j}-\bar{x_j})}{\sqrt{\sum_{k=1}^{m}(x_{k,i}-\bar{x_i})^2 \sum_{k=1}^{m}(x_{k,j}-\bar{x_j})^2}}$$

$\leftarrow -1$ $\quad 1$

- Cosine similarity:

$$S_{ij} = \frac{x_i{}^T \cdot x_j}{\|x_j\|\|x_i\|} \rightarrow 0, 1$$

$\because = \cos\theta\,(x_i, x_j)$

Also not a metric dist.

cosθ (x_i, x_j)

if they are fully anticorrelated, $d_{ij} = -1$, if fully correlated, $d_{ij} = 1$

But doesn't accomplish 0-inequality. So, not a metric dist.

Qualitative Features

- Jaccard similarity:

$$S_{ij} = \frac{|i \cap j|}{|i \cup j|}$$

$i = 10001001\ j = 01001011; S_{ij} = \frac{2}{5}$

- Hamming distance:

$$D_{ij} = |i \cup j| - |i \cap j|$$

i.e. minimum number of changes that you need to turn i in j.

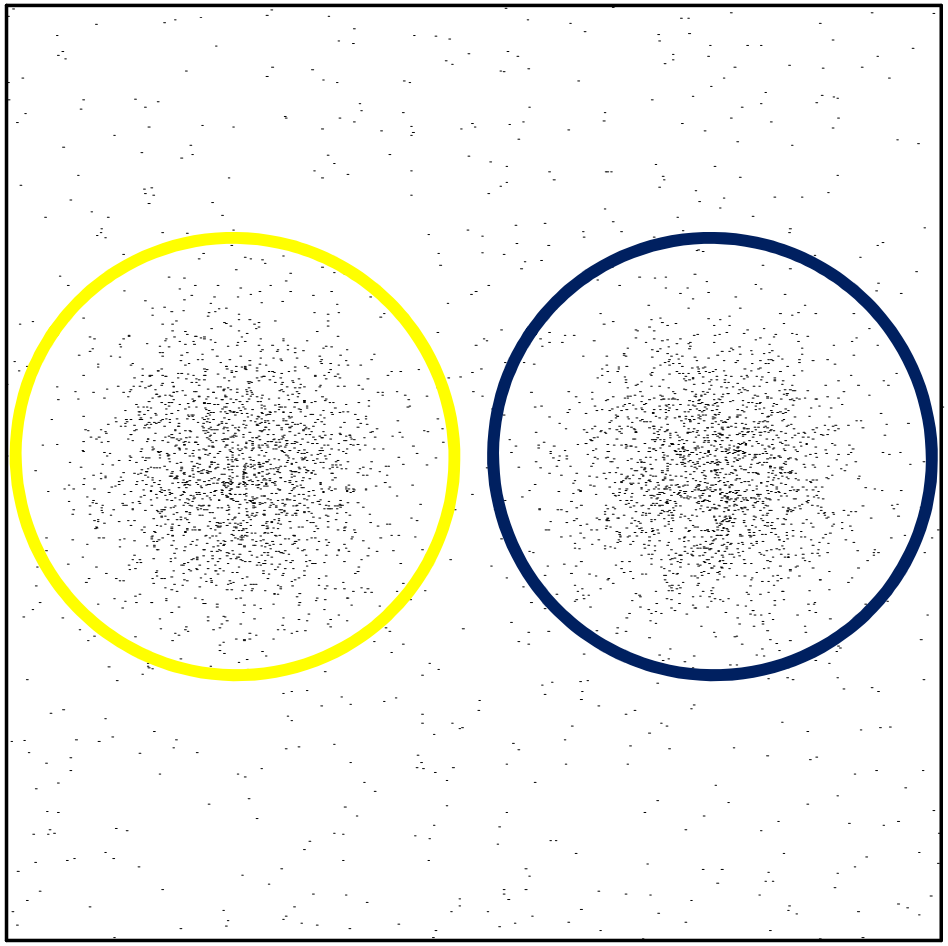
More complicated distances

- Working in the metric can extremely simplify the clustering work.
- A good metric can dramatically improve the performance of an algorithm.
- However, usually they need to compute a simplest distance as starting point.
- Example: Geodesic distance → (in Isomap starting from euclidean)

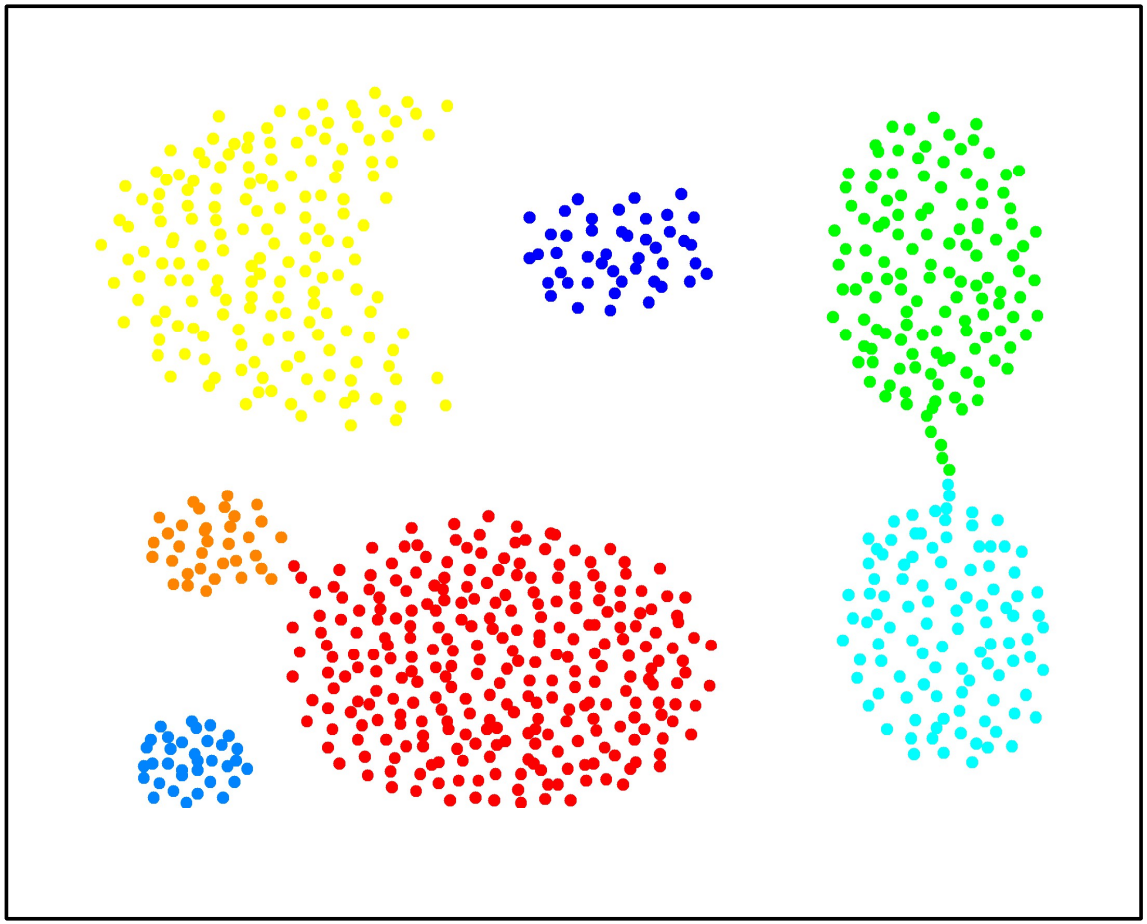
→ we can start with a simple (euclidean, hamming etc.) dist, & then convert it to something more complicated.

CLUSTERING

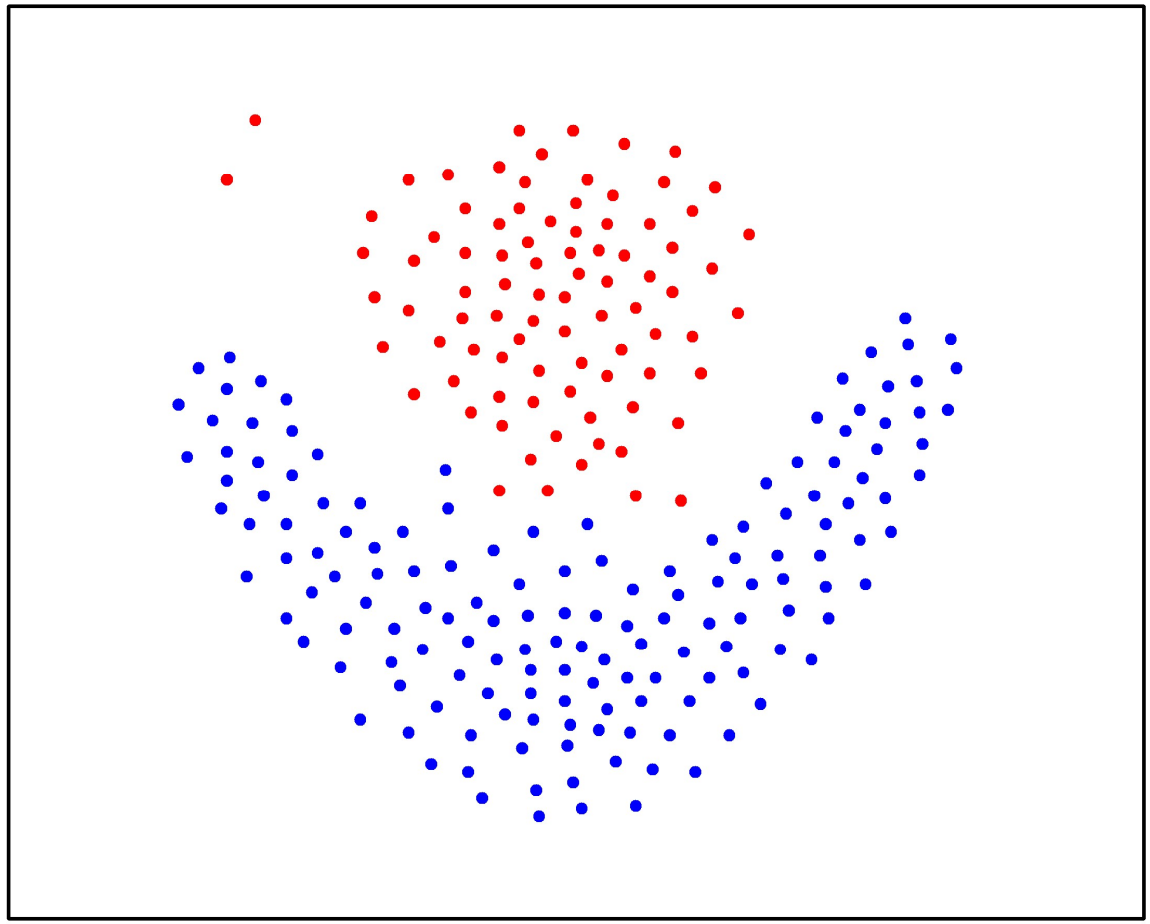
What is a cluster???



Other cluster examples…

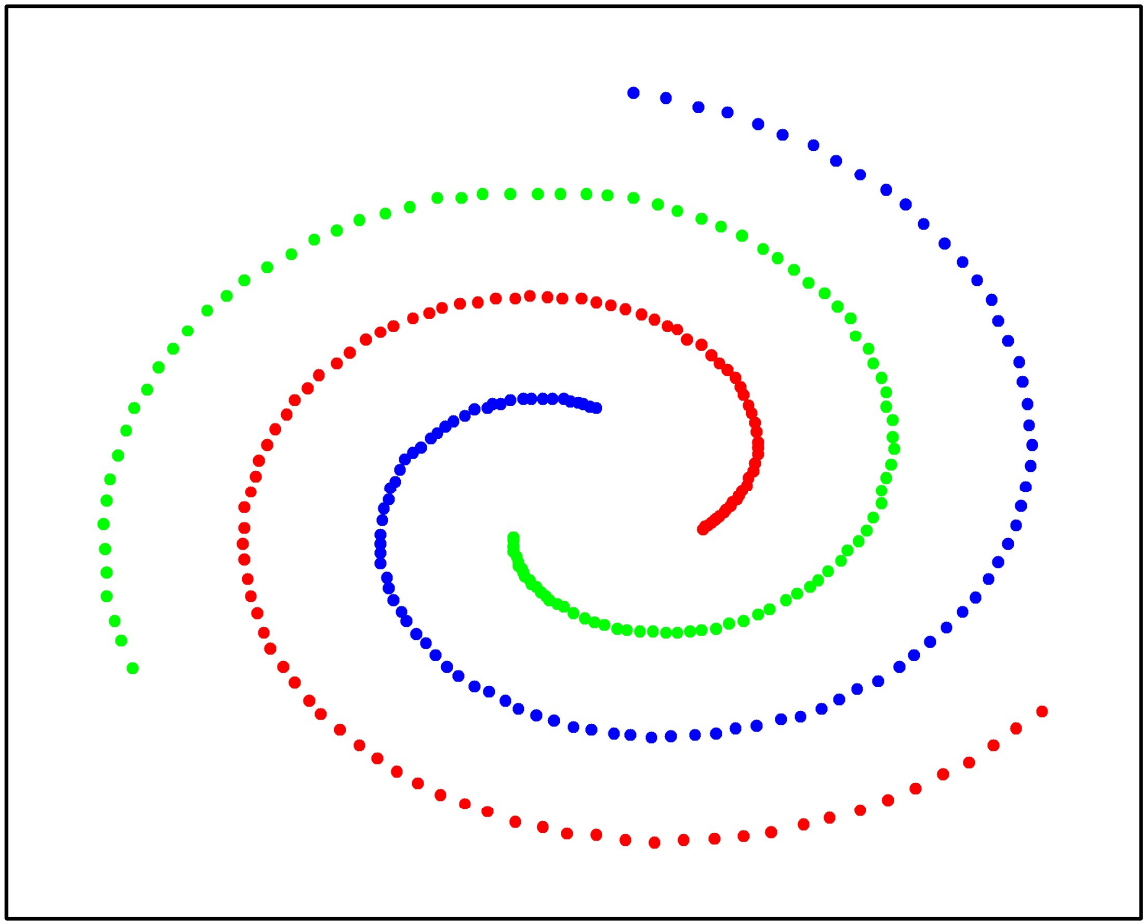


Other cluster examples…

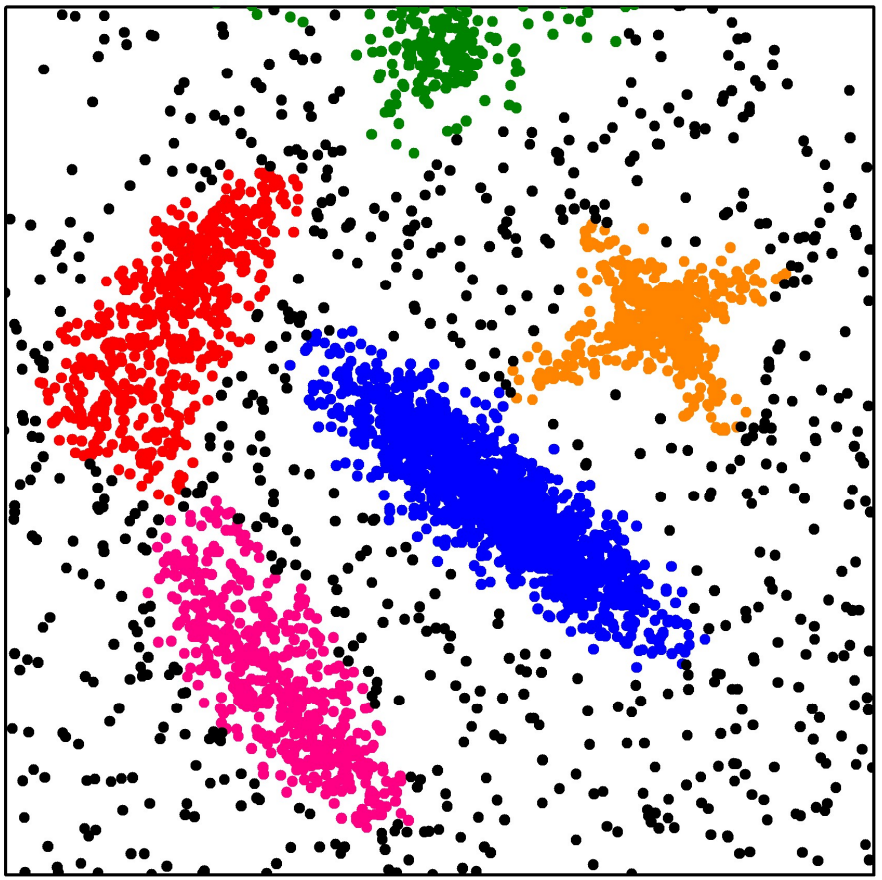


Not so easy to define mathematically

Other cluster examples…



Other cluster examples…

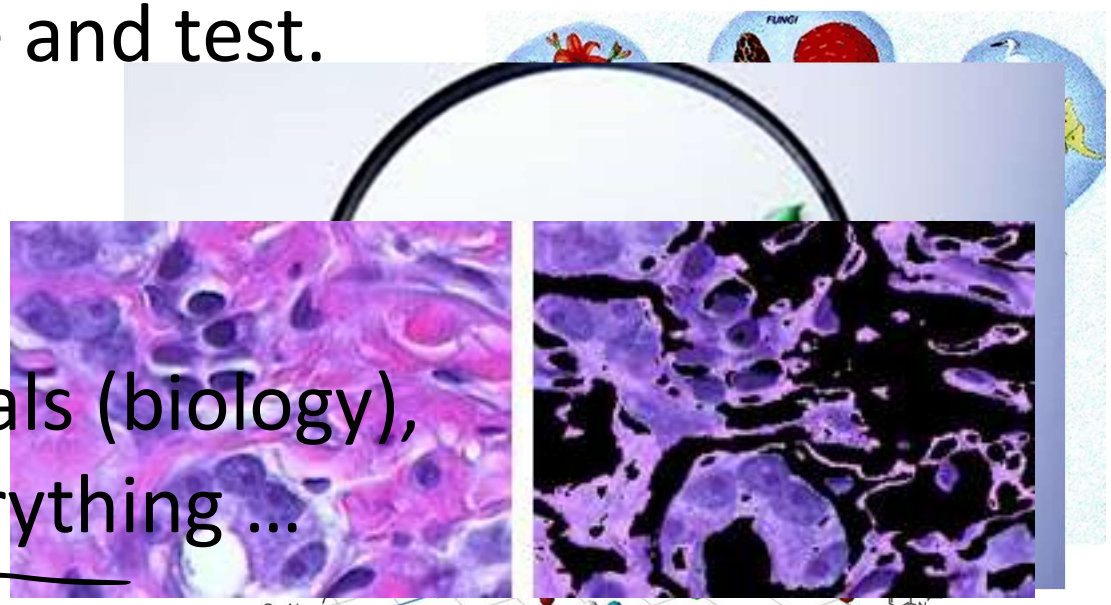


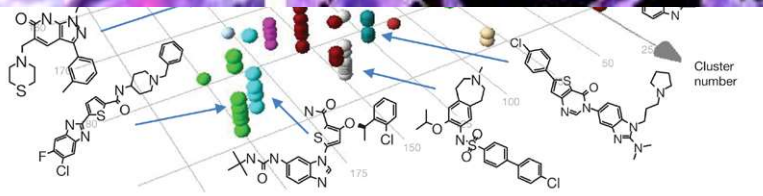
→ no single def of what a cluster is

Data mining technique that can be used for:

(Uses of clustering)

- Decide which set of drugs from a big library we should synthetize and test.
- WWW (googling…)
- Image recognition
- Classify plants, animals (biology), books(libraries), everything …



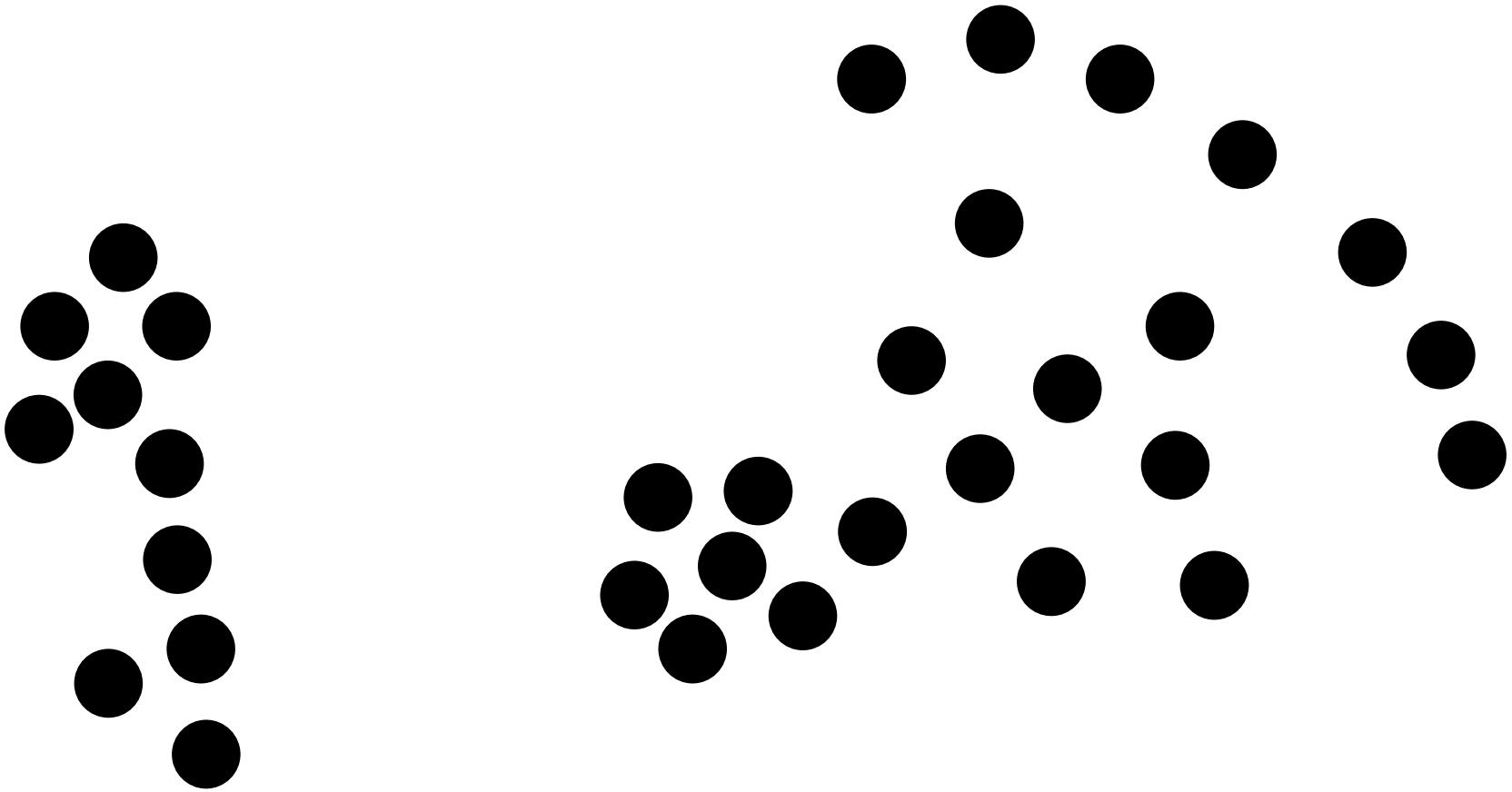


classifying labelled data.

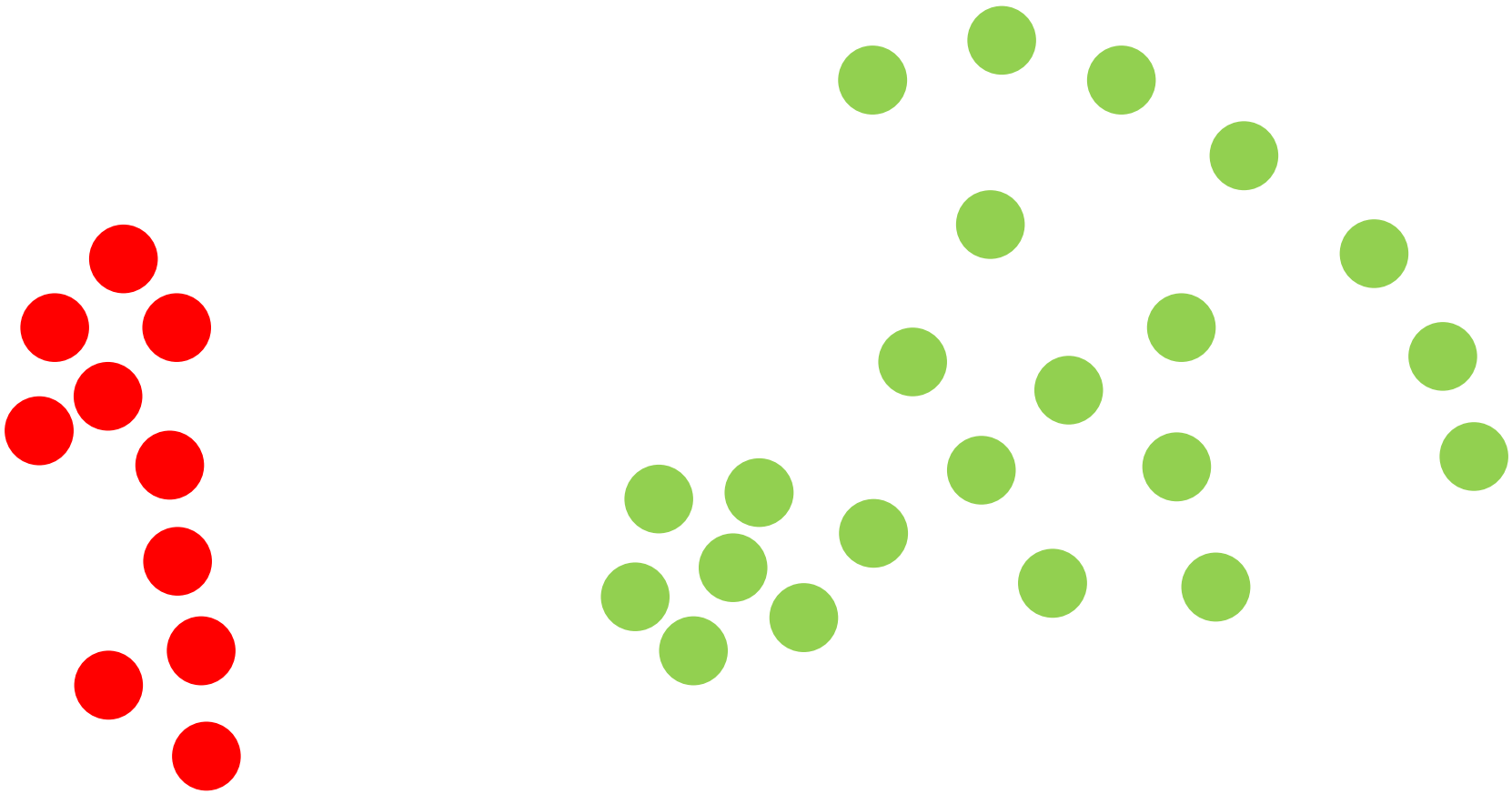
Types of clustering: Flat, hierarchical and fuzzy clustering

classification based on output of clustering method.

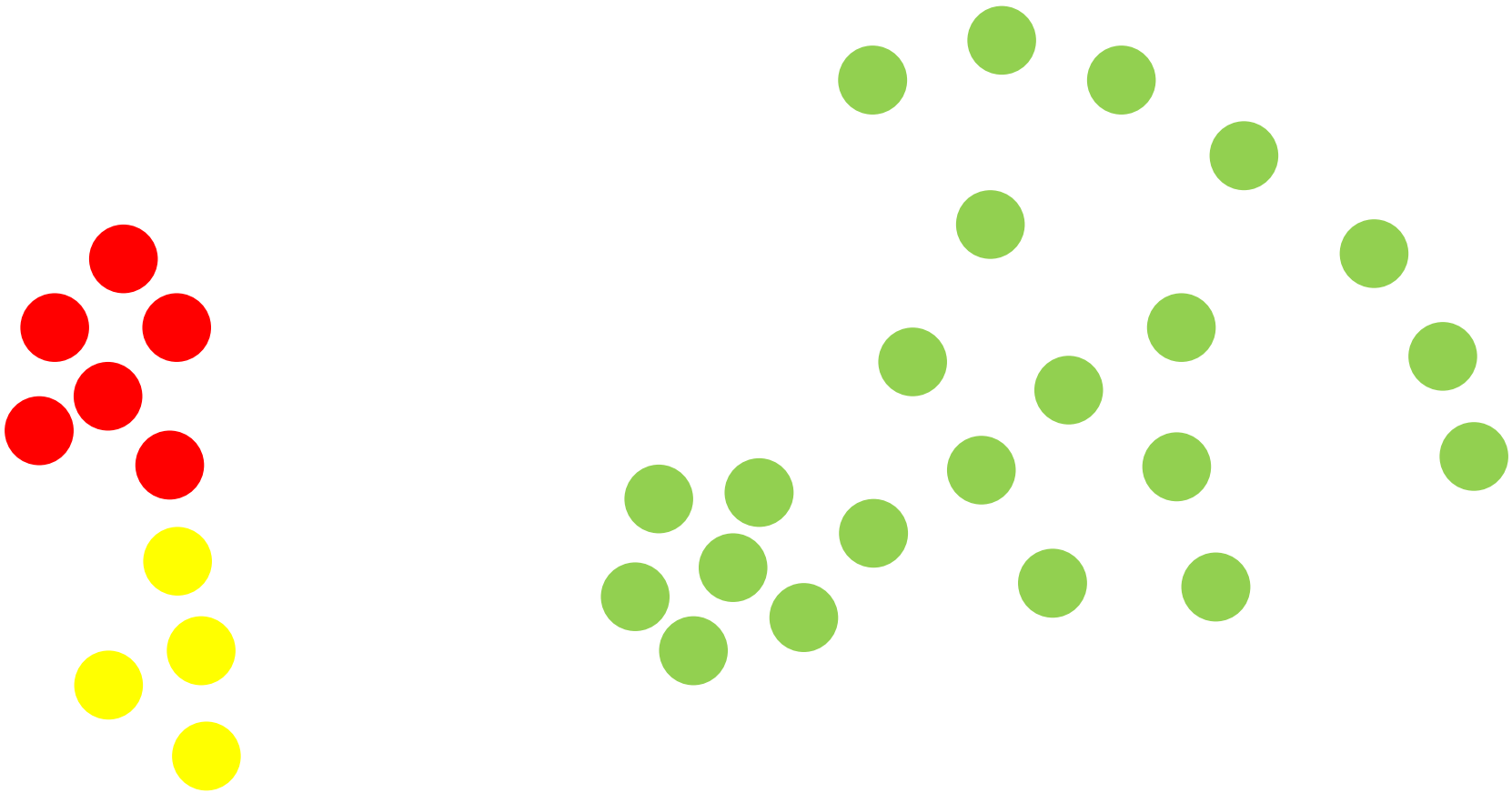
What is a cluster (revisited)



What is a cluster (revisited)



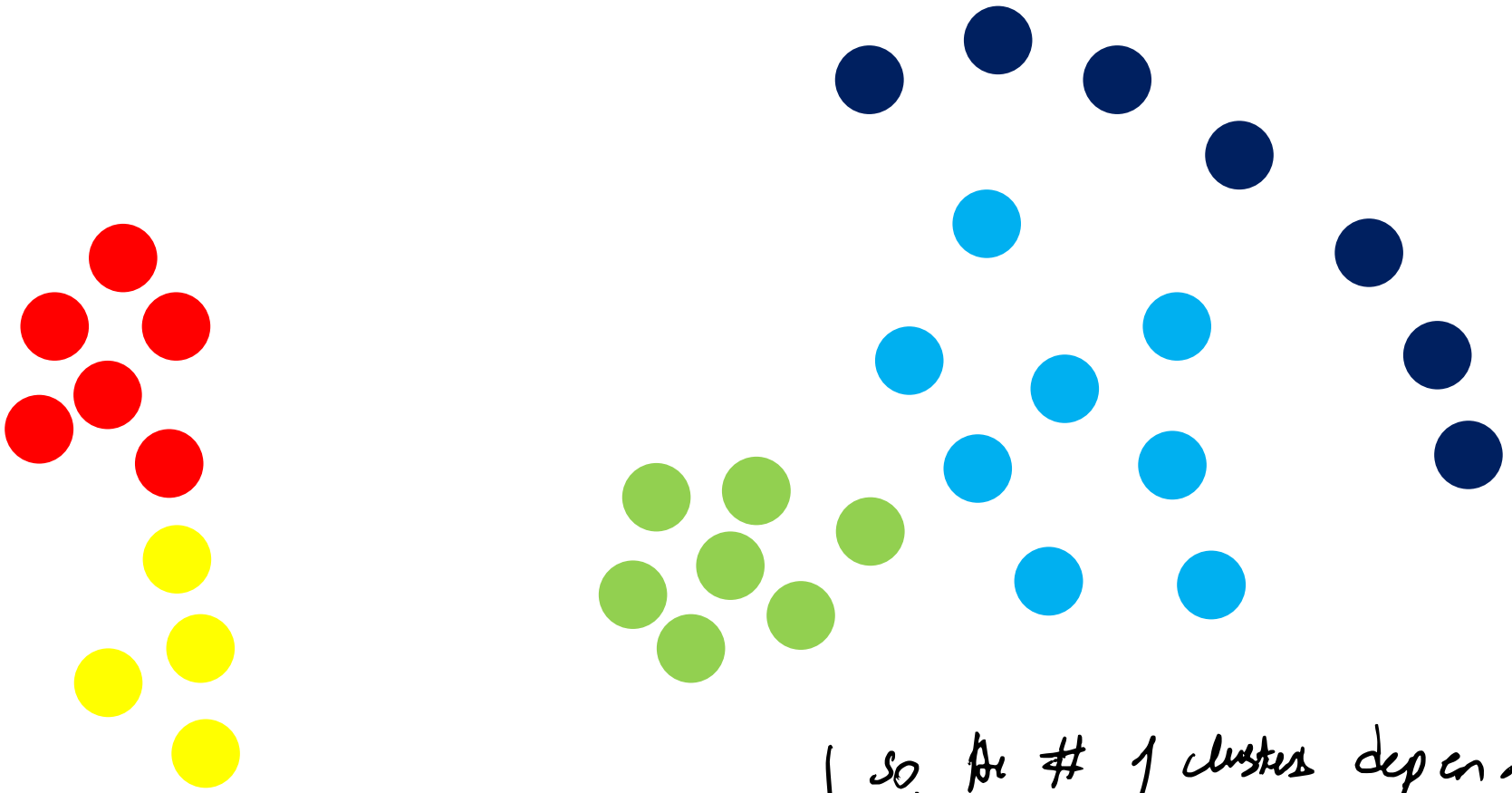
What is a cluster (revisited)



What is a cluster (revisited)

What is a cluster (revisited)

What is a cluster (revisited)



→ also depends on chosen features, metric dist. used etc.

so the # of clusters depends on algorithm + how we proceed.

Some consideration about Clustering

- Tautology: the result of the clustering process depends on your cluster definition
- And it depends on the metric
- And it also depends on the features that you have chosen

Flat, fuzzy and hierarchical clustering

- Flat clustering performs a hard partition of the data
- Fuzzy clustering is a flat clustering algorithm with soft element assignation (explained later)
- Hierarchical clustering generates a tree instead of a single partition. Can be agglomerative (joining elements) or divisive (dividing the dataset) → (NOT a hard partition)

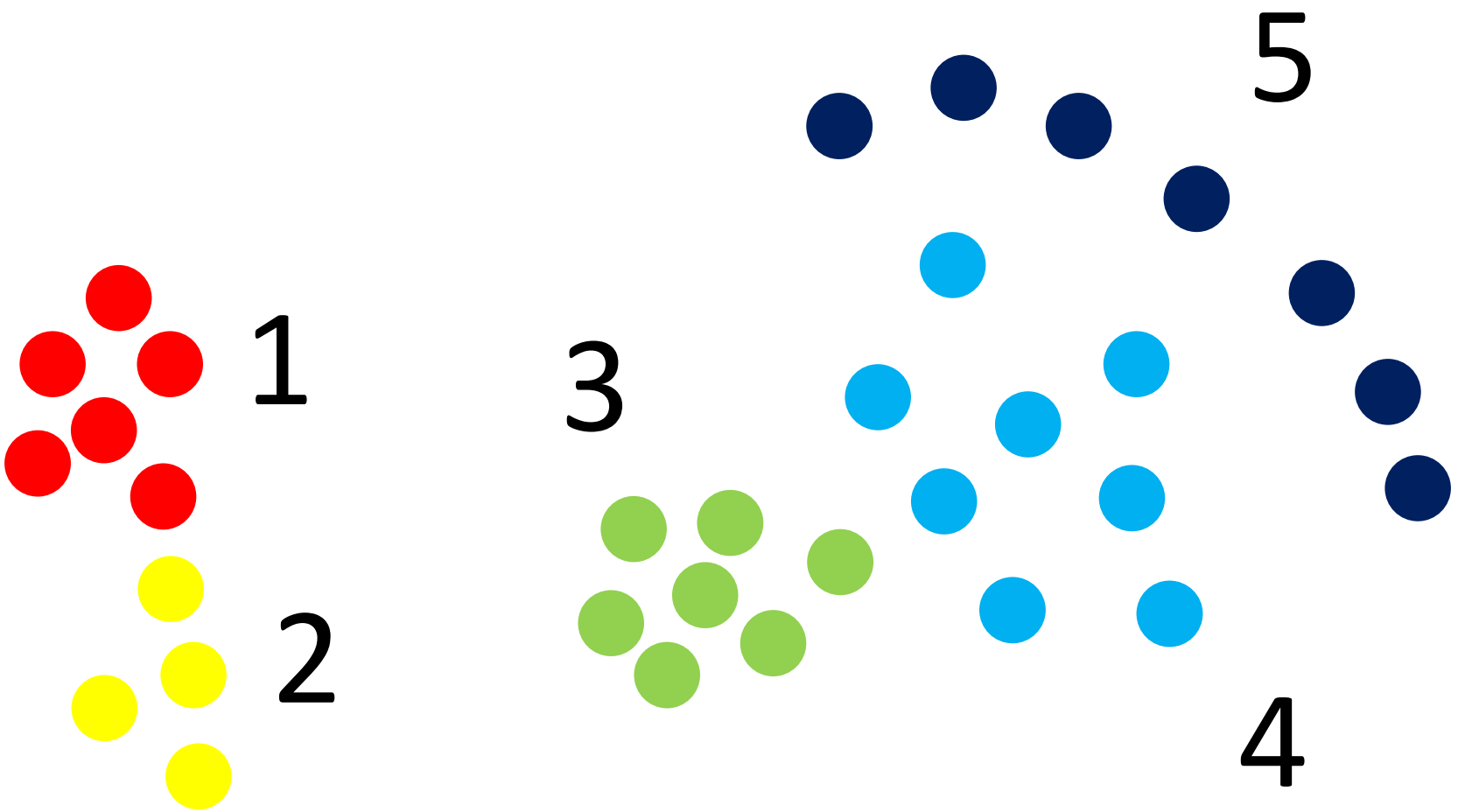
Flat clustering

- Each element is assigned to a single cluster.

$$Cl(i) = l$$

- Traditionally, the number of clusters (k) should be given to the algorithm as an external parameter. Nowadays, many clustering algorithms estimate internally this parameter.
- usually when one performs clustering one looks for a hard partition.
- Can not deal well with multilevel structures

Flat clustering



Fuzzy clustering

- Each element is assigned to *all* the clusters in the dataset with a given degree of membership.

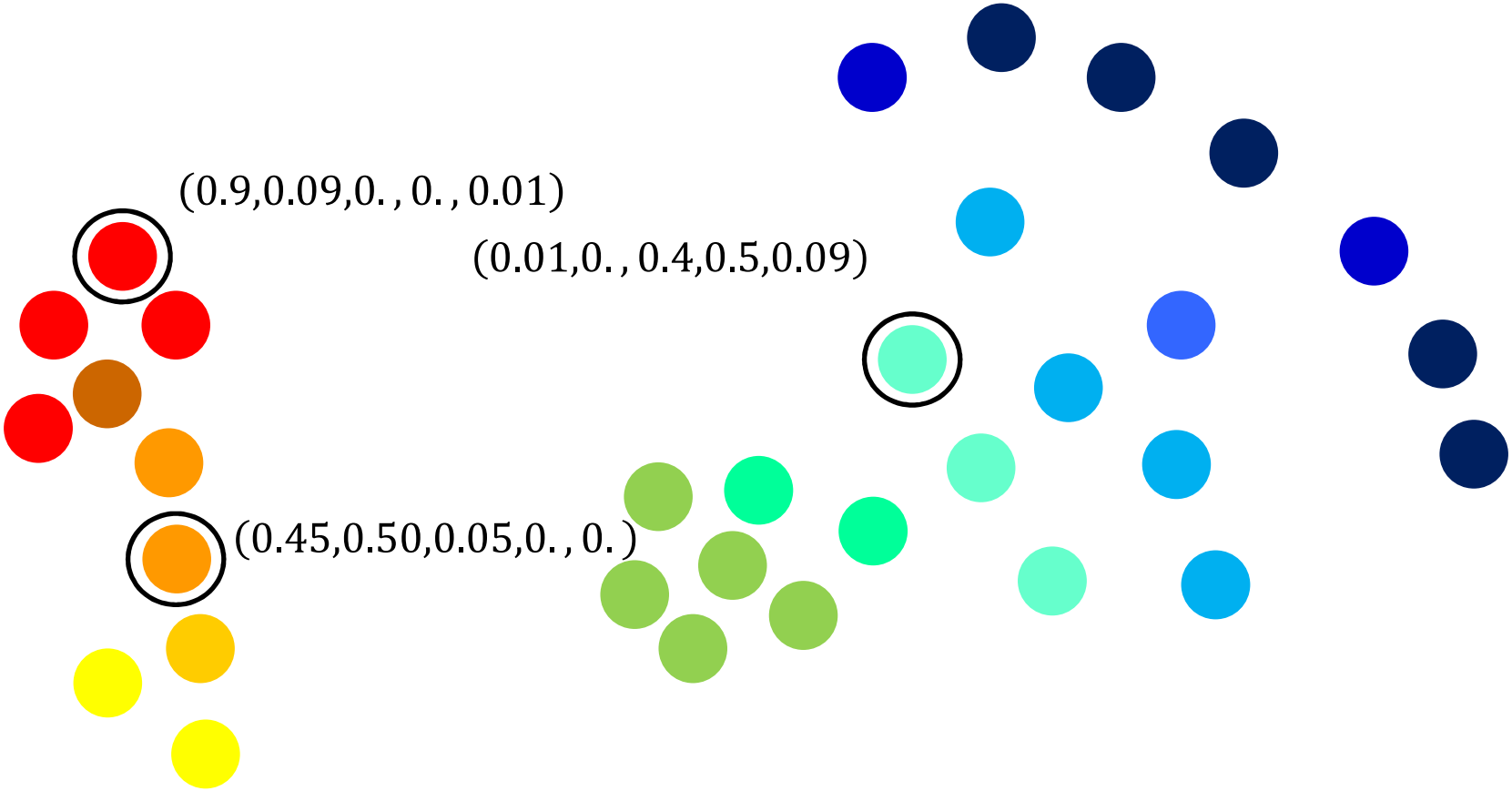
$$\overrightarrow{Cl}(i) = (u_1, u_2, \dots u_l \dots, u_k)$$

- The membership vector is normalized

$$\sum_{l=1}^{k} u_l = 1$$

- Again, the number of clusters should be provided as an external parameter.
- It may be difficult to transform in a hard partition

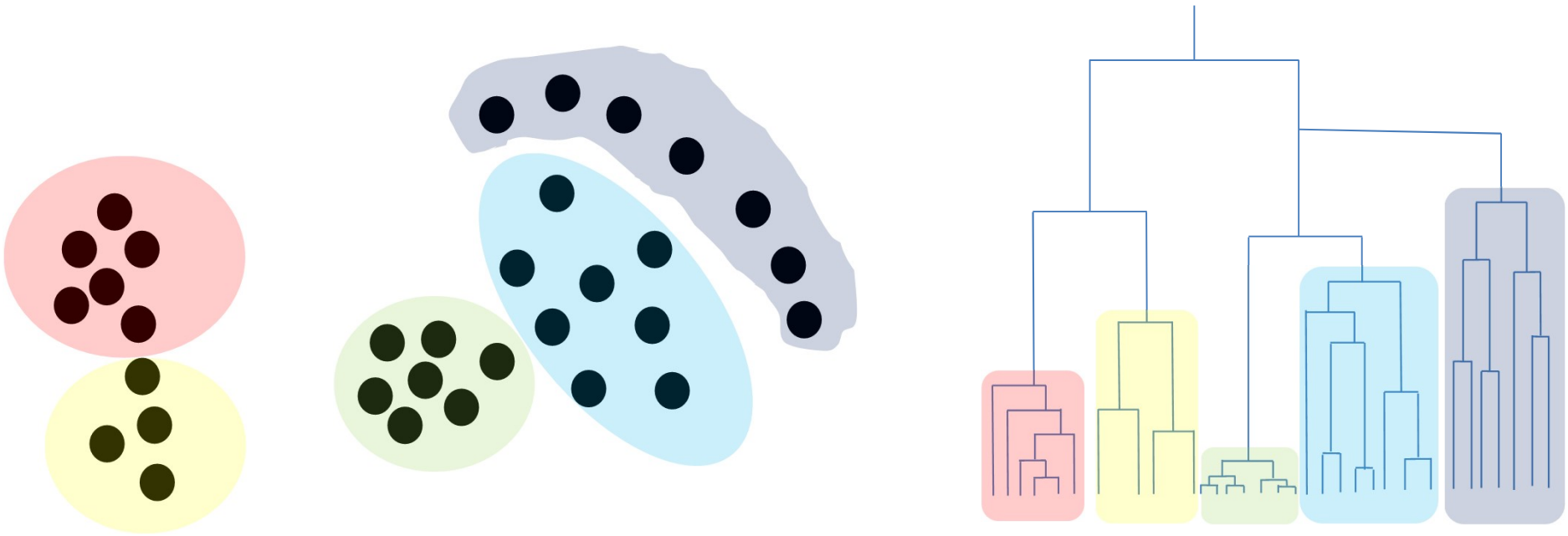
Fuzzy clustering



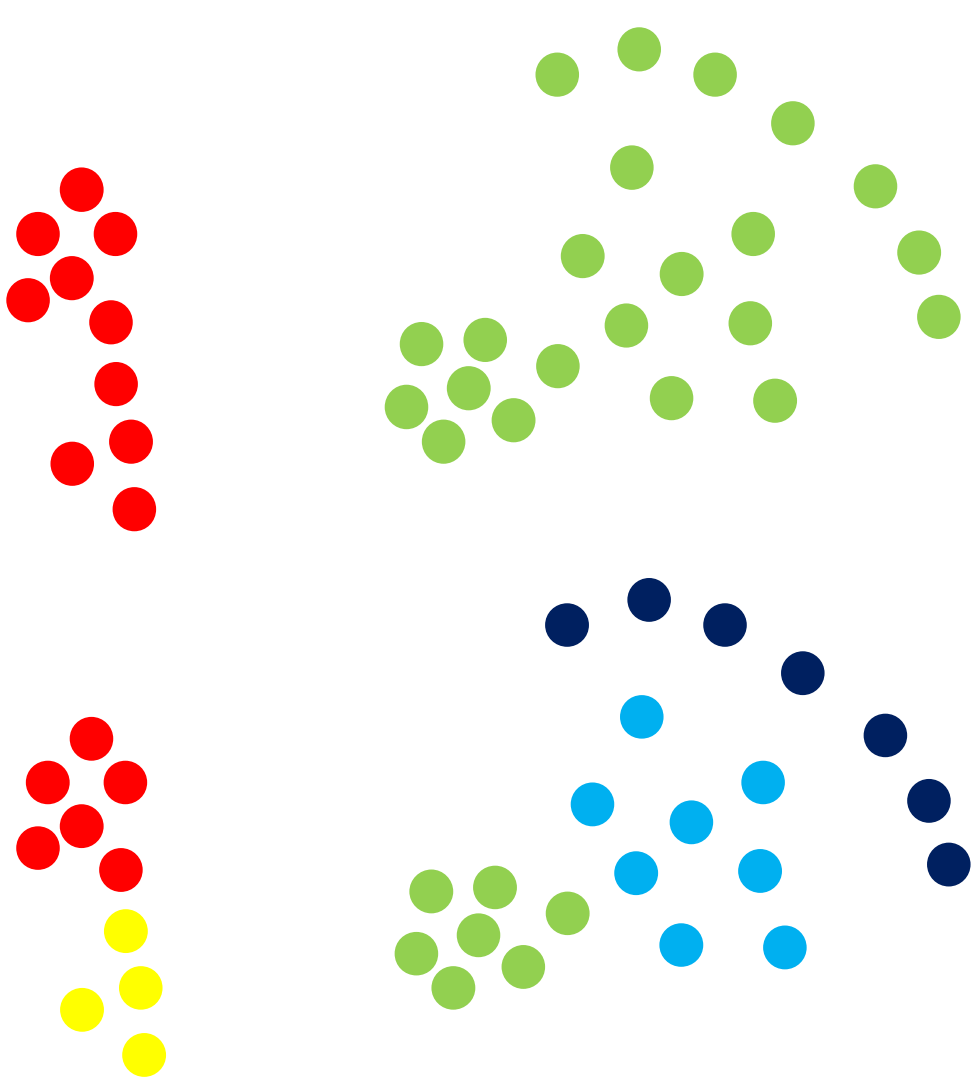
Hierarchical clustering

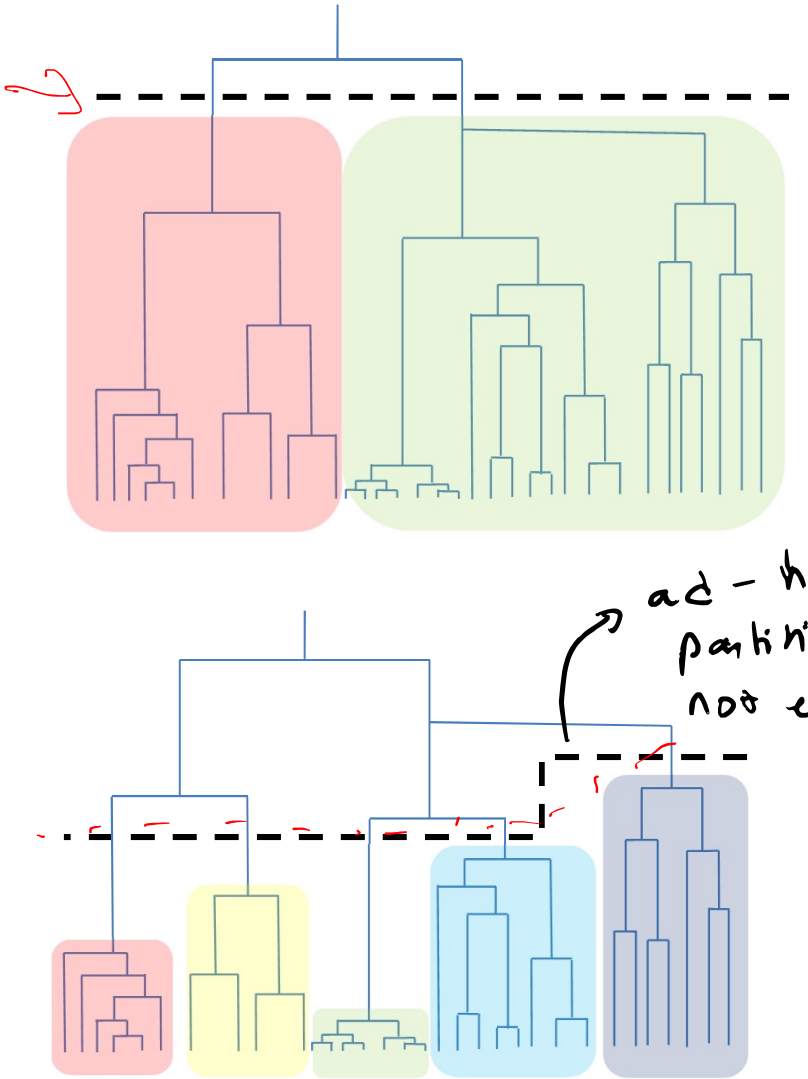
- Produces a set of nested clusters organized as a hierarchical tree
- Can be visualized as a dendrogram (A tree-like diagram that records the sequences of merges or splits)
- No assumptions on the number of clusters
- Hierarchical clusterings may correspond to meaningful taxonomies (biological trees / taxonomies for eg.)
- It can be transformed in many hard partitions

Hierarchical clustering



Flattening the hierarchical clustering





K-means: A flat clustering algorithm

MacQueen, J. (1967, June). Some methods for classification and analysis of multivariate observations. In Proceedings of the fifth Berkeley symposium on mathematical statistics and probability (Vol. 1, No. 14, pp. 281-297).

K-means clustering

- Attempts to minimize the *intracluster* distance while maximizing the *intercluster* distance.
- It is based on the concept of cluster centroid, i.e. the average position of the cluster elements.
- Still widely used.
- It can be easily parallelized and linearized.
- User must provide *k*

K-means objective function

- Objective function:

$$O(z) = \sum_{l=1}^{k} \sum_{i=1}^{n} \delta(z_i, l) \; \|\overrightarrow{x_i} - \overrightarrow{c_l}\|^2$$

- z is an array with n components that reflects the assignation in clusters.
- $\overrightarrow{c_l}$ is the vector with the coordinates of the l-th cluster centroid. $\overrightarrow{c_l} = \frac{\sum_{i=1}^{n} \delta_{z_{il}} \overrightarrow{x_i}}{\sum_{i=1}^{n} \delta_{z_{il}}}$

Globally minimizing this func^n is an NP hard problem so we develop an algo to do this minimization locally.

k-means algorithm

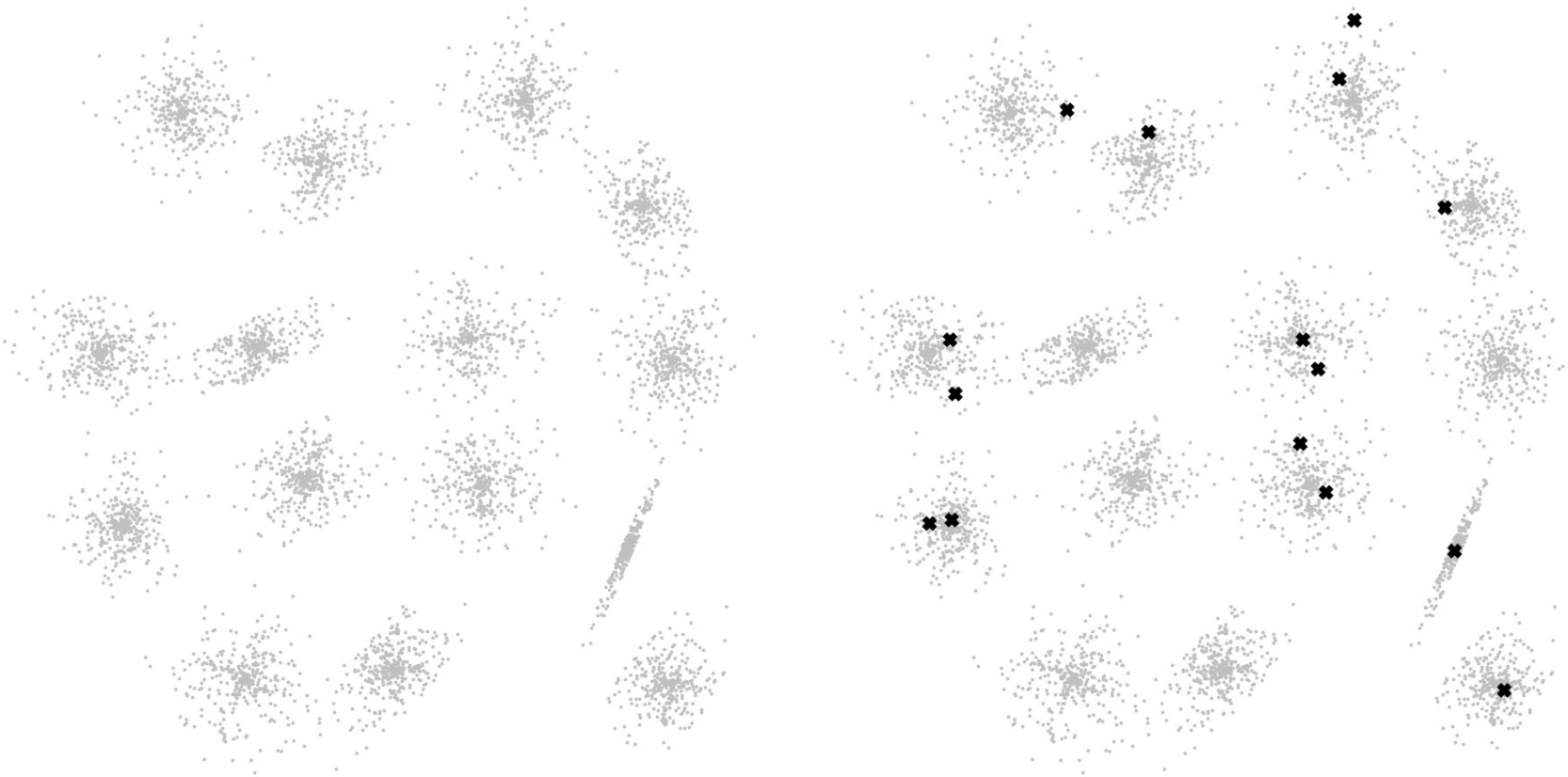
Input: cluster size *k*, instances $\{\overrightarrow{x_i}\}_{i=1}^{n}$

Output: cluster membership assignments $\{z_i\}_{i=1}^{n}$

1. Initialize *k* cluster centroids $\{\overrightarrow{c_l}\}_{l=1}^{k}$ (randomly from the data set).

randomly from the data points

1. Initialize *k* cluster centroids



gives us a "Voronoi" partition of the space.

k-means algorithm

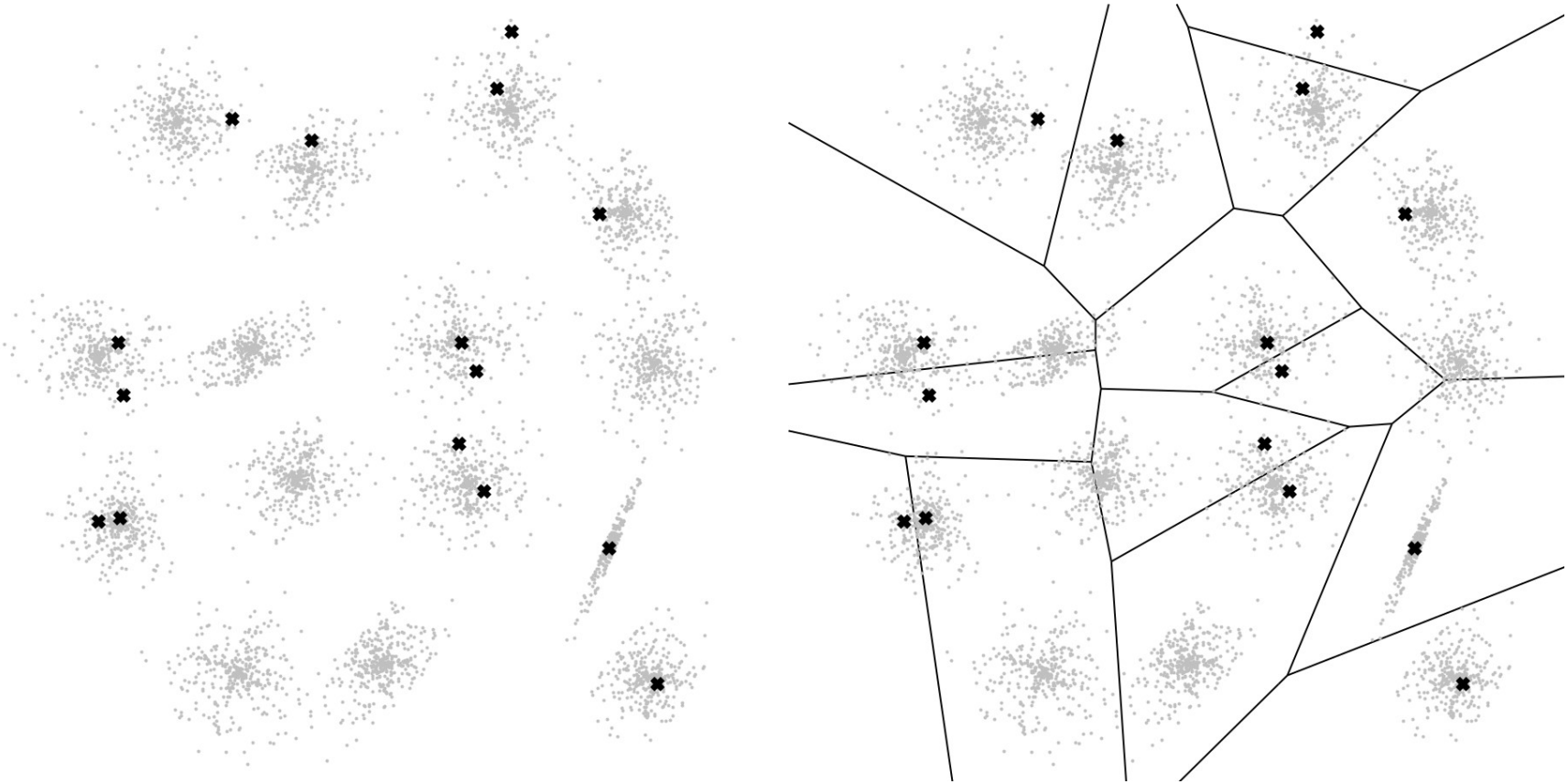
Input: cluster size *k*, instances $\{\overrightarrow{x_i}\}_{i=1}^{n}$

Output: cluster membership assignments $\{z_i\}_{i=1}^{n}$

1. Initialize *k* cluster centroids $\{\overrightarrow{c_l}\}_{l=1}^{k}$ (randomly from the data set).
2. Repeat until no instance changes its cluster membership:
   a) Decide the cluster membership of instances by assigning them to the nearest cluster centroid $z_i = argmin_l(d(\overrightarrow{c_l}, \overrightarrow{x_i}))$

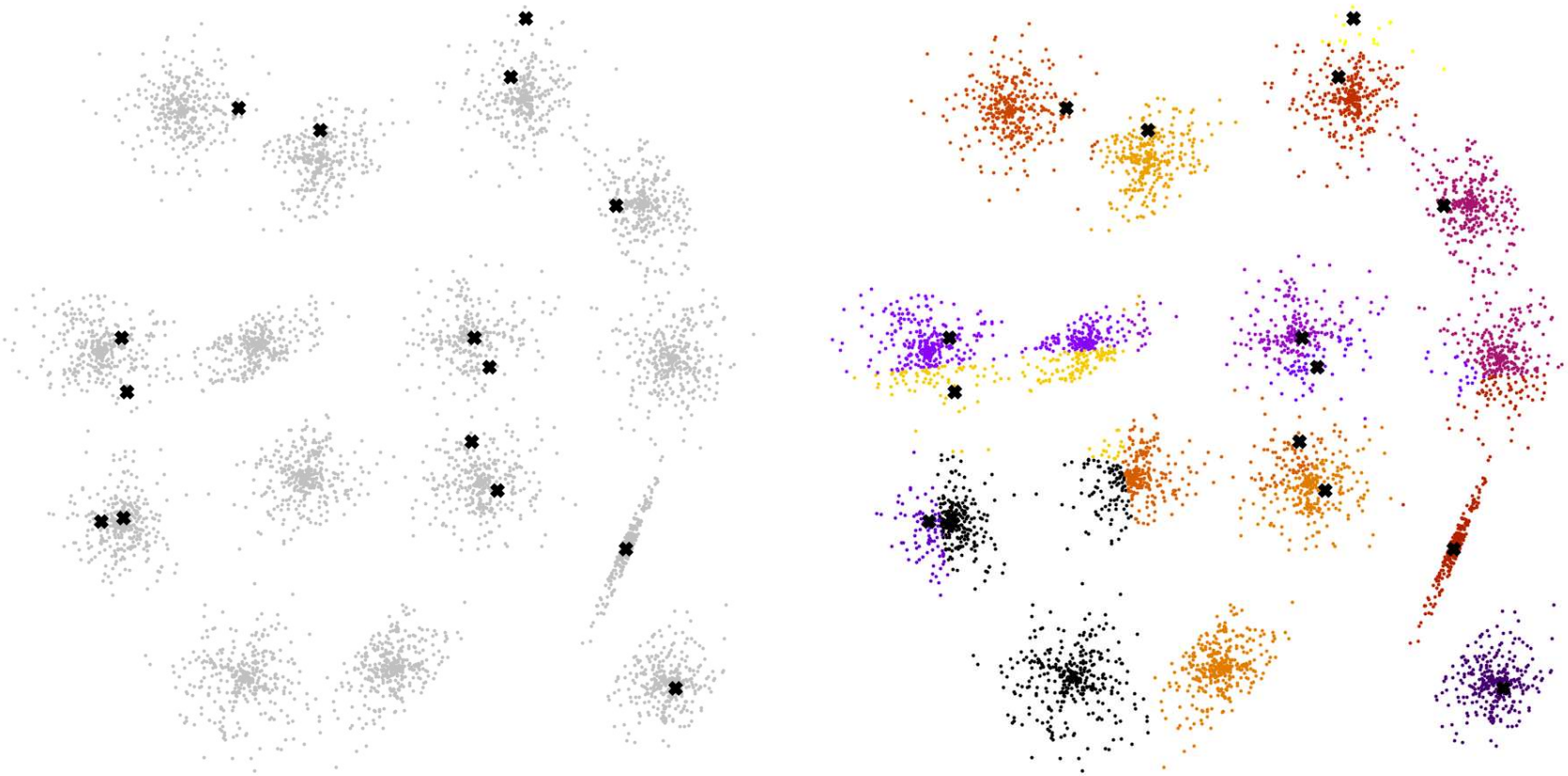
2. a) Nearest cluster centroid

$$z_i = argmin_l(d(\overrightarrow{c_l}, \overrightarrow{x_i}))$$



Voronoi partition

2. a) Nearest cluster centroid



k-means algorithm

Input: cluster size *k*, instances $\{\overrightarrow{x_i}\}_{i=1}^{n}$

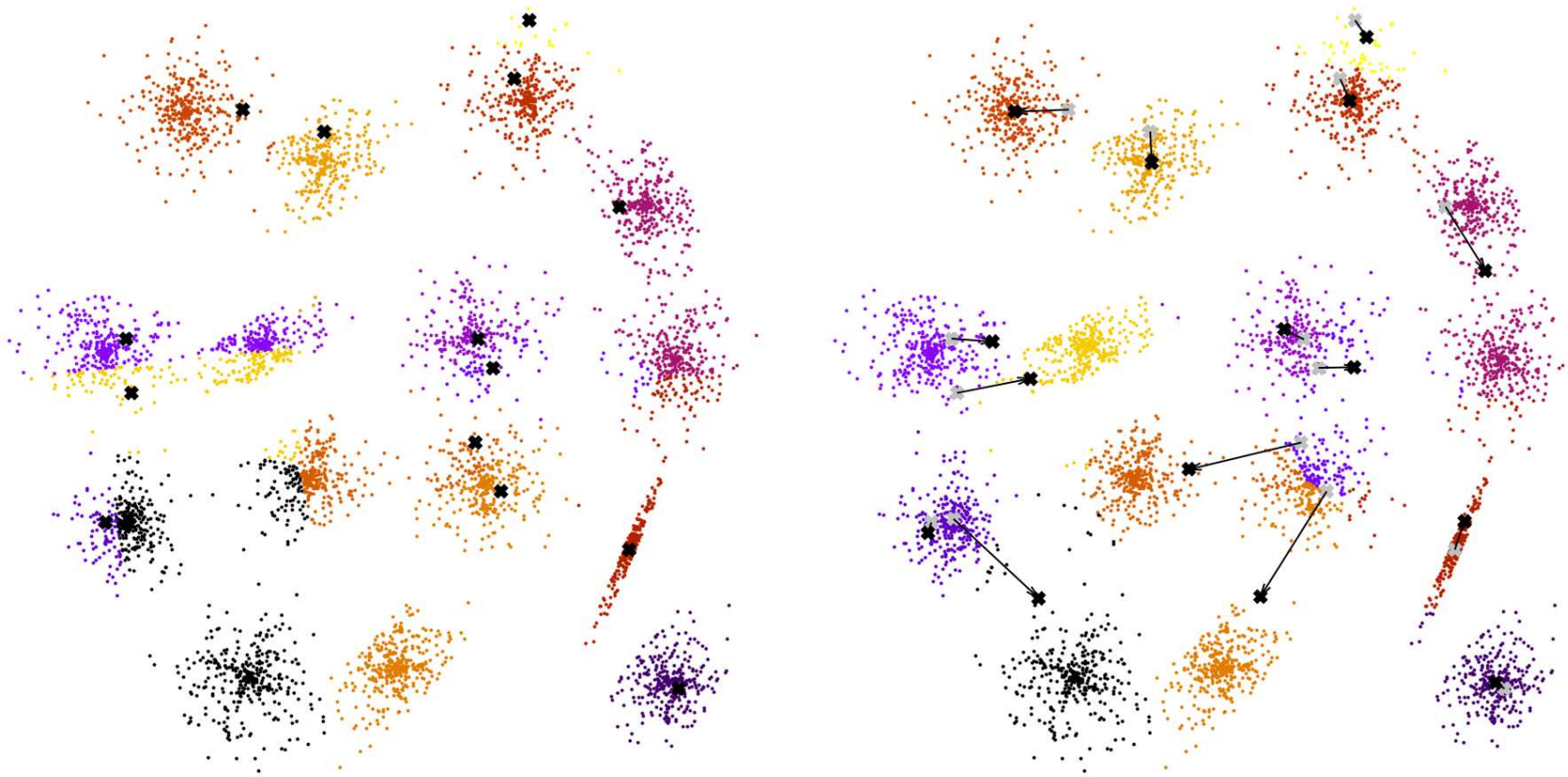
Output: cluster membership assignments $\{z_i\}_{i=1}^{n}$

1. Initialize *k* cluster centroids $\{\overrightarrow{c_l}\}_{l=1}^{k}$ (randomly from the data set).
2. Repeat until no instance changes its cluster membership:
   a) Decide the cluster membership of instances by assigning them to the nearest cluster centroid $z_i = argmin_l(d(\overrightarrow{c_l}, \overrightarrow{x_i}))$
   b) Update the *k* cluster centroids based on the assigned cluster membership

$$\overrightarrow{c_l} = \frac{\sum_{i=1}^{n} \delta_{z_{il}} \overrightarrow{x_i}}{\sum_{i=1}^{n} \delta_{z_{il}}}$$

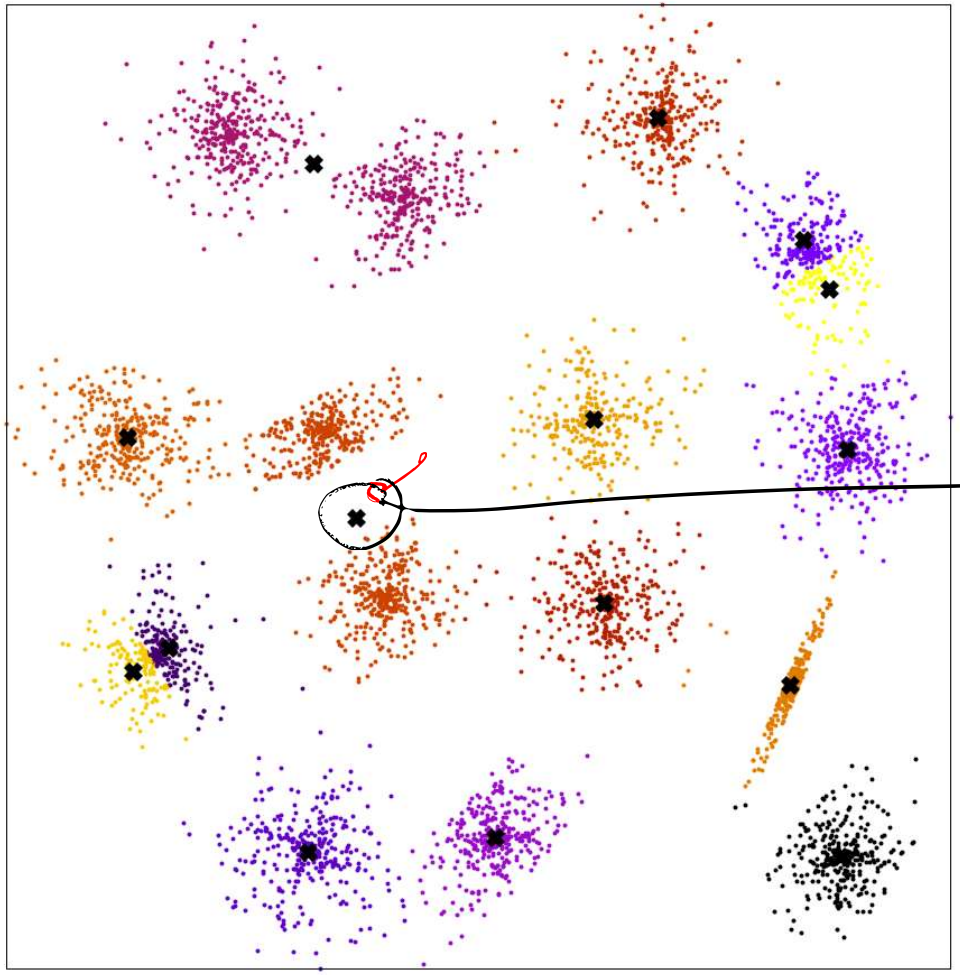
2. b) Recompute centroids

$$\overrightarrow{c_l} = \frac{\sum_{i=1}^{n} \delta_{z_{il}} \overrightarrow{x_i}}{\sum_{i=1}^{n} \delta_{z_{il}}}$$



k-means illustration

- Randomly pick *k* centers.
- Assign each point to its nearest center.
- Recompute centers.
- Iterate…



will we achieve the first partition

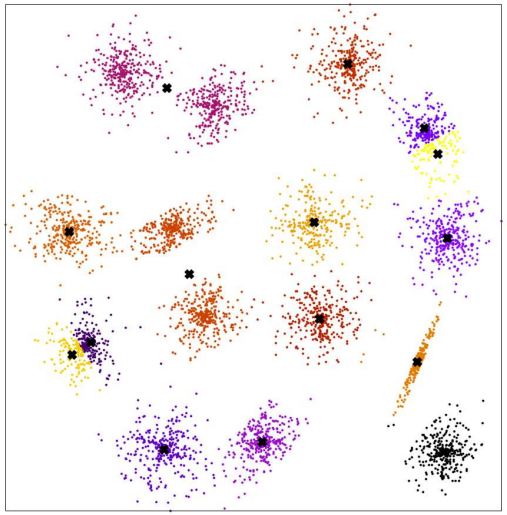
consequence of arriving to "local minima" NOT "global minima".

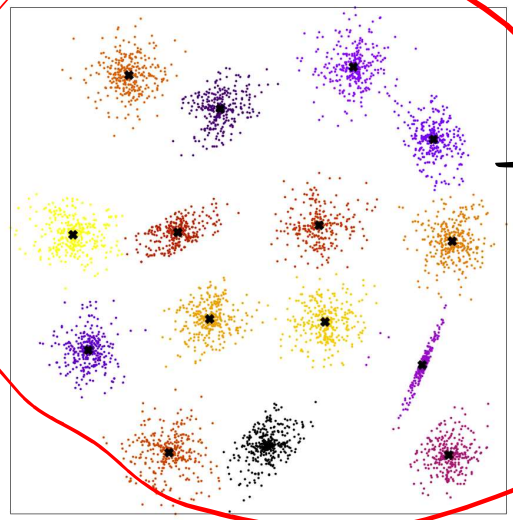
K-means weakness 4 things

- Initialization sensitive (local optimization) →k-means++
- Which k-employing →Scree test
- Sensitive to outliers →k-medoids
- employs Euclidean distance →k-medoids
- Only for spherical clusters →kernel k-means (advanced)

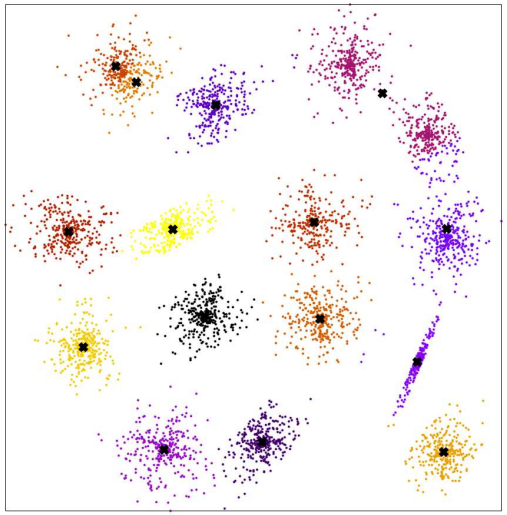
Different initializations

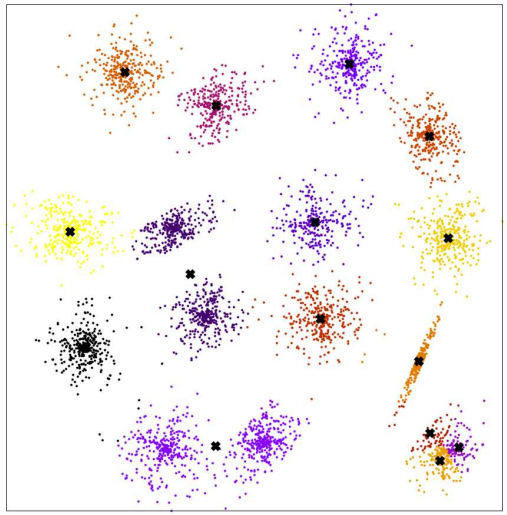
give very diff. results





→ correct, rest are wrong.





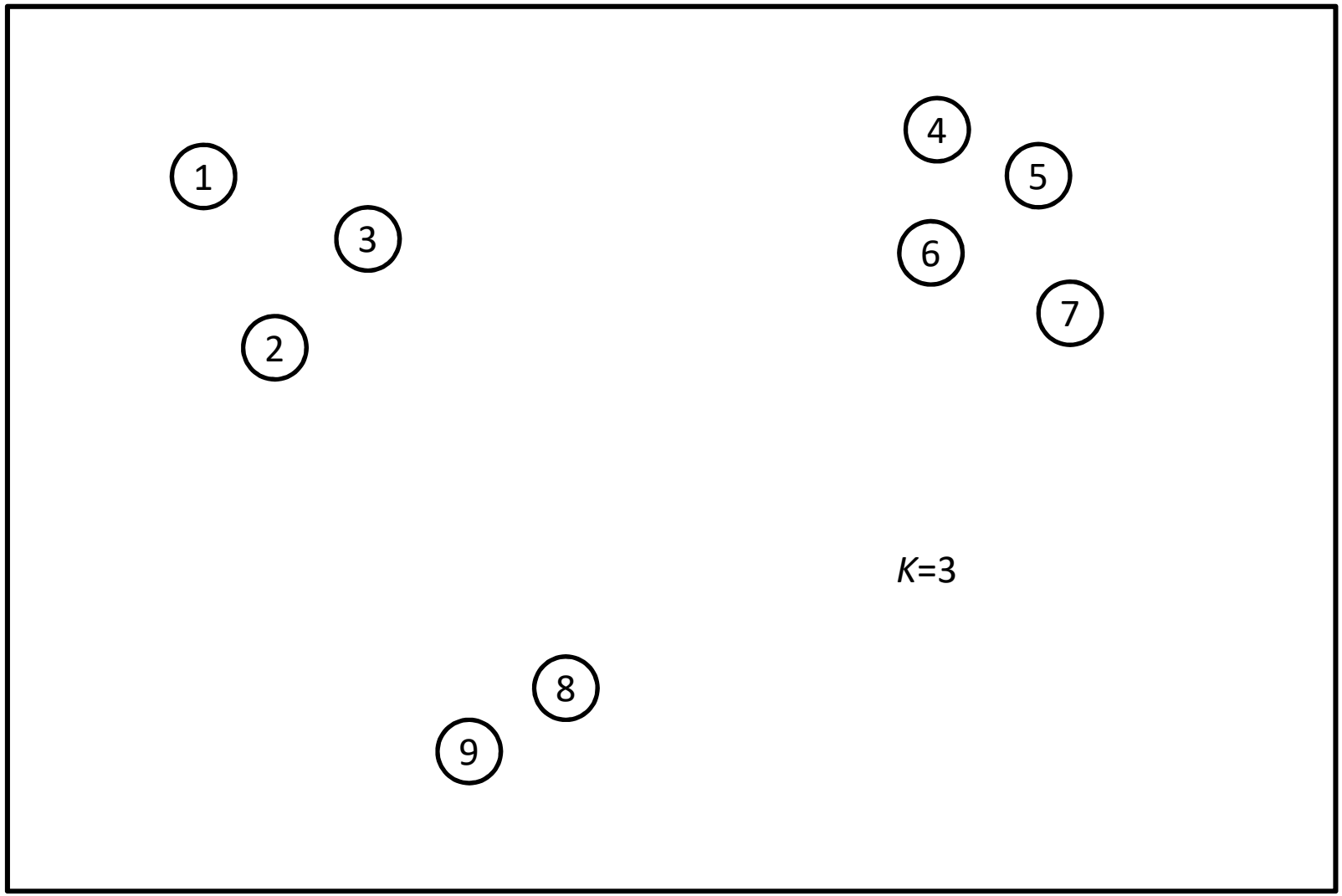
Better initialization: *k*-means++

1. Choose the first cluster center at random
2. Repeat until all *k* centers have been found
   - For each instance compute $\mathrm{D_x} = \min_{\mathrm{k}}[d(x, c_k)]$
   - Choose a new cluster center with probability $p(x) \propto D_x^2$ *new center should be far away from existing centers*
3. Run *k*-means with selected centers as initialization

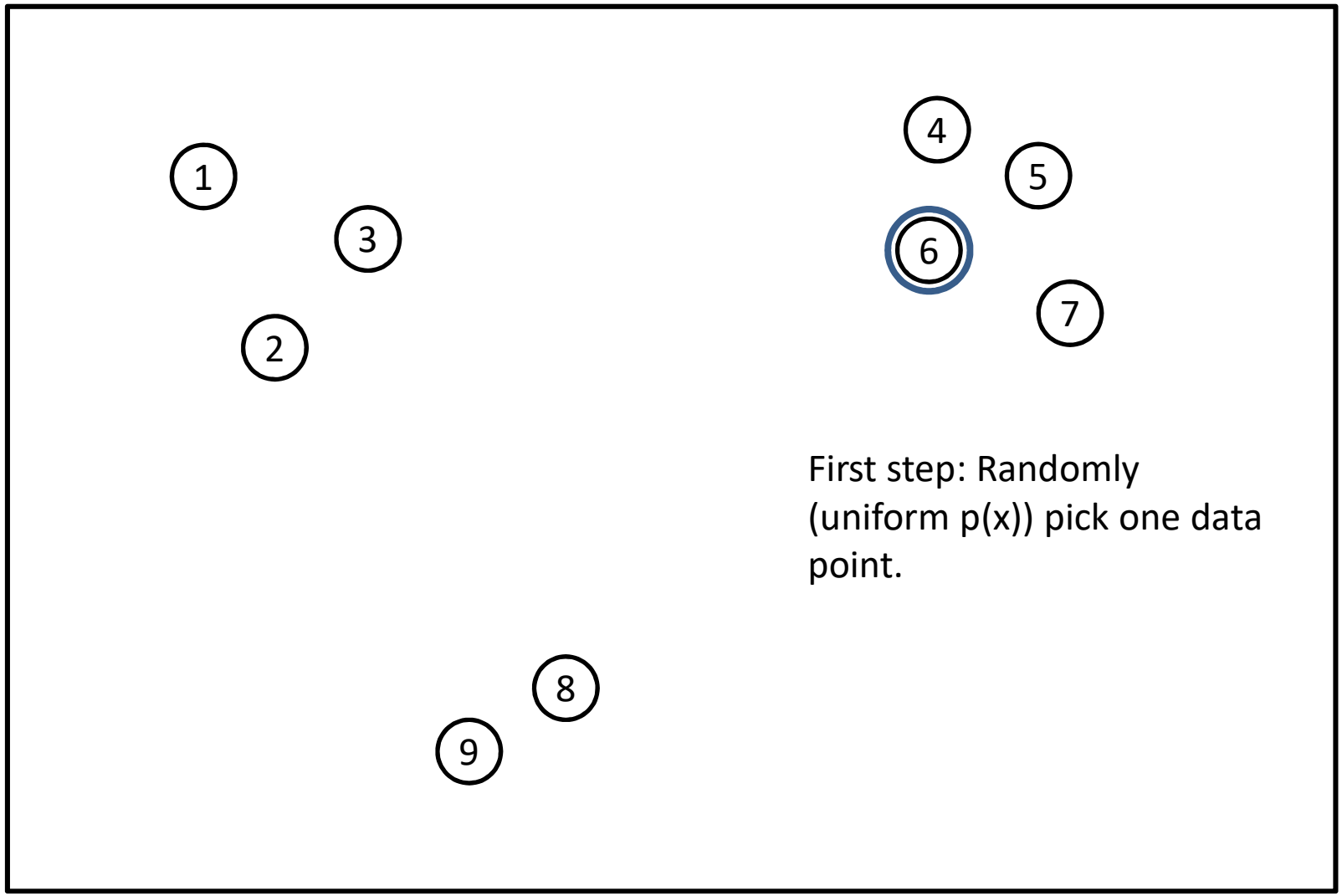
min dist. from each data point to an already assigned center.

while maintaining random assignment.

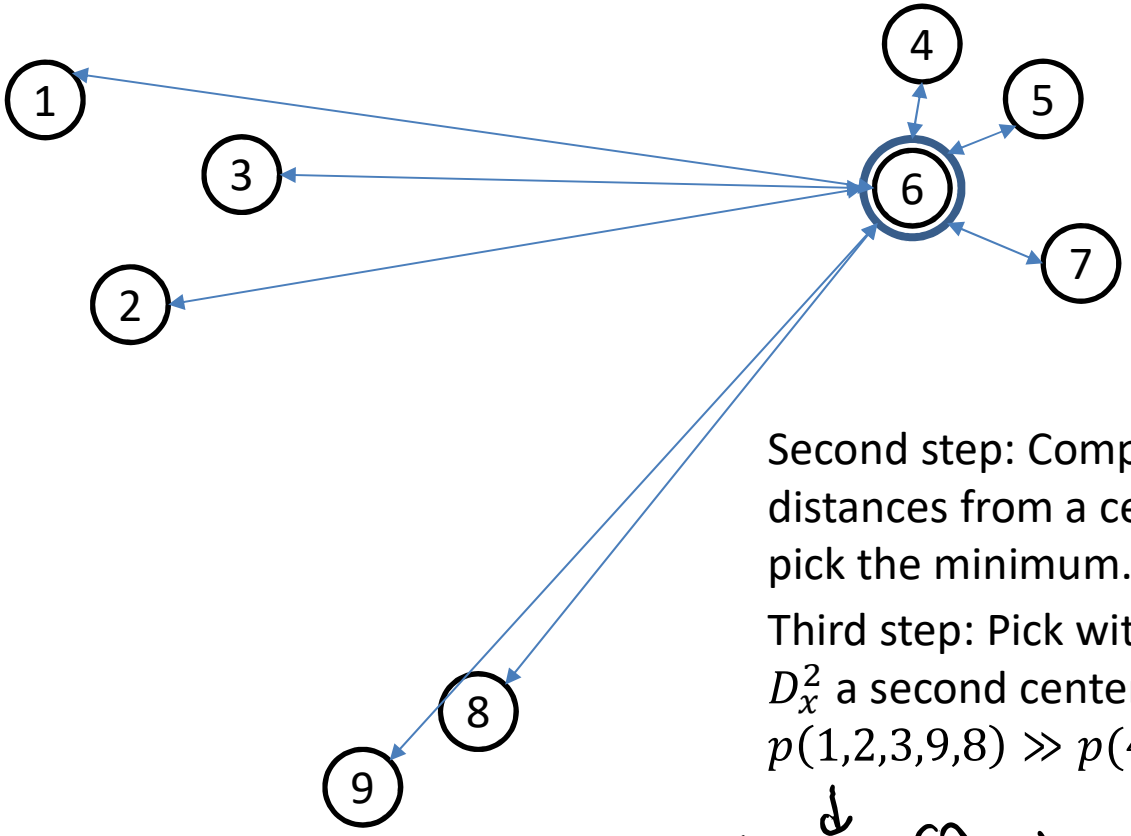
K-means ++



K-means ++


1
2
3
4
5
6
7
8
9
First step: Randomly (uniform p(x)) pick one data point.

K-means ++



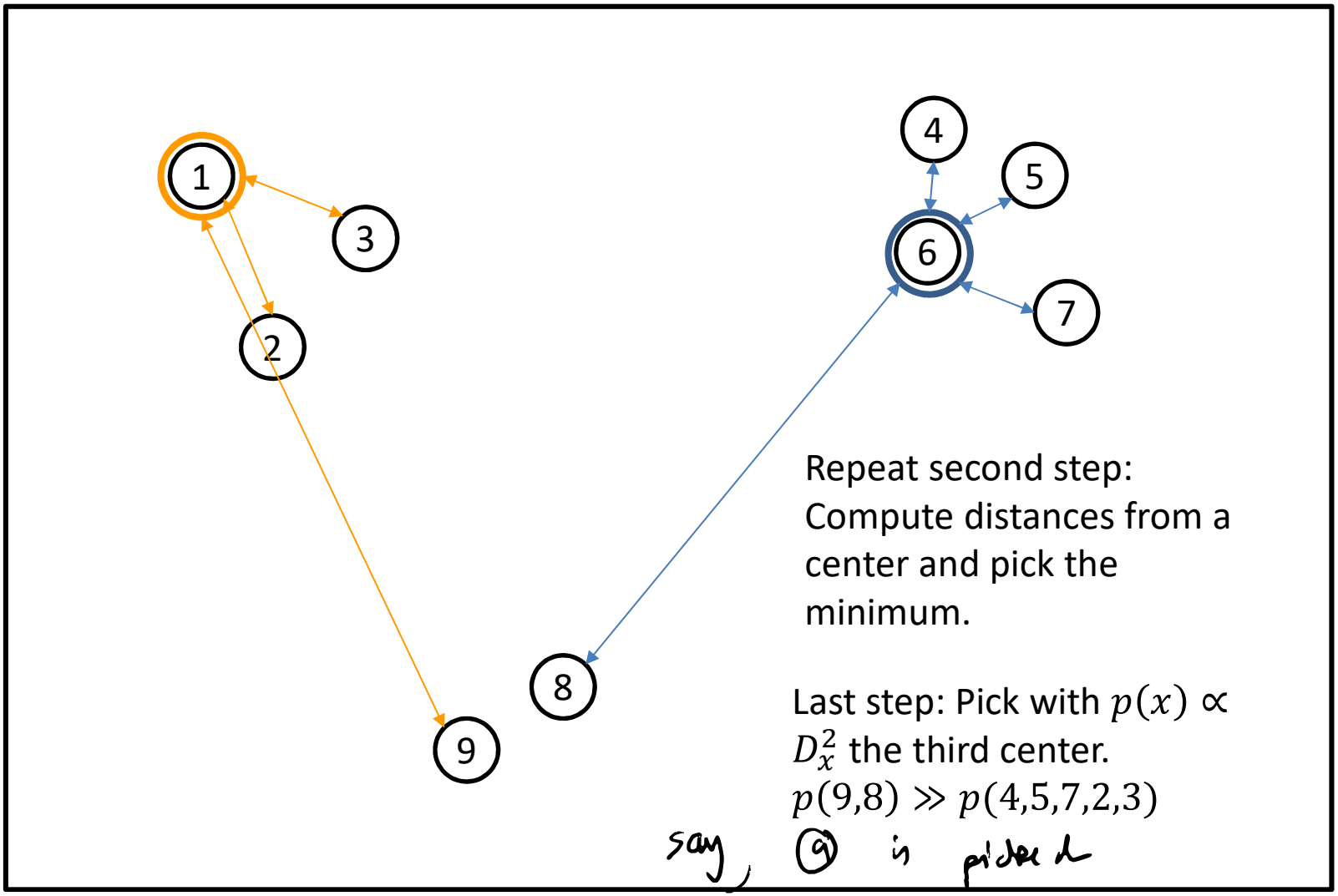
Second step: Compute distances from a center and pick the minimum.

Third step: Pick with $p(x) \propto D_x^2$ a second center.

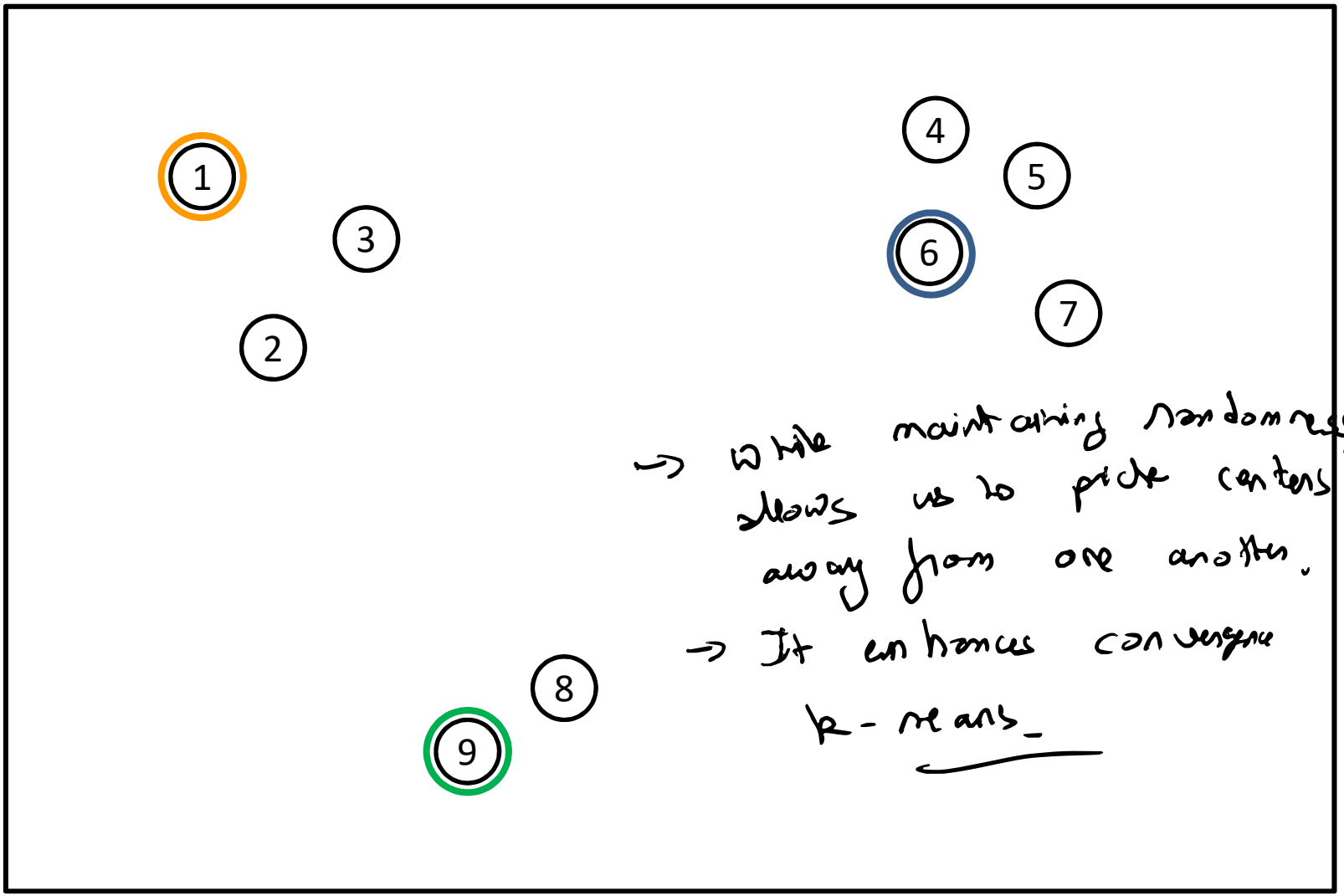
$p(1,2,3,9,8) \gg p(4,5,7)$

let's say (1) is picked.

K-means ++


1
3
2
9
8
4
5
6
7
Repeat second step:
Compute distances from a center and pick the minimum.
Last step: Pick with $p(x) \propto D_x^2$ the third center.
$p(9,8) \gg p(4,5,7,2,3)$
say 9 is picked

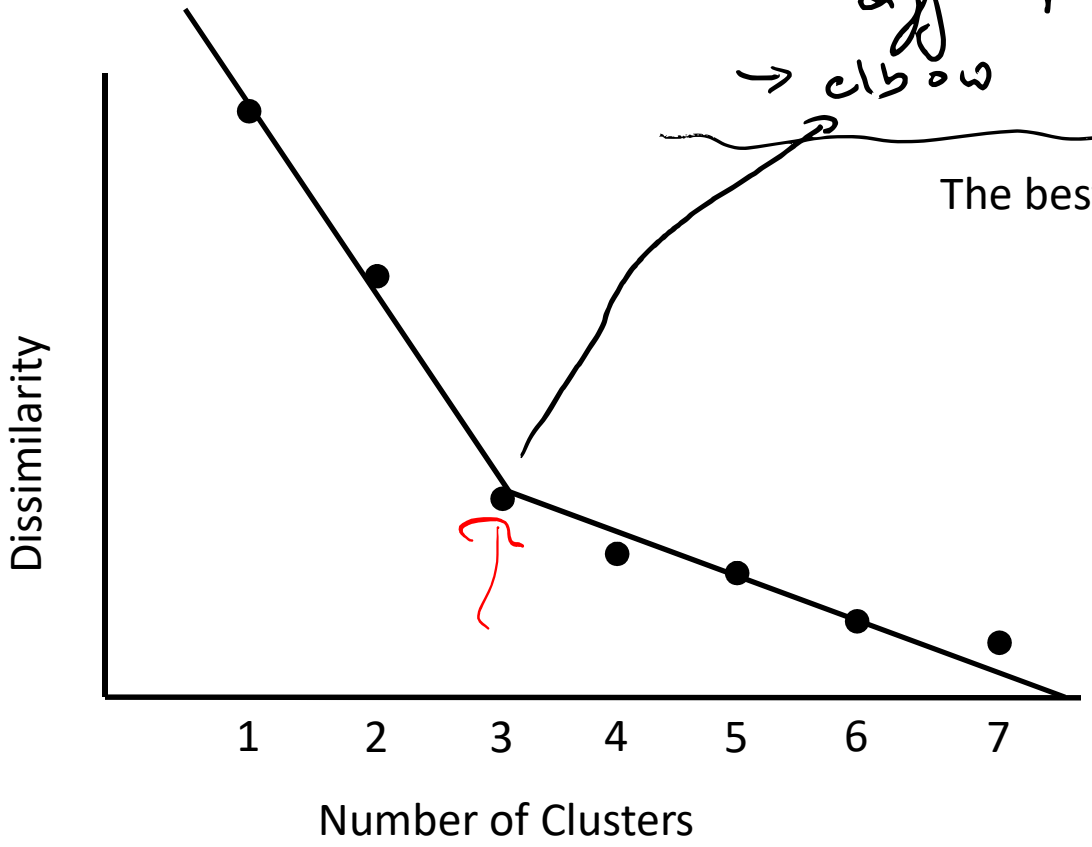
K-means ++


1
2
3
4
5
6
7
8
9
→ While maintaining randomness, it allows us to pick centers far away from one another.
→ It enhances convergence of k-means.

K-means weakness

- Initialization sensitive (local optimization) →k-means++
- Which k-employing →Scree test (better methods exist)
- Sensitive to outliers →k-medoids
- employs Euclidean distance →k-medoids
- Only for spherical clusters →kernel k-means (advanced)

Scree test for k-means



→ compute loss func = for diff 'k'

→ elbow indicates # of clusters

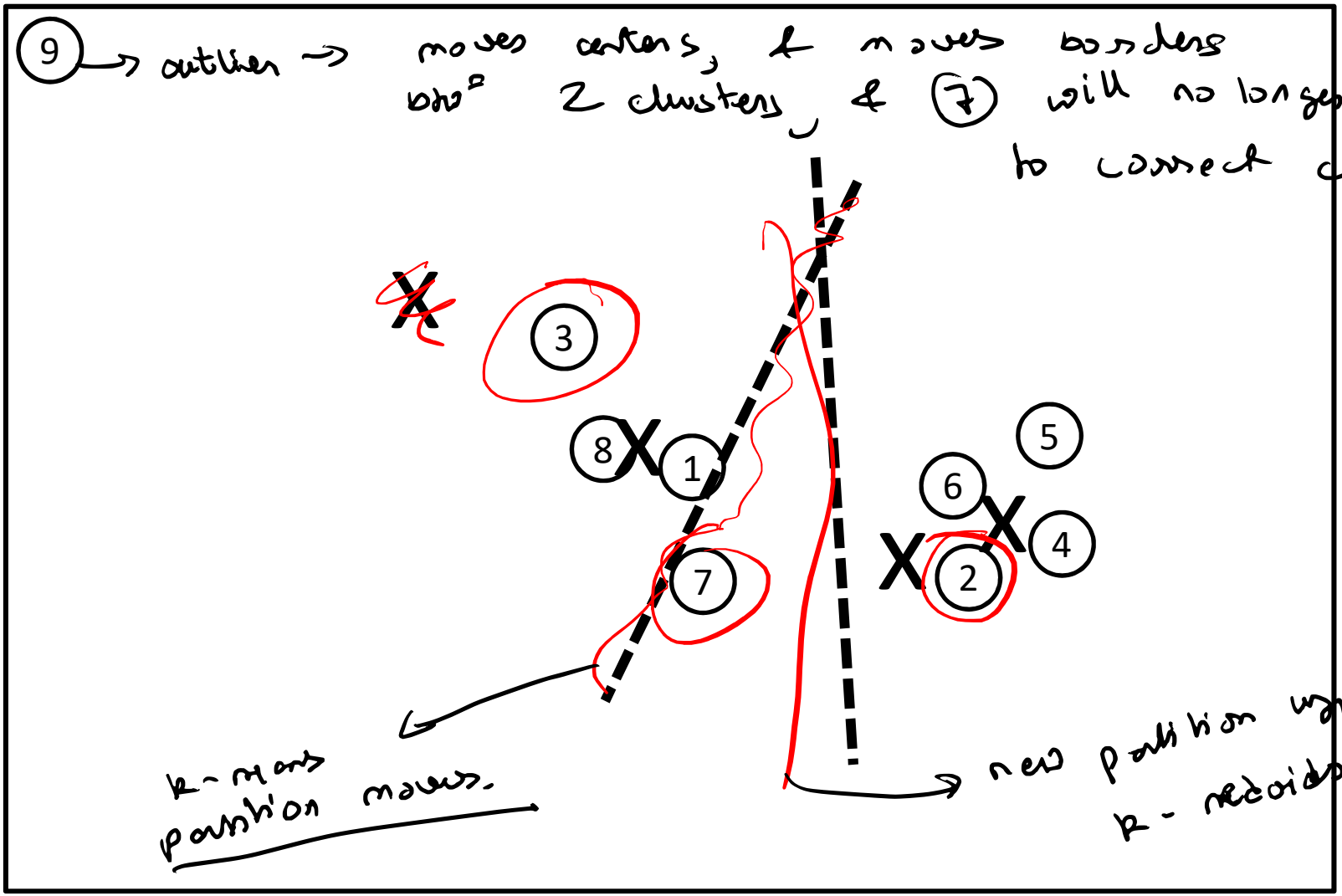
The best K is 3.

"k" not always so clear.

K-means weakness

- Initialization sensitive (local optimization) →k-means++
- Which k-employing →Scree test
- Sensitive to outliers →k-medoids
- employs Euclidean distance →k-medoids
- Only for spherical clusters →kernel k-means (advanced)

Sensitivity to outliers



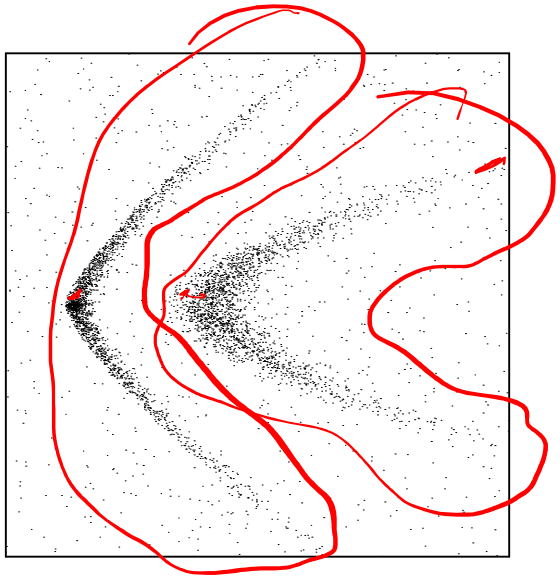
K-medoids

- It can be used with all kind of distances.
- Reduces the impact of outliers.
- Instead of working with centroids, it works with medoids, i.e. the most central element of the cluster.
- The cluster medoid is the element with minimum sum of distances to the rest of the elements of the cluster

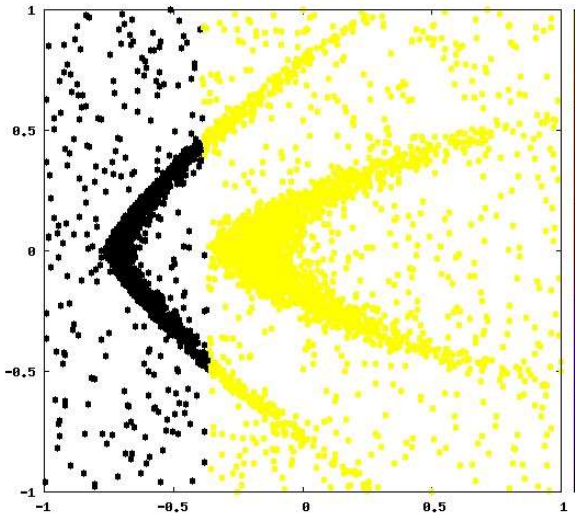
K-means weakness

- Initialization sensitive (local optimization) →k-means++
- Which k-employing →Scree test
- Sensitive to outliers →k-medoids
- employs Euclidean distance →k-medoids
- Only for spherical clusters →kernel k-means (advanced)

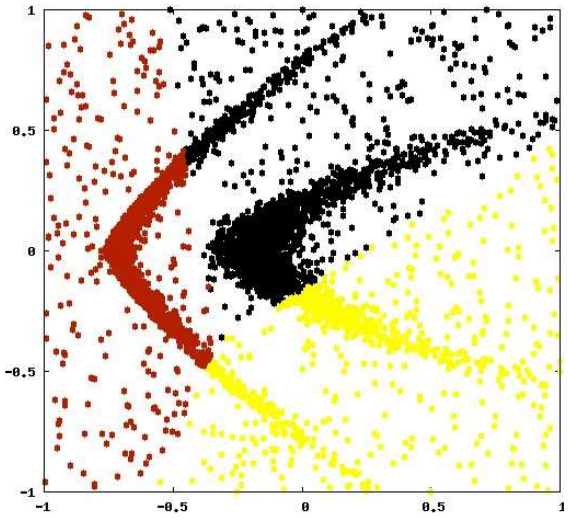
→ k-means / medoids etc will never obtain the correct clustering partition in such a case with this def^n of loss func?



K=2



K=3



K=4