Master in
High Performance Computing

Unsupervised Machine Learning

Foundations

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

The course:

- Overview of Machine Learning, some critical concepts and techniques.
- Manifold Learning (Dimensionality reduction and Intrinsic dimension estimation) and density estimation.
- Clustering.

What about you?

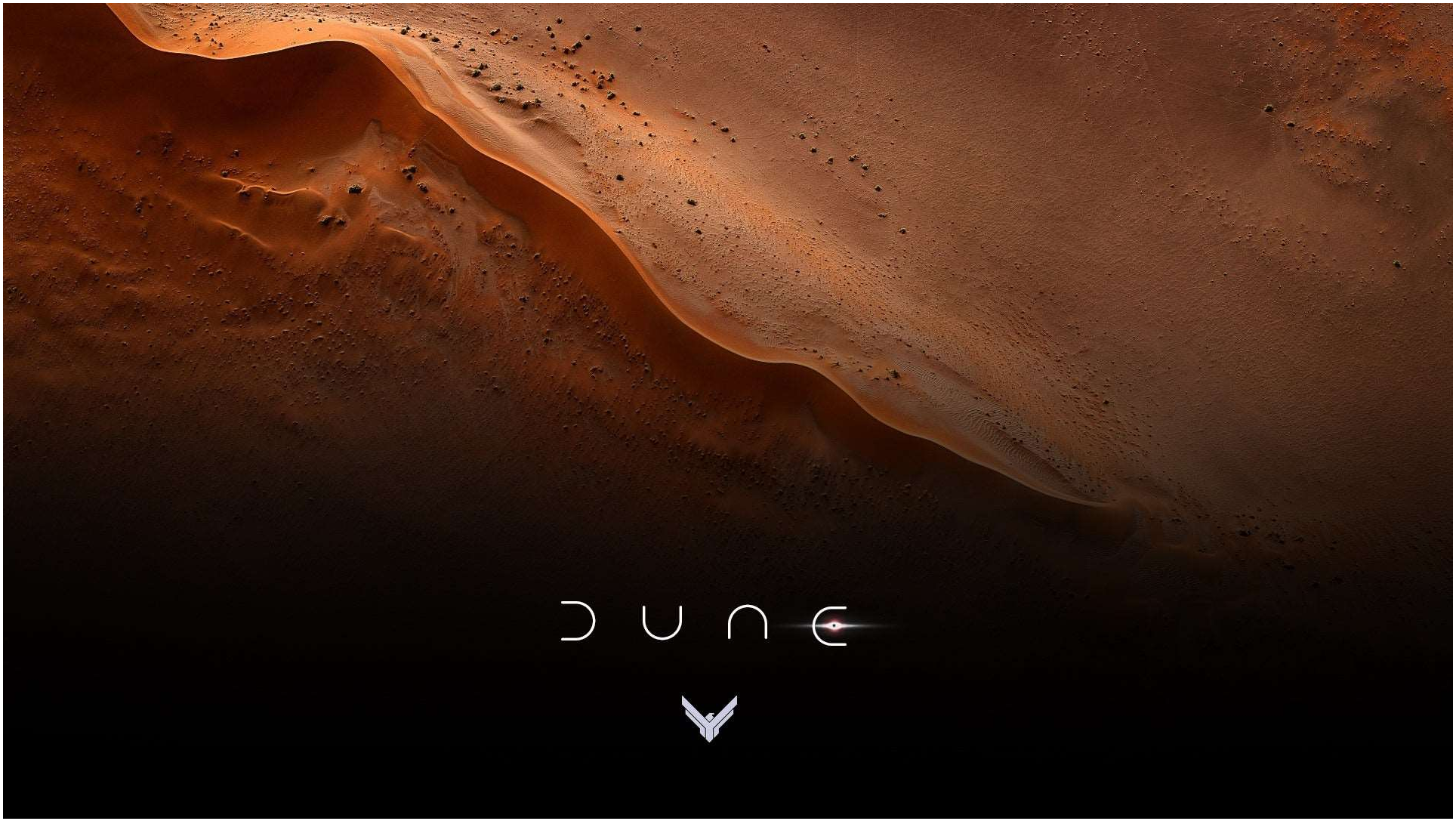
Master in High Performance Computing

Overview of Machine Learning

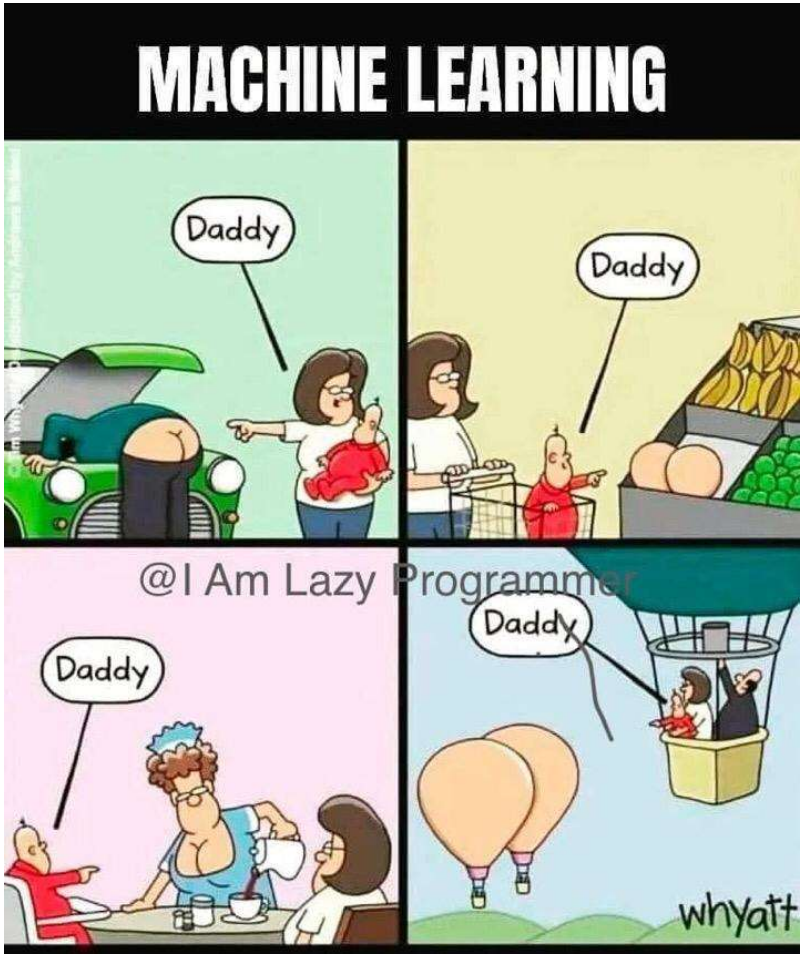
Method classification and terminology.

“Thou shalt not make a machine to counterfeit a human mind.”

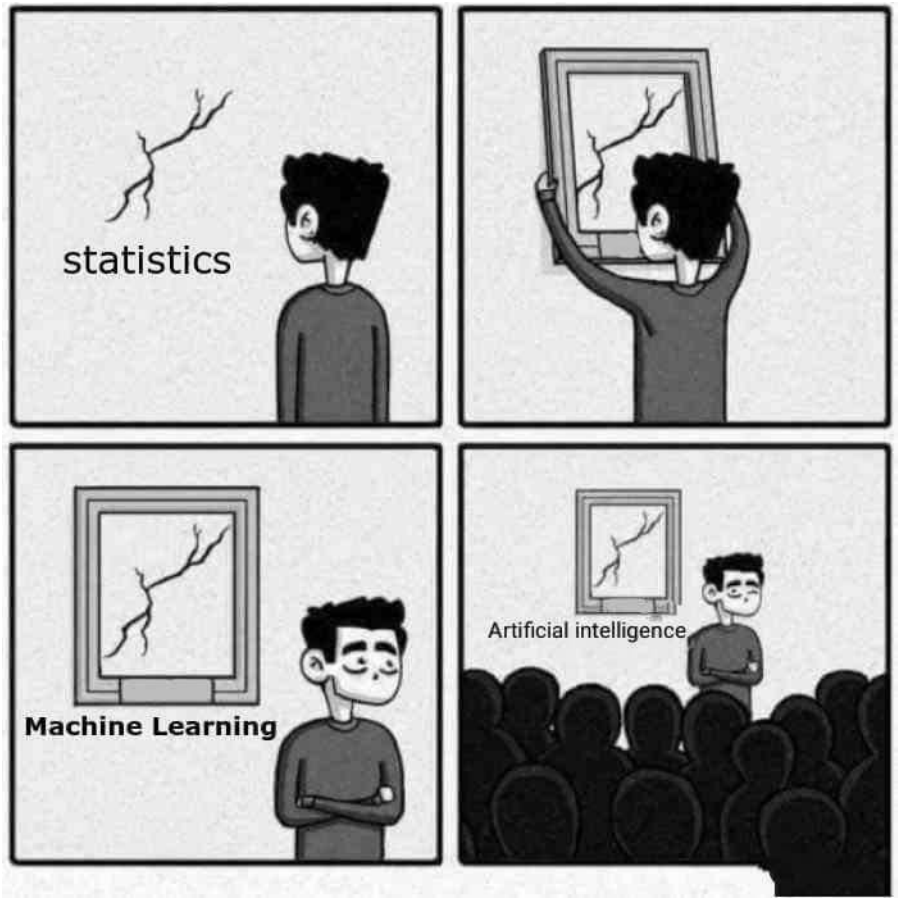
Frank Herbert



Let’s lower expectations



What is Machine Learning?



What is Machine Learning?

Algorithms that infer information from the data in an implicit way.

What is Machine Learning?

Algorithms *that infer information from the data in an implicit way.*

- Machine Learning is performed by computers, but we don't care too much about *which* computers.

What is Machine Learning?

Algorithms *that* ***infer information*** *from the data in an implicit way.*

- Machine Learning is performed by computers, but we don't care too much about *which* computers.
- Infer information==Learning. It can take many forms.

What is Machine Learning?

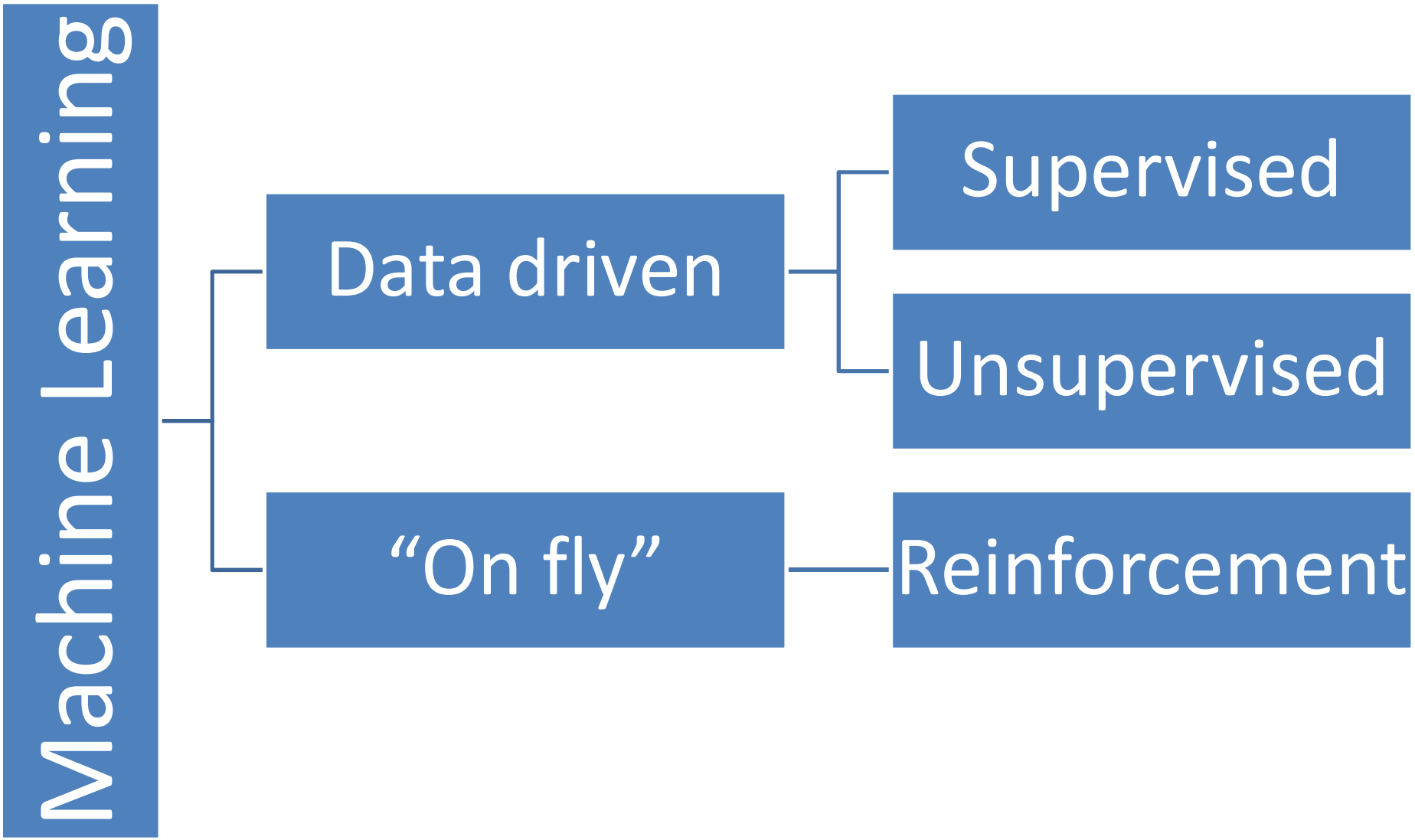
***Algorithms** that **infer information** from the **data** in an implicit way.*

- Machine Learning is performed by computers, but we don't care too much about *which* computers.
- Infer information==Learning. It can take many forms.
- For learning you need data. Again it can take many forms.

What is Machine Learning?

***Algorithms** that **infer information** from the **data** in an **implicit way**.*

- Machine Learning is performed by computers, but we don't care too much about *which* computers.
- Infer information==Learning. It can take many forms.
- For learning you need data. Again it can take many forms.
- The way is implicit because is not information that comes directly with the data (is not that we are reading a book).


Machine Learning
Data driven
Supervised
Unsupervised
"On fly"
Reinforcement

Data driven Machine Learning

Supervised vs *unsupervised* Machine Learning

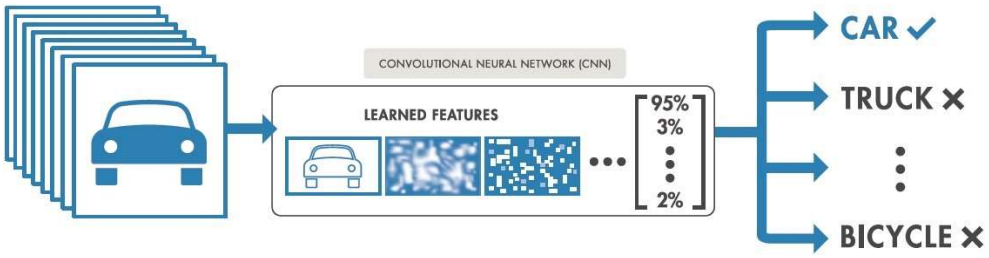
- ***Supervised* Learning**
  - From a learning data set, with a known response, builds a model.
  - Performs predictions about the response of new data based on this model.

Set of $\{\overrightarrow{x_i}, g_i\}$, (features and ground truth properties)

$$f(\boldsymbol{w}, \vec{x}) = g$$

Learn $\boldsymbol{w}$ from $\{\overrightarrow{x_i}, g_i\}$

Apply $f(\boldsymbol{w}, \vec{x})$ with the learned $\boldsymbol{w}$ to new data $\{\overrightarrow{x_j}\}$



Supervised vs *unsupervised* Machine Learning

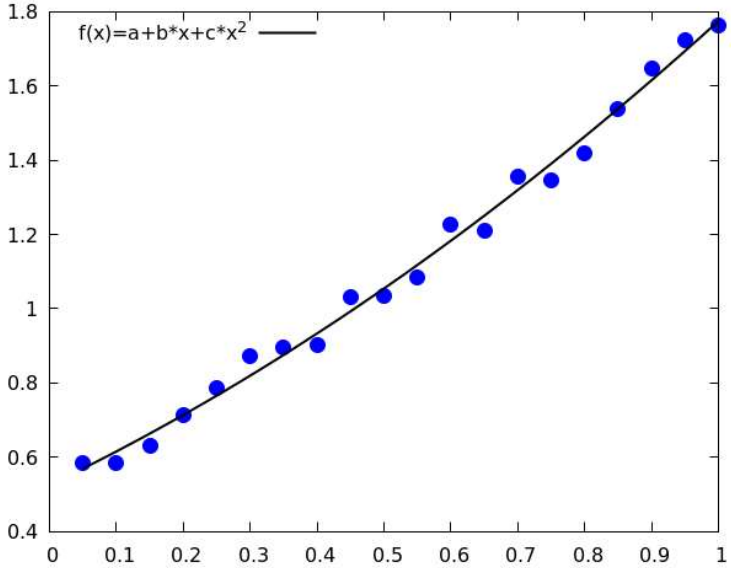
- ***Supervised* Learning**
  - From a learning data set, with a known response, builds a model.
  - Performs predictions about the response of new data based on this model.

$$\{\overrightarrow{x_i}, g_i\}$$

- ***Unsupervised* Learning**
  - Attempts to devise the inner structure of the data set, without any known response.

$$\{\overrightarrow{x_i}\}$$

Simplest algorithms in data driven learning: Linear Regression



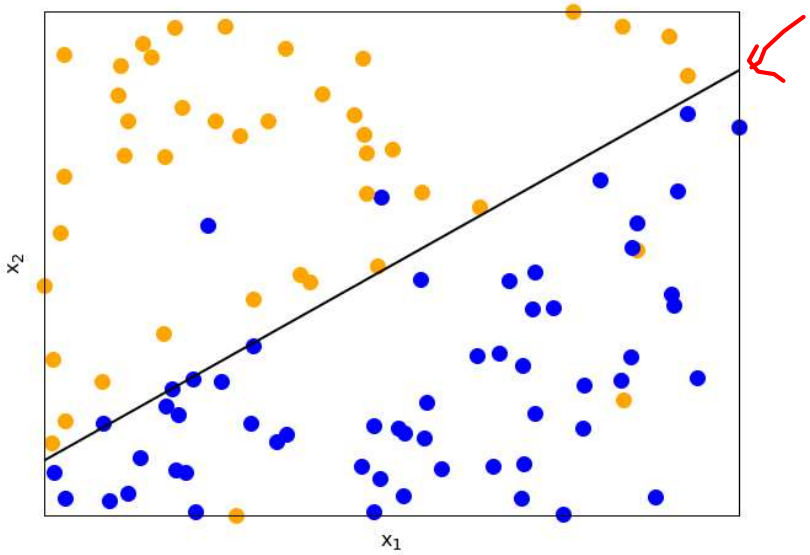
$\vec{x} : \rightarrow x_i$

$f(a, b, c | x_i)$

$\sum (y_i - f)^2 \rightarrow$ Metrix

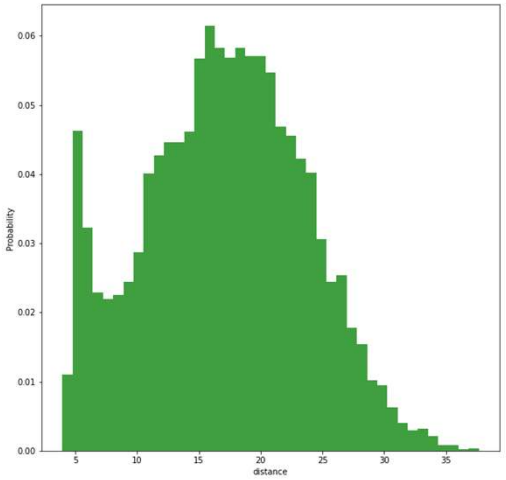
Solution

Simplest algorithms in data driven learning: Logistic Regression



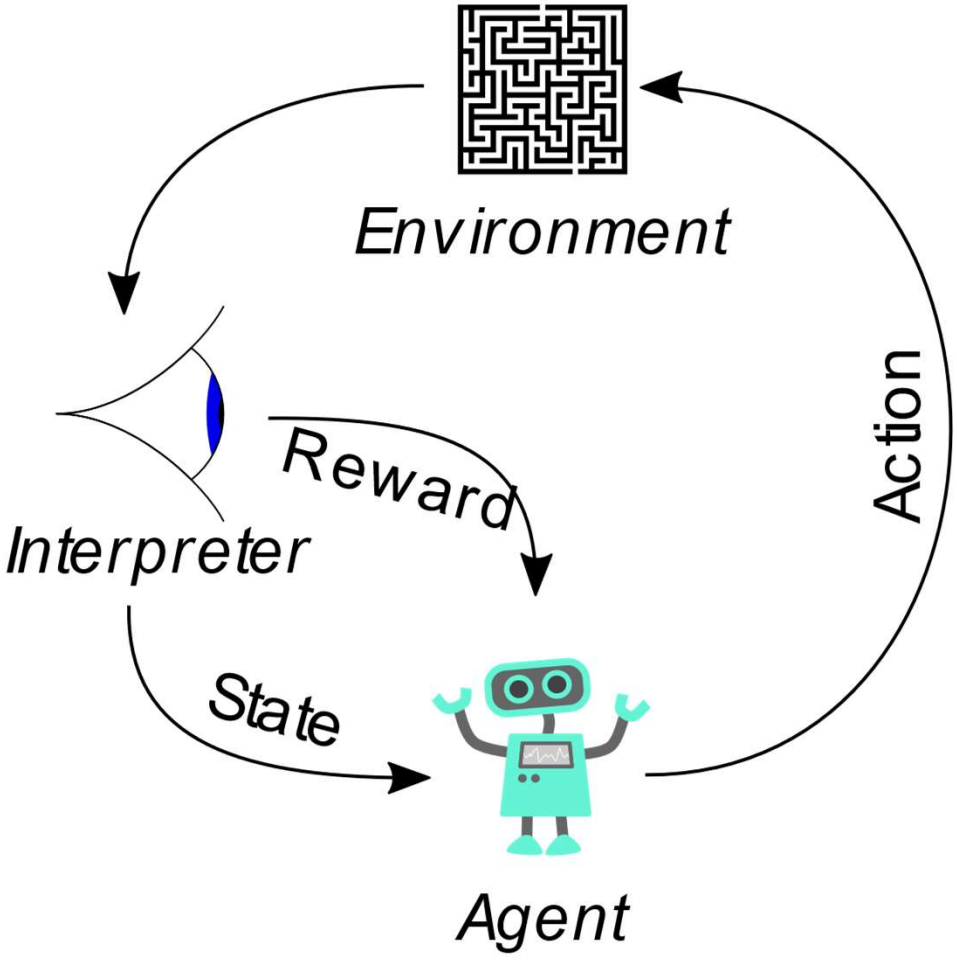
$$p(b) = \frac{e^{-\beta_0 - \beta_1 x}}{1 + e^{-}}$$

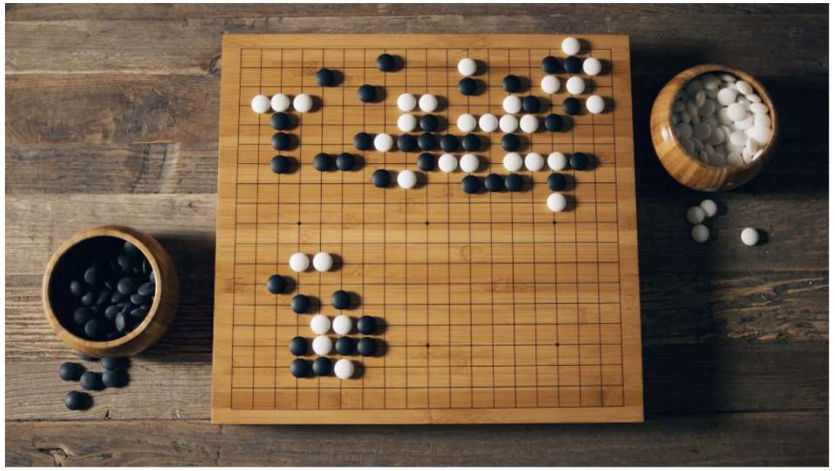
Simplest algorithms in data driven learning: Histograms



“On fly” Machine Learning

Reinforcement Learning





By Megajuice - Own work, CC0,
https://commons.wikimedia.org/w/index.php?curid=57895741

Basic Concepts

- Data, task
- Curse of dimensionality
- Loss, train/test
- Generalization, overfitting
- Bias/variance trade off

The data and the task

- How is a data set?

$\vec{x}_1 \quad \vec{x}_2 \quad \dots \quad \vec{x}_N$

- What do we want it for?

Classification

Regression

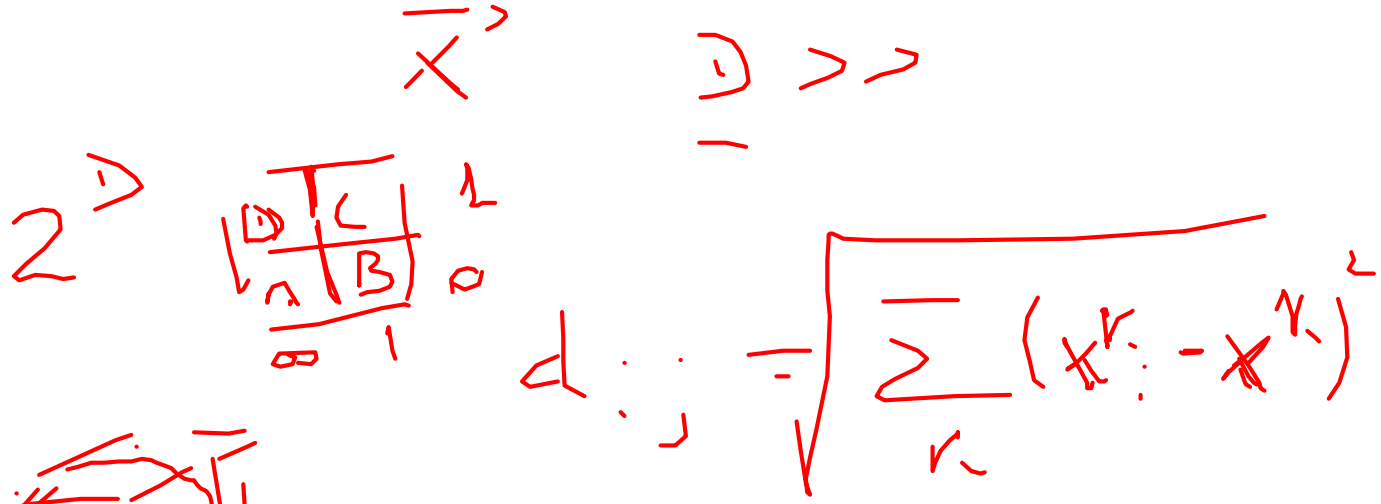
Projection

Density

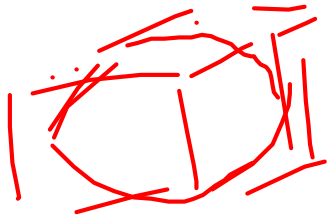
Clustering

The curse of dimensionality

Combinatorics



Distance



Sampling

The Loss function

Mean Absolute Error

$$L = \frac{1}{N} \sum |f(x) - g|$$

Mean Squared Error

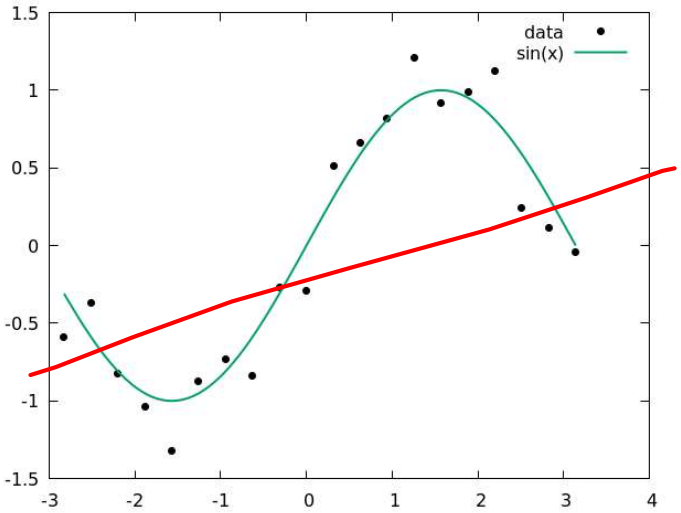
$$L = \frac{1}{N} \sum (f(x) - g)^2$$

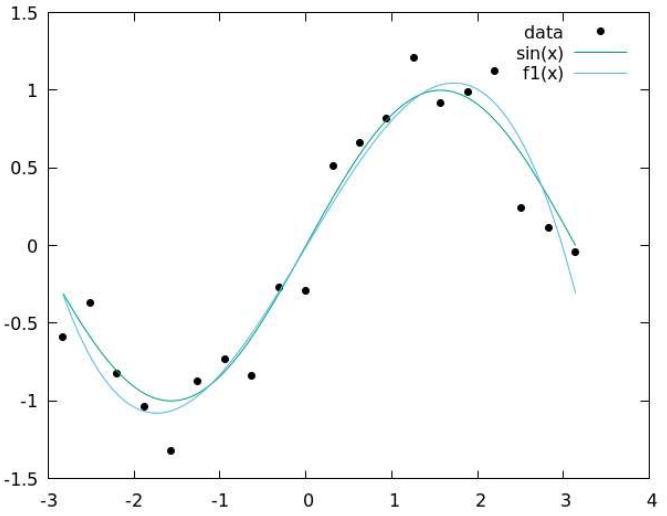
Cross Entropy

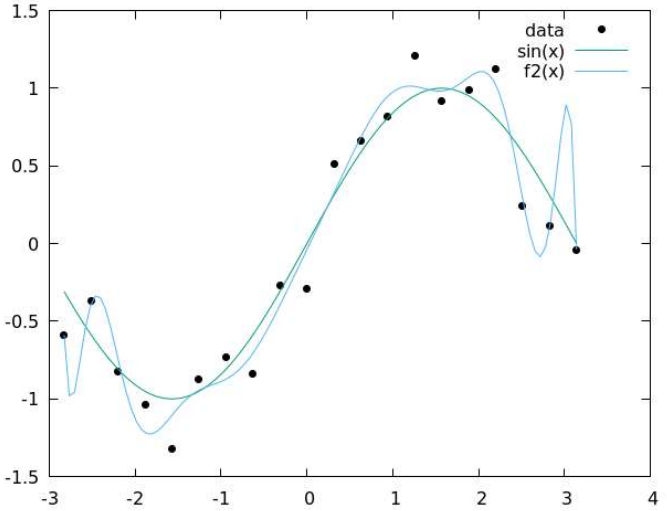
$$L = \sum \log(|R_w|)$$

Train, validation and test data sets

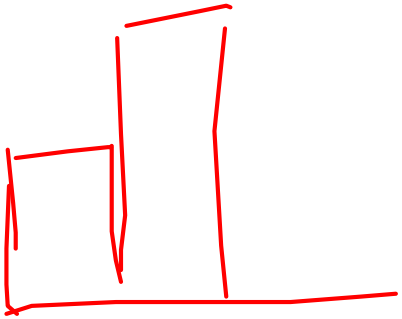
Over/Underfitting



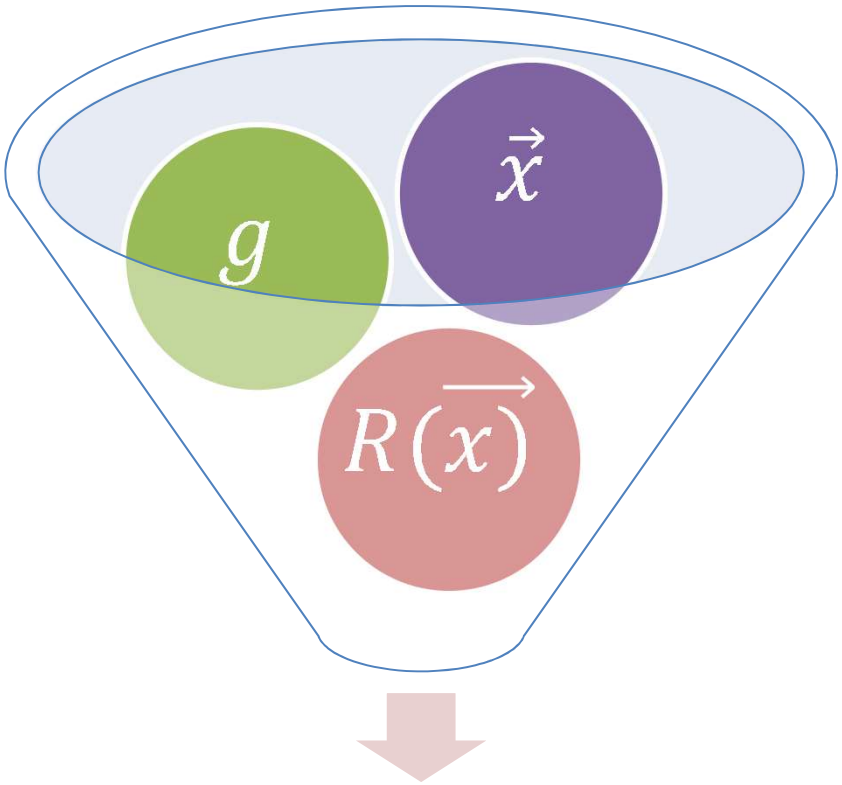




The bias/variance tradeoff



Algorithms



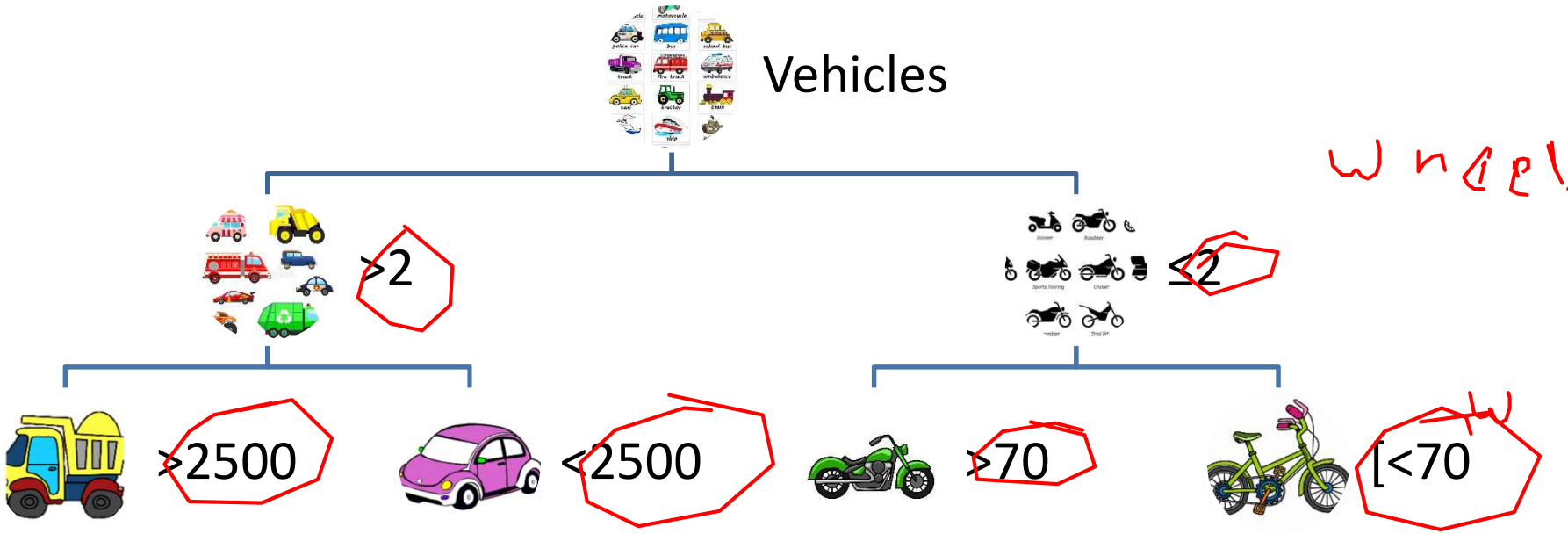
Knowledge

Algorithms for Supervised Learning:

- Classification/Regression:
  - Linear and Logistic regression
  - Tree based algorithms (random forest)
  - K-Nearest Neighbor
  - Kernel based algorithms
  - Dense/Convolutional Neural Networks

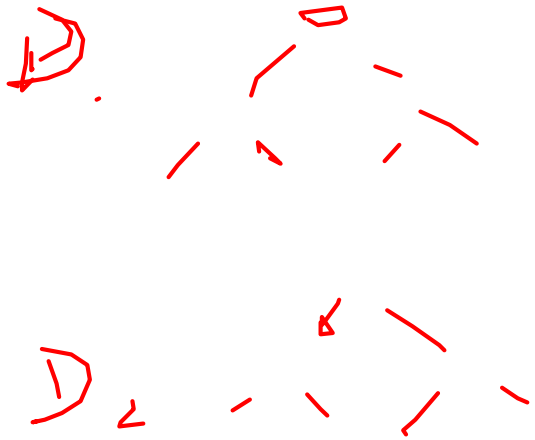
Tree based algorithms

Decision Trees

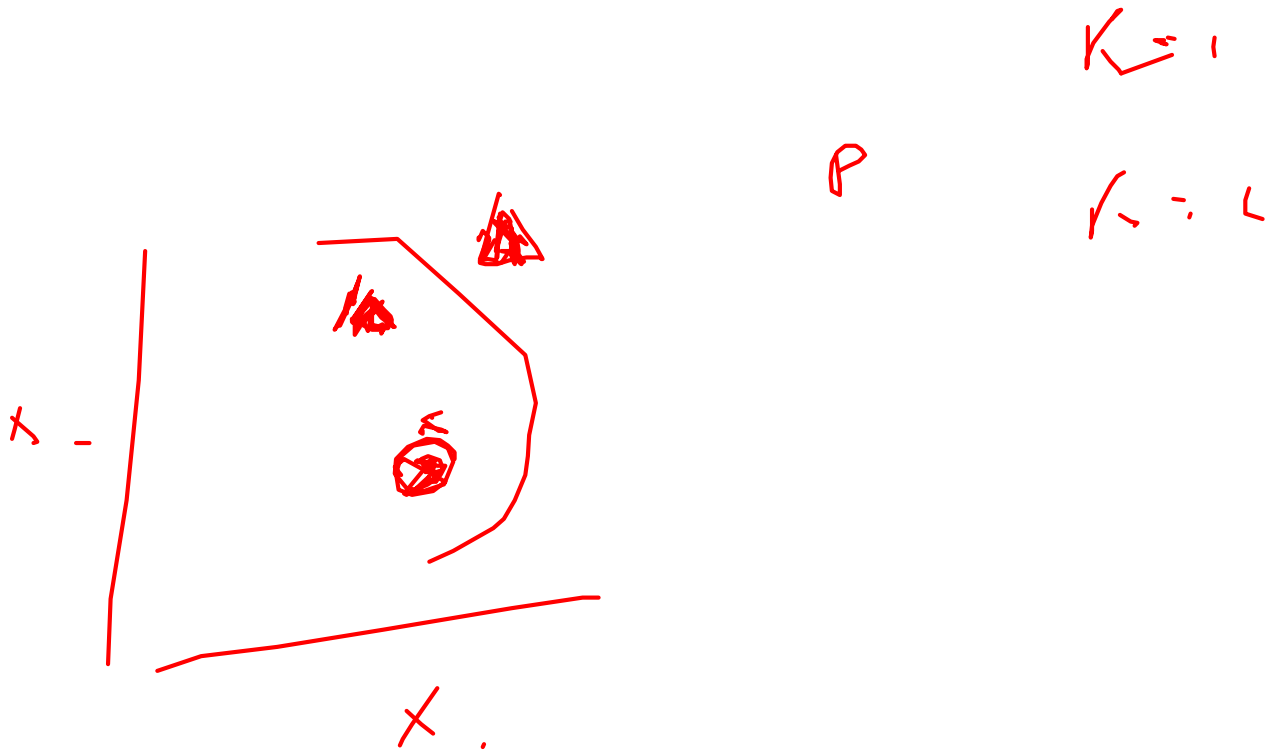


Tree based algorithms

Bagging and random forests



k-Nearest Neighbors



Kernel Based Methods

$$S_{ij} = k(\vec{x}_i, \vec{x}_j)$$

$$\exp\left(-\frac{d_{ij}^2}{\sigma}\right) =$$

$$\langle \phi_i \cdot \phi_j \rangle$$

$$x \rightarrow \phi$$

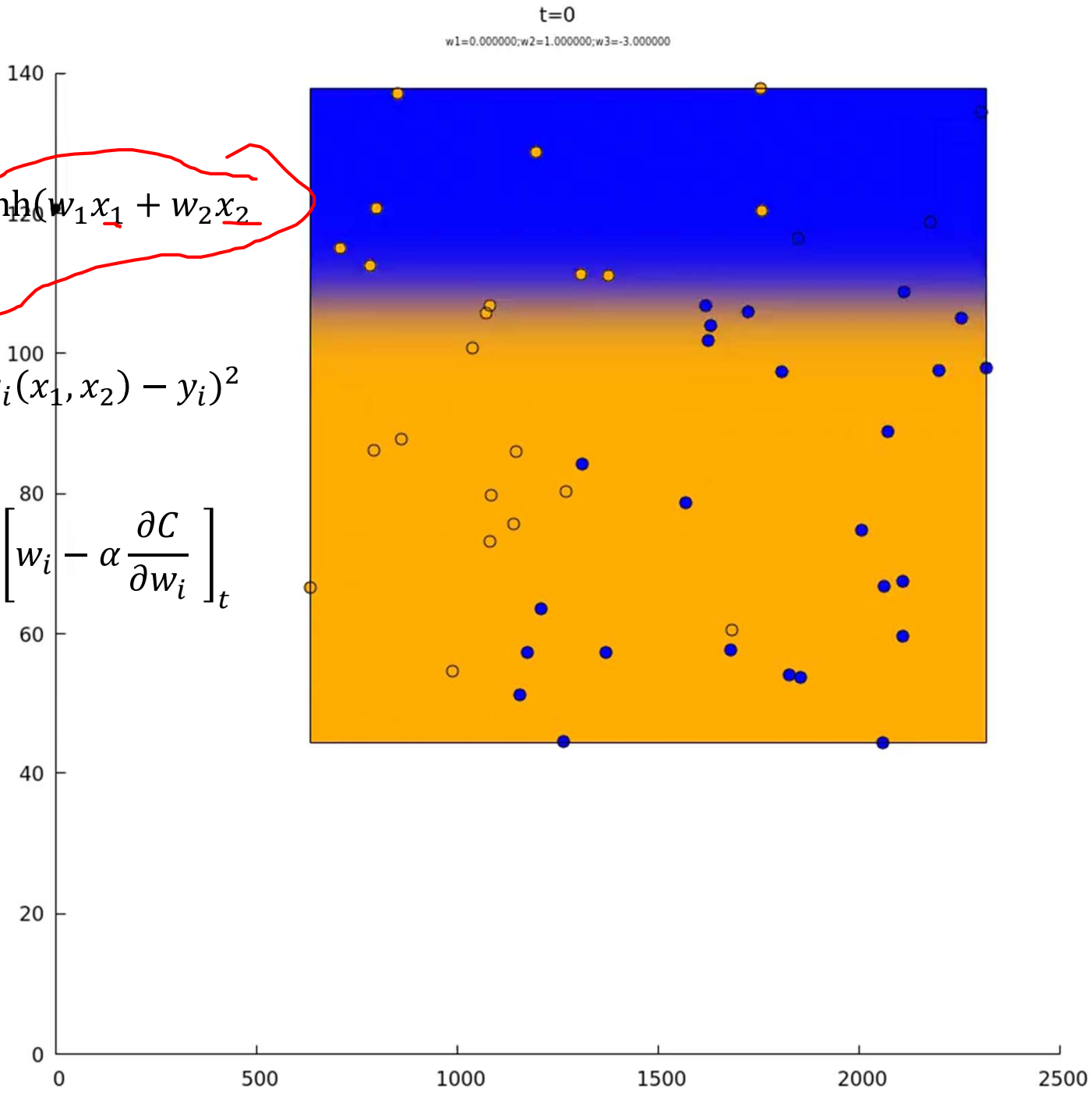
Neural Networks

NEURAL NETWORKS!!!

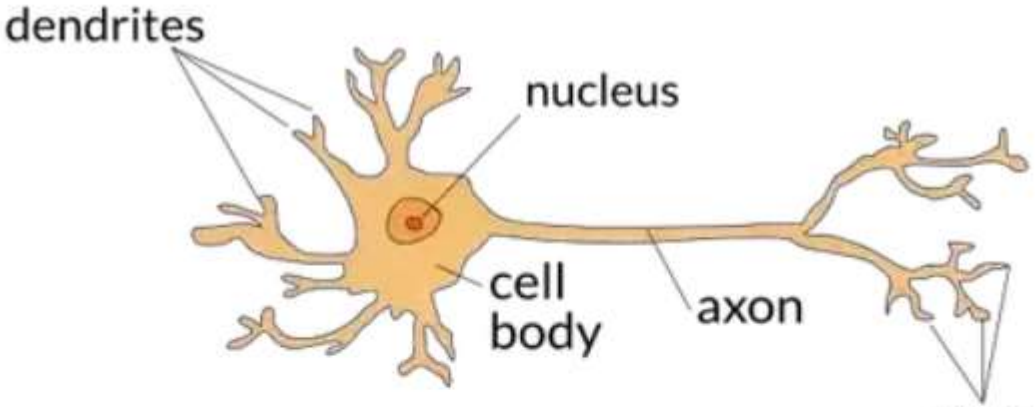
$$g(x_1, x_2) \tanh(w_1 x_1 + w_2 x_2 + w_3)$$

$$C = \sum_i (g_i(x_1, x_2) - y_i)^2$$

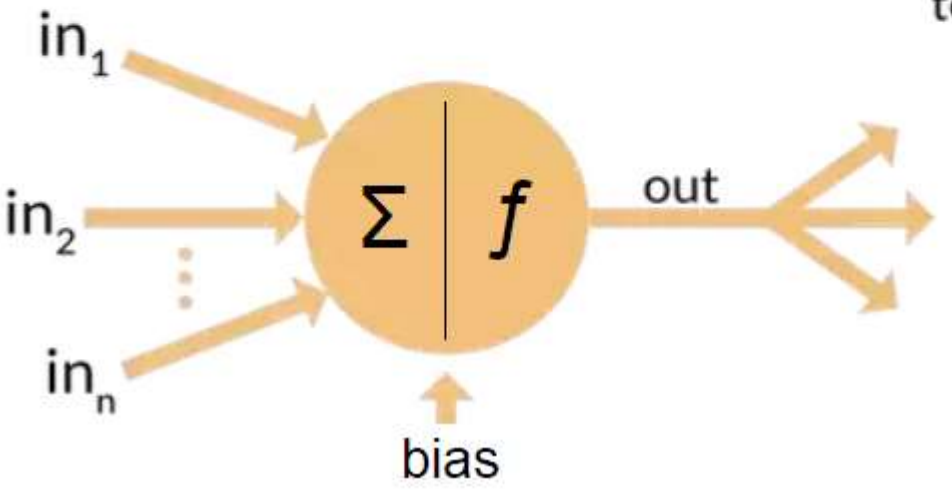
$$[w_i]_{t+1} = \left[w_i - \alpha \frac{\partial C}{\partial w_i} \right]_t$$



Natural vs Artificial Neurons

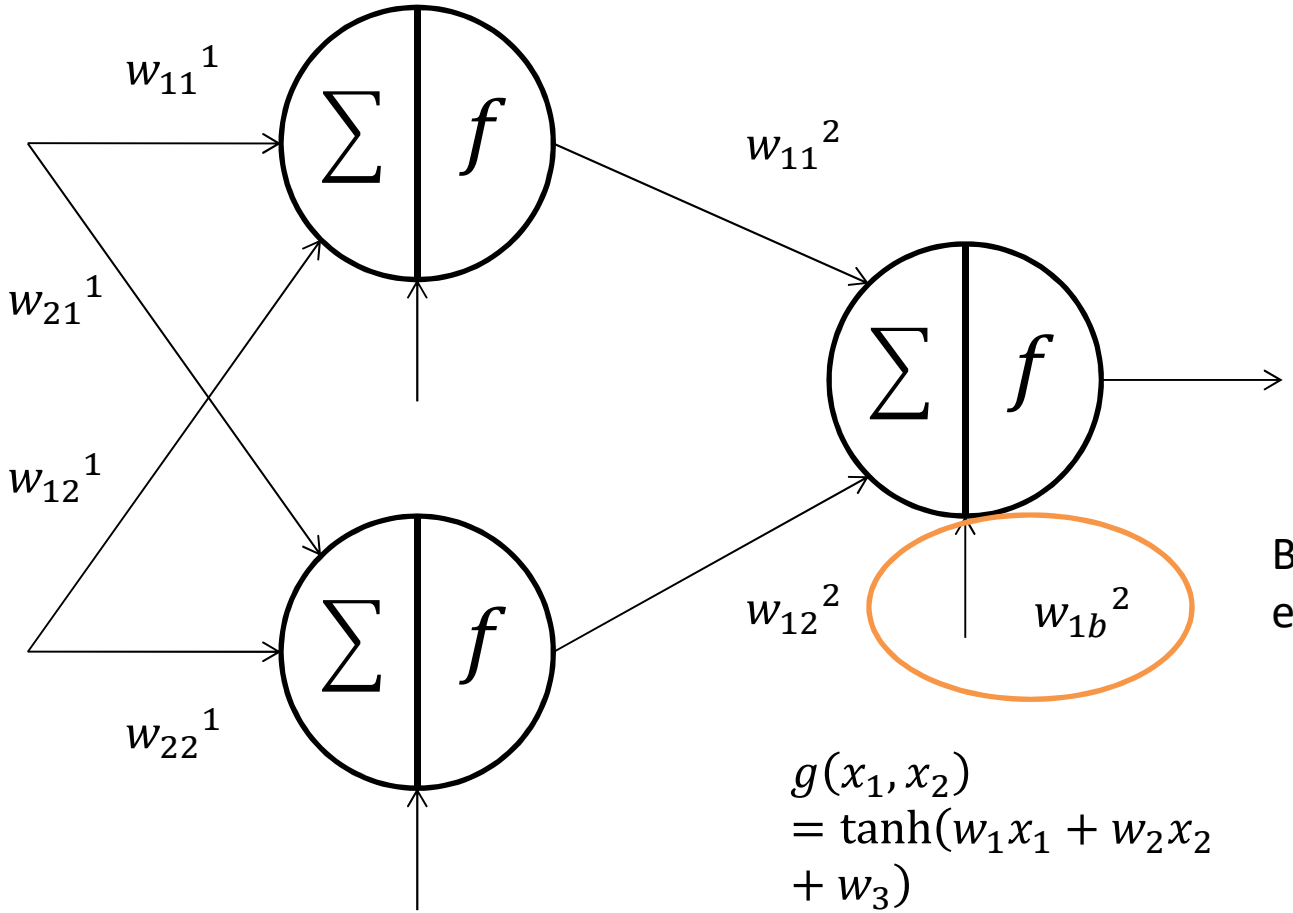


$$f_i(x_1, x_2) = \tanh(w_1 x_1 + w_2 x_2 + w_3)$$



Only hyperplanar separations

The next step: Towards non-linear separations

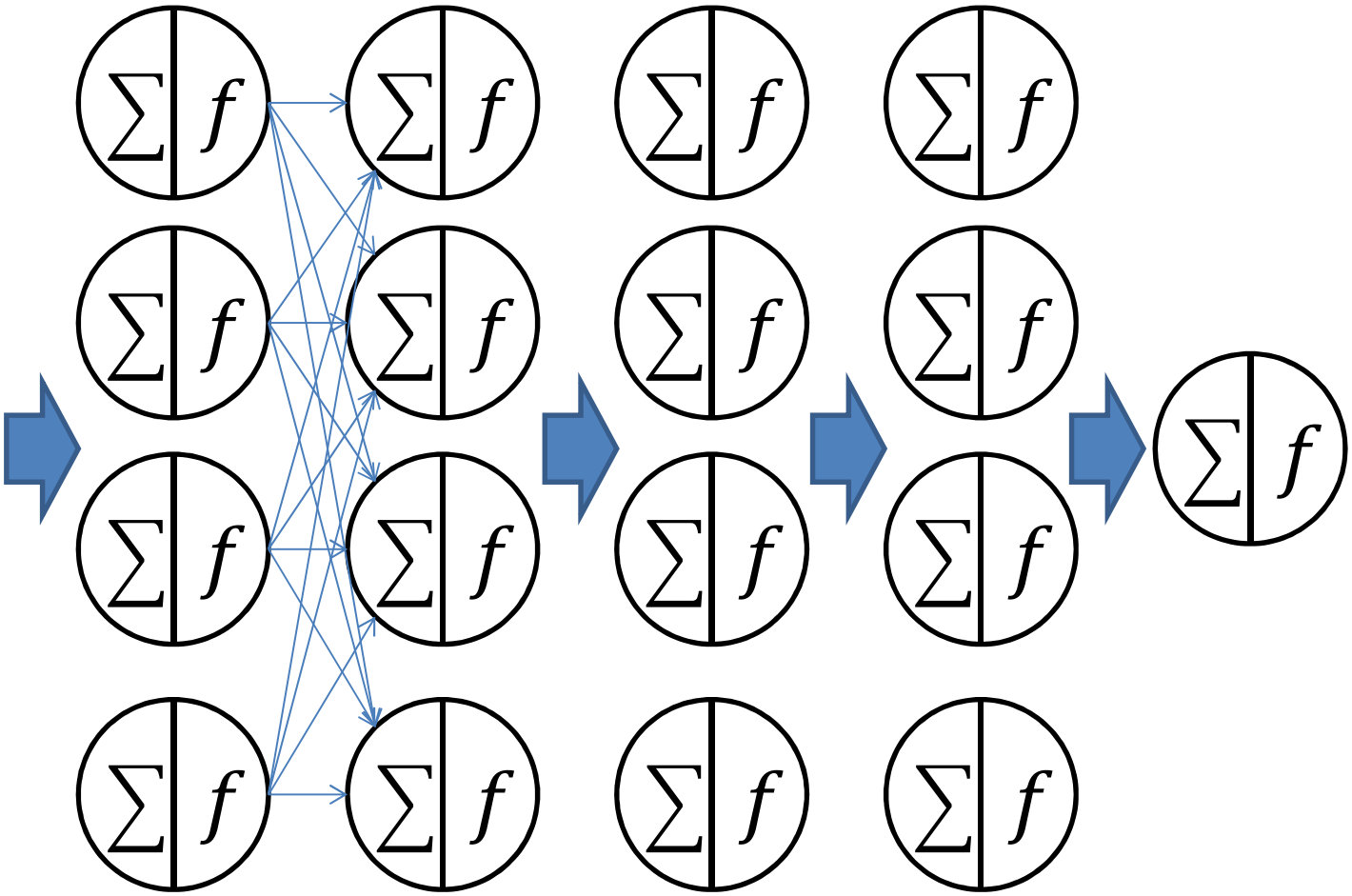


Bias term: Add an extra input =1

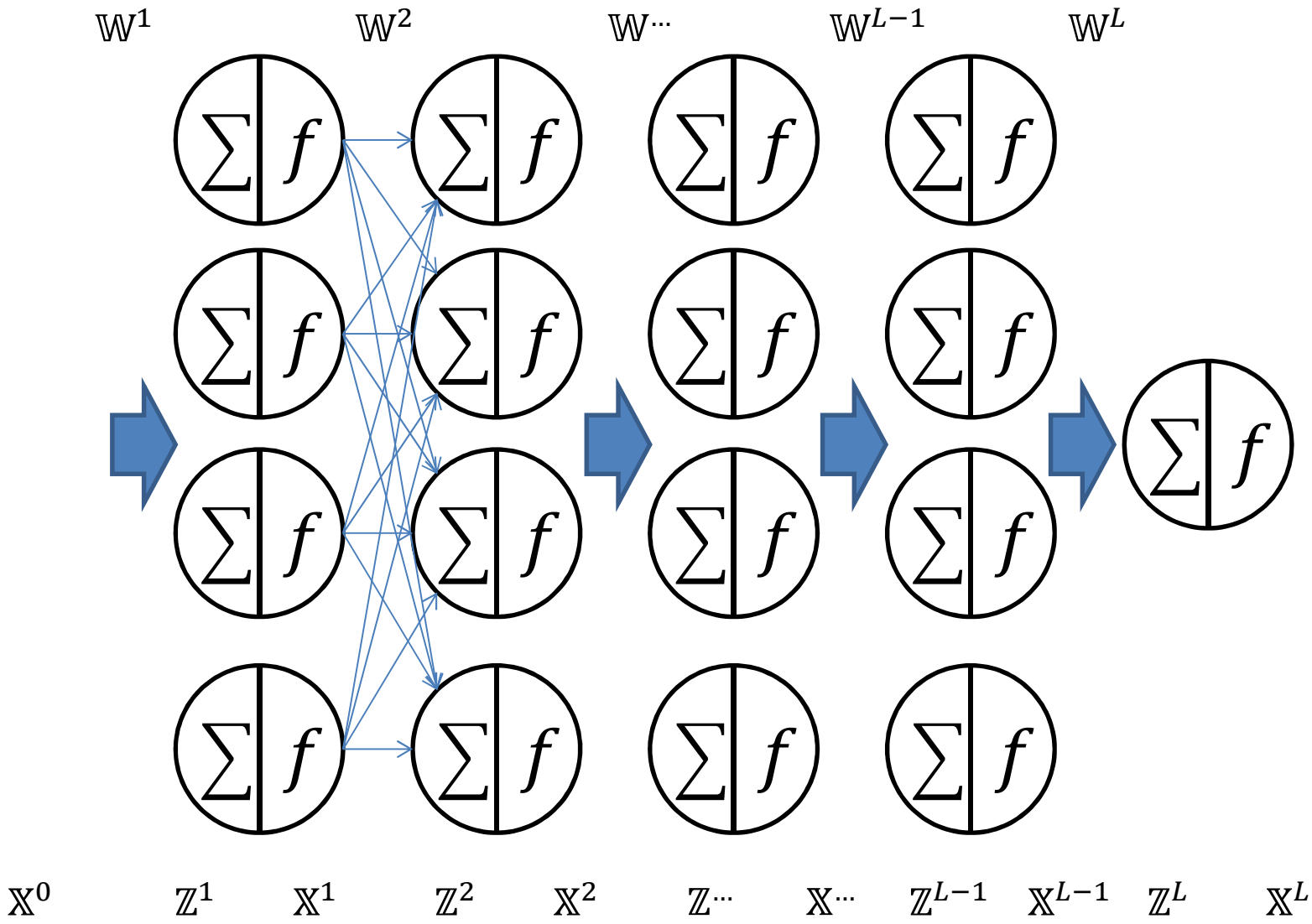
$$g(x_1, x_2) = \tanh(w_1 x_1 + w_2 x_2 + w_3)$$

$$g(x_1, x_2) = \tanh(w_{11}{}^2(\tanh(w_{11}{}^1 x_1 + w_{12}{}^1 x_2 + w_{1b}{}^1)\ ) + w_{12}{}^2(\tanh(w_{\ }{}^1 x_{\ } + w_{\ }{}^1 x_{\ } + w_{\ }{}^1)\ ) + w_{1b}{}^2)$$

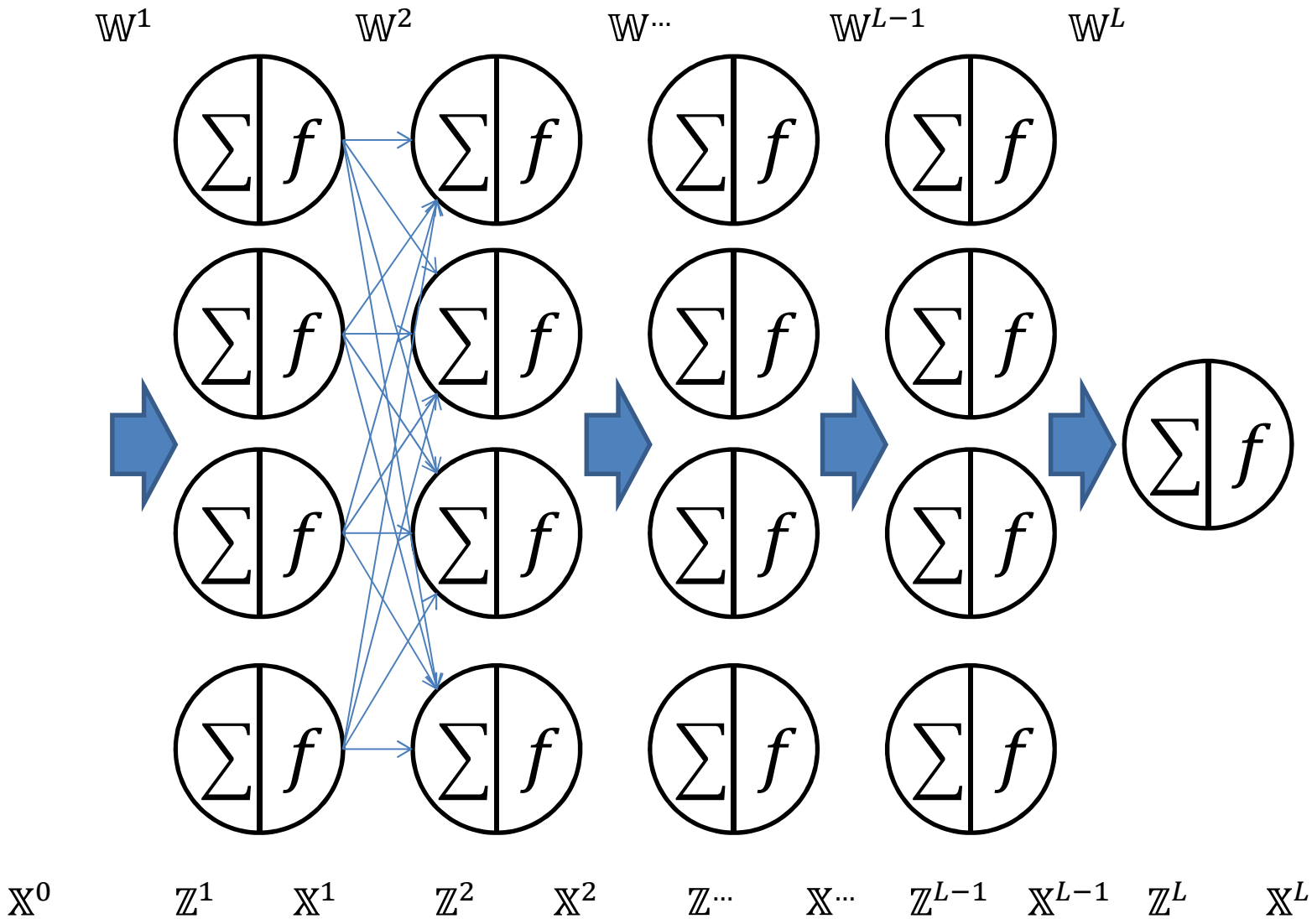
It can go messy really fast…



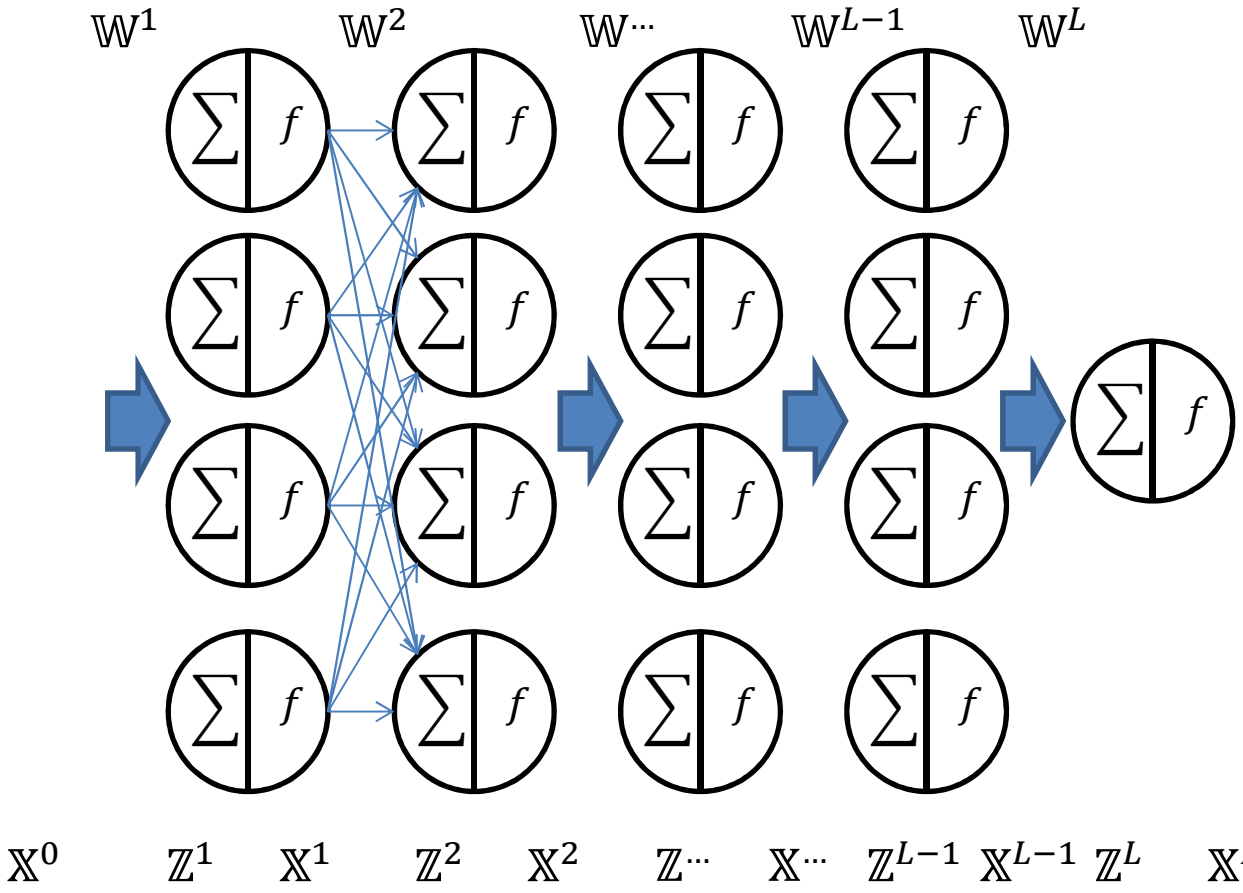
But we have MATRICES



But we have MATRICES



But we have MATRICES



$$g(\mathbb{X}^0) = \mathbb{X}^L = f(\mathbb{Z}^L) = f(\mathbb{X}^{L-1}\mathbb{W}^L) = f(f(\mathbb{Z}^{L-1})\mathbb{W}^L)$$
$$= f(f(\mathbb{X}^{L-2}\mathbb{W}^{L-1})\mathbb{W}^L) \ldots$$

Algorithms for Unsupervised Learning

- Manifold Learning (Dim. Red. & ID est.):
  - PCA
  - K-PCA
  - ISOMAP
  - t-SNE
  - Autoencoders
- Density Estimation:
  - Histograms
  - Kernel Density Estimation
  - k-Nearest Neighbor
  - Generative Adversarial NN
- Clustering:
  - k-means/c-means, kernel k-means, spectral clustering…
  - Hierarchical clustering
  - Density Based clustering
  - Self Organizig Maps

Master in High Performance Computing

Algorithms for Unsupervised Learning

Dimensionality Reduction and Density Estimation

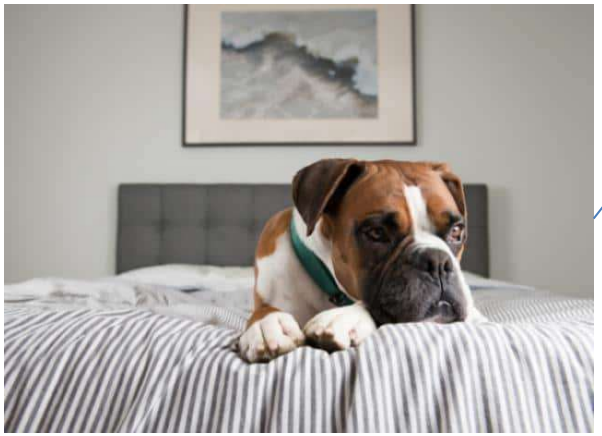
Algorithms for Unsupervised Learning

- Manifold Learning (Dim. Red. & ID est.):
  - PCA
  - K-PCA
  - ISOMAP
  - t-SNE
  - Autoencoders
- Density Estimation:
  - Histograms
  - Kernel Density Estimation
  - k-Nearest Neighbor
  - Generative Adversarial NN
- Clustering:
  - k-means/c-means, kernel k-means, spectral clustering…
  - Hierarchical clustering
  - Density Based clustering
  - Self Organizig Maps

Dimensionality reduction and Intrinsic Dimension estimation.

MANIFOLD LEARNING METHODS

A view from supervised learning



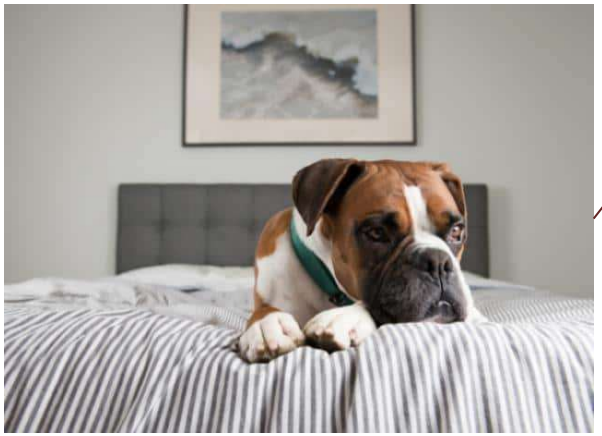
D=1280x960=1228800

Supervised ML

CAT

DOG

A view from supervised learning



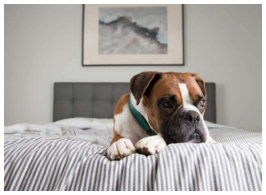
D=1280x960=1228800

Supervised ML

NATURE

HOME

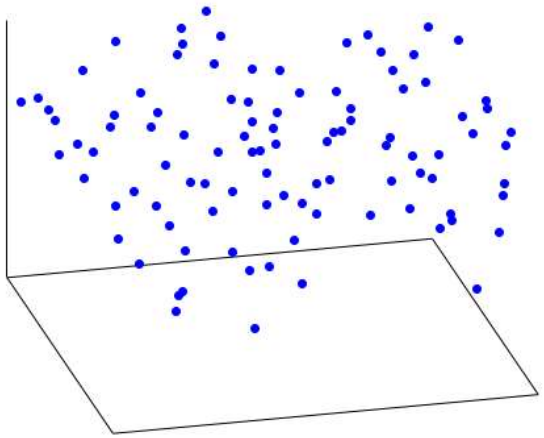
A view from supervised learning

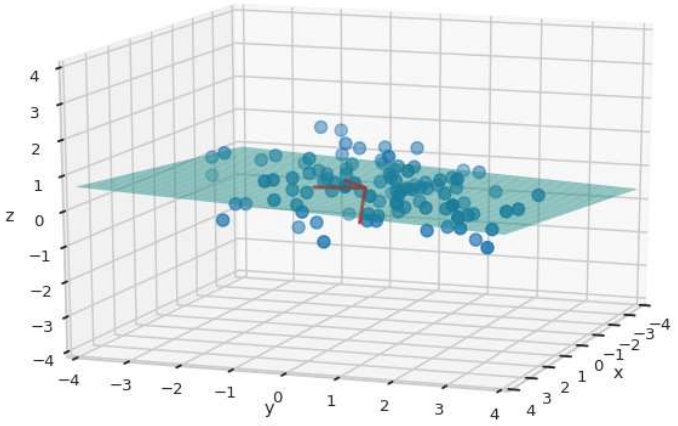


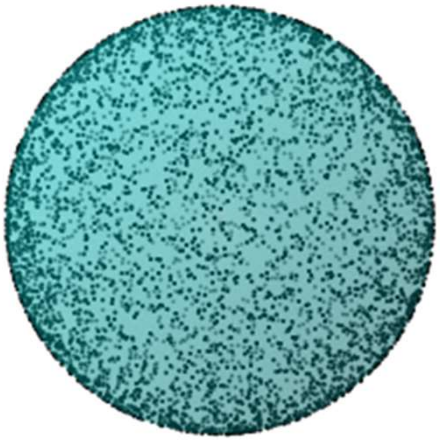
D=1280x960=1228800

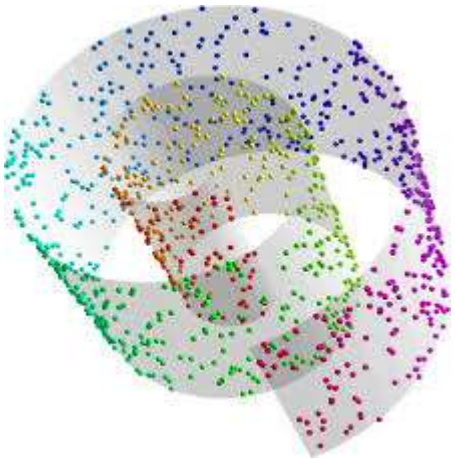
The data lay in a manifold whose dimension is the total number of INDEPENDENT classification tasks that can be performed.

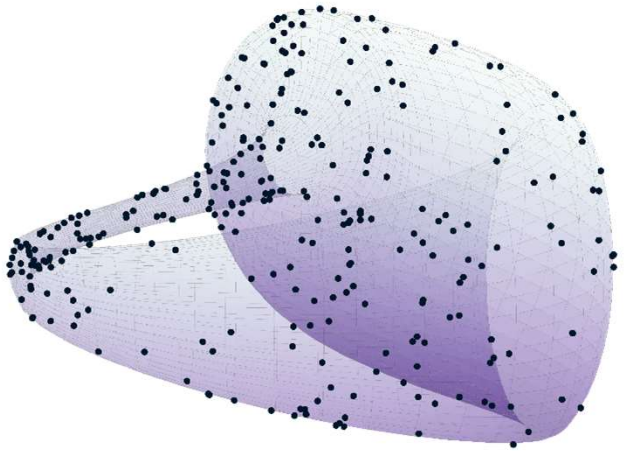
MANIFOLDS



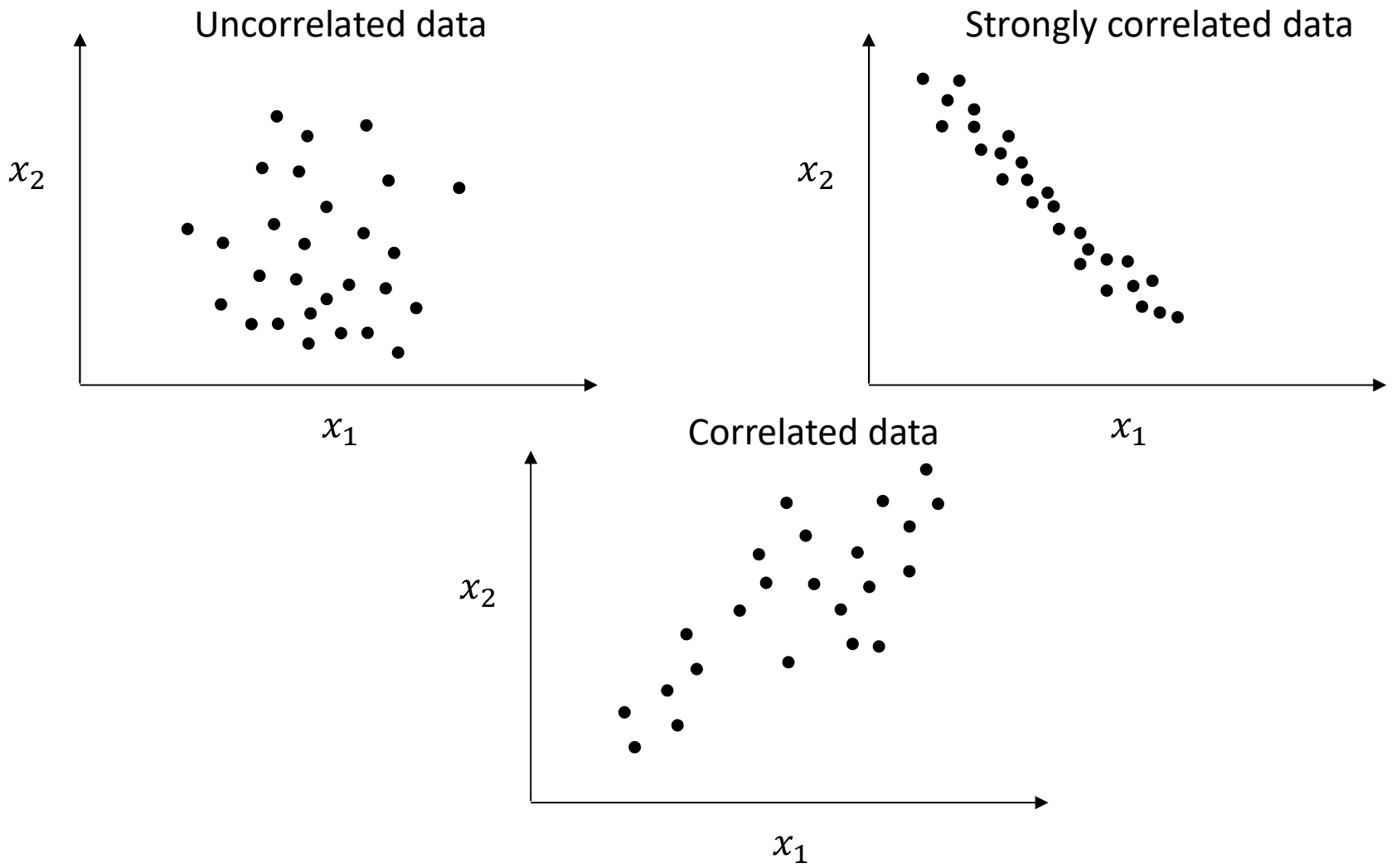




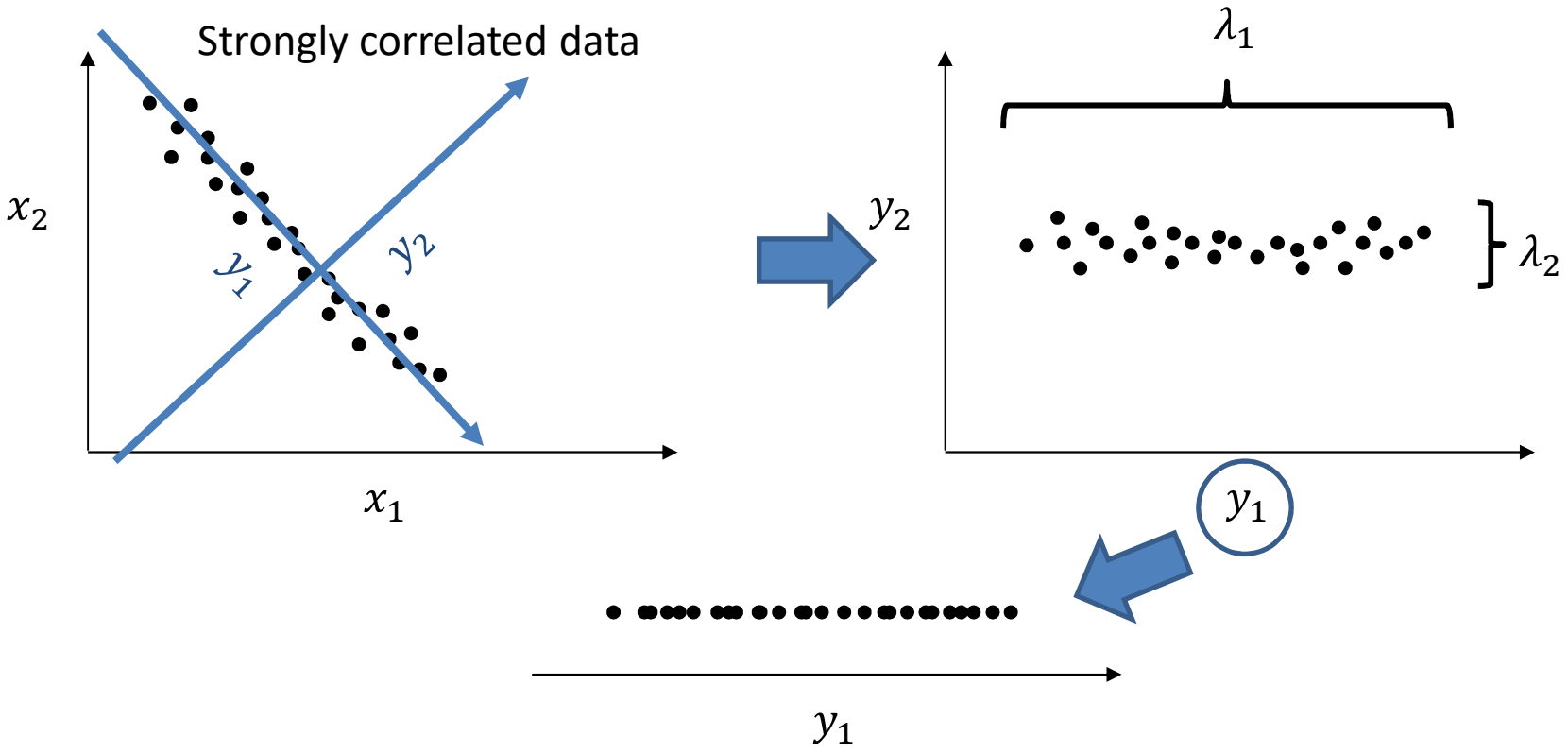




STRONG CORRELATIONS REDUCE DIMENSIONALITY



What is the task that we want to perform?



NOTE: We just need a rotation of the space that reduces (cancels) covariances

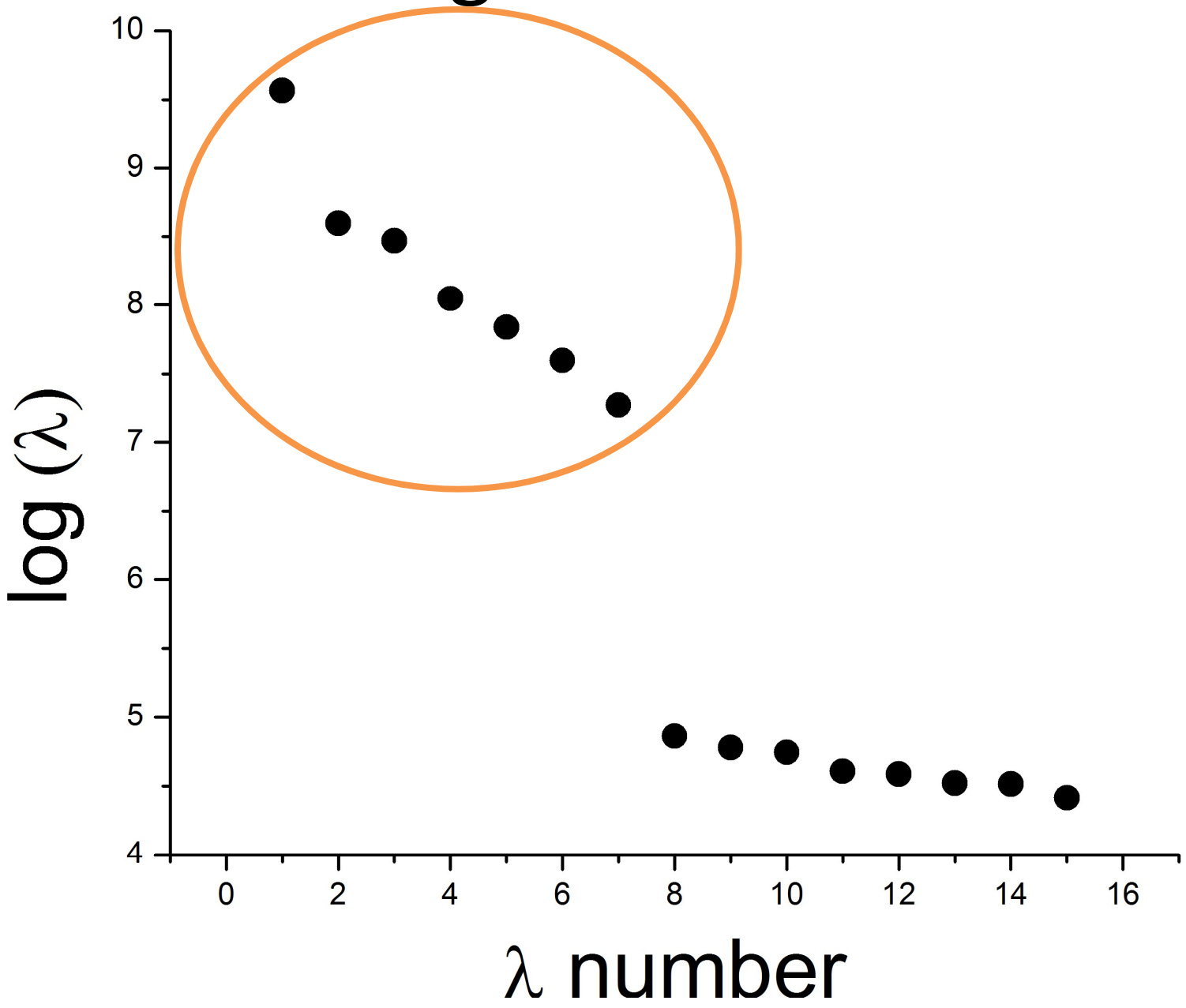
DIAGONALIZATION OF THE COVARIANCE MATRIX

PCA. How to...

Given n observations $\{x_1, x_2, \cdots, x_n\}$ of m-dimensional column vectors

1. Compute the mean vector $\mu = \frac{1}{n}\sum_1^n x_i$
2. Compute the covariance matrix by MLE $\mathbb{C} = \frac{1}{n}\sum_1^n (x_i - \mu)(x_i - \mu)^T$
3. Compute the eigenvalue/eigenvector pairs (λ_i, u_i) of $\mathbb{C}$ with $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \cdots \geq \lambda_m$ (Diagonalize/SVD) and sort.
4. Choose the number of dimensions (d) in which project the data
5. Project by ${y_i}^{(j)} = {x_i}^T u_j$

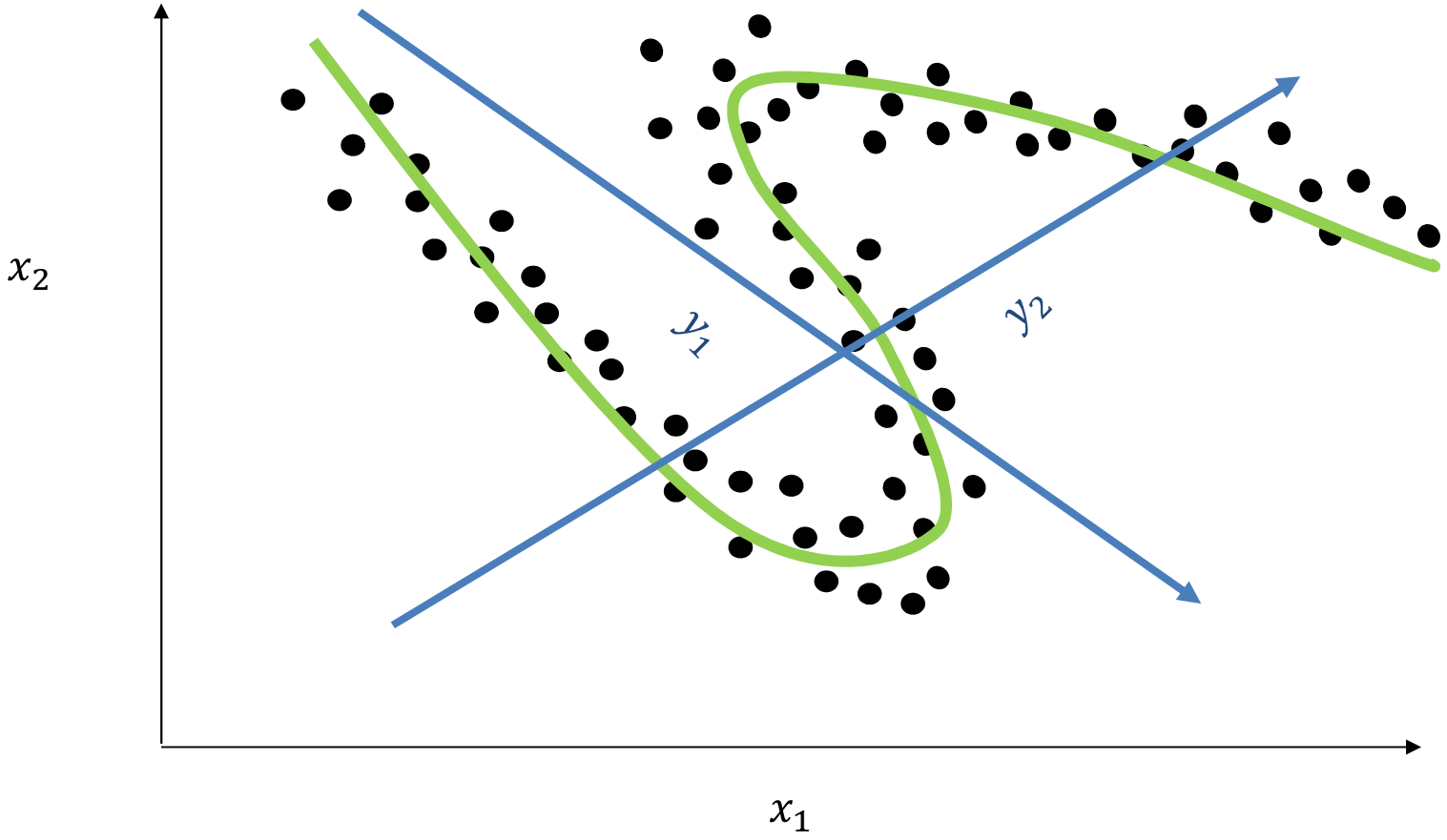
Choosing the number of PC's



PCA problems

- Poorly suited for non-linear transformations

Addressing the non-linear problem

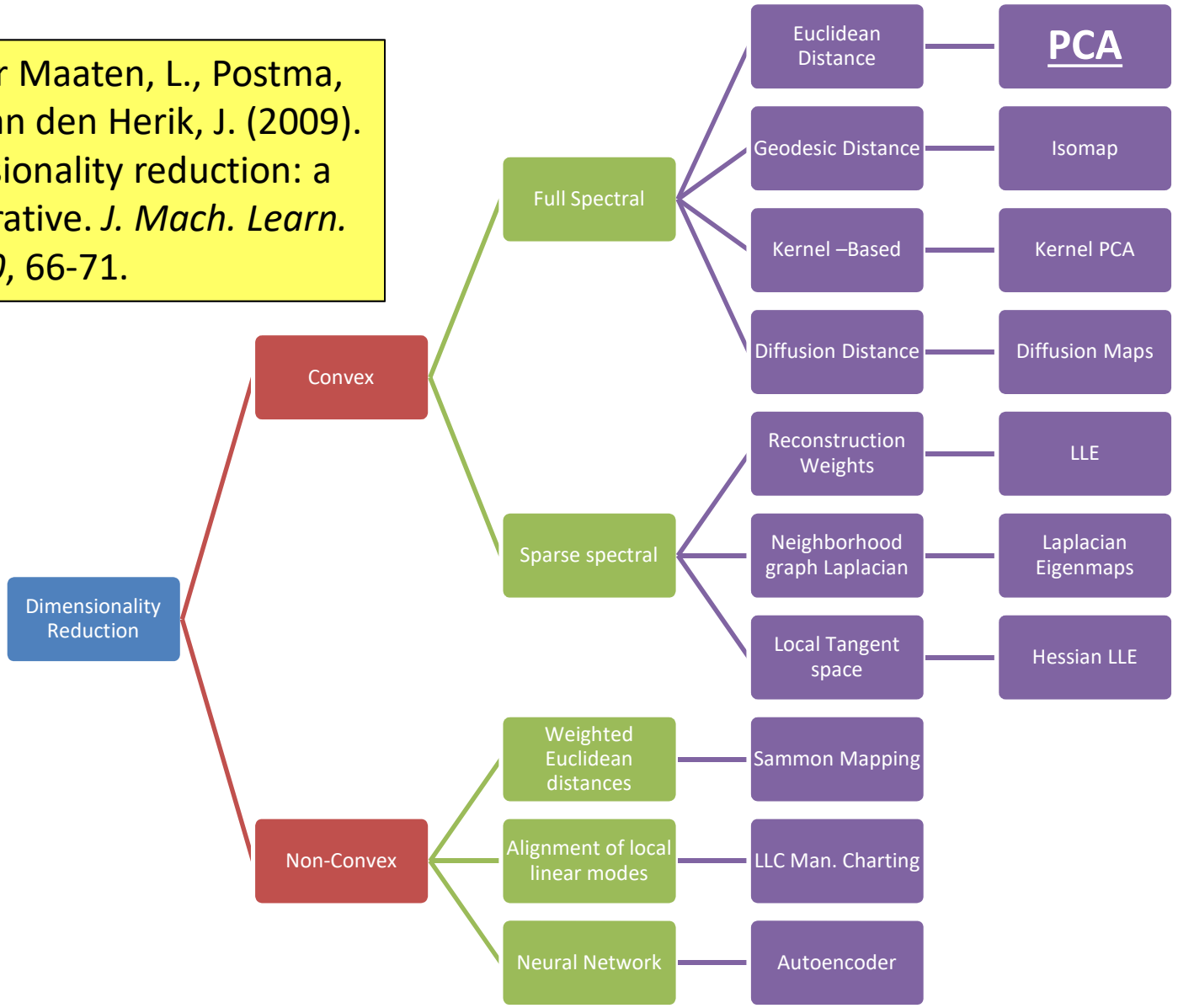


PCA problems

- Poorly suited for non-linear transformations
- Not able of capturing invariances
- By using the covariance along the whole dataset, is poor suited for problems not well described by this parameter.
- Not scale invariant
- Focused on large pairwise distances

Beyond PCA

Van Der Maaten, L., Postma, E., & Van den Herik, J. (2009). Dimensionality reduction: a comparative. *J. Mach. Learn. Res.*, *10*, 66-71.



Focusing on preserving the distances: Multidimensional Scaling

Δ^{ij} Distance in the ambient space

δ^{ij} Distance in the projected space

Stress: Measure of the discrepancy between both distances.
We try to minimize it.

Focusing on preserving the distances: Multidimensional Scaling

$$S = \sum_{\mathrm{i,j}} \left(\left(\Delta^{ij}\right)^2 - \left(\delta^{ij}\right)^2 \right)^2$$

Classical MDS: Formally equivalent to PCA when using Euclidean Distance. Analytical minimization.

$$S = \frac{\sum_{i,j} \left(\left(\Delta^{ij}\right) - \left(\delta^{ij}\right) \right)^2}{\sum_{i,j} \delta^{ij}}$$

Metric MDS: Preserves the relative importance of the distances. Numerical minimization.

$$S = \frac{\sum_{i,j} \left(g\left(\Delta^{ij}\right) - f\left(\delta^{ij}\right) \right)^2}{\sum_{i,j} f(\delta^{ij})^2}$$

Non-metric MDS: You choose what to preserve. Numerical minimization.

Classical MDS

1.- Obtain the Grahm matrix

$$\hat{G}^{ij} = -\frac{1}{2}\left(\Delta^{ij} - \frac{1}{N^2}\sum_{i,j} \Delta^{ij} - \frac{1}{N}\left(\sum_{i} \Delta^{ij} + \sum_{j} \Delta^{ij} \right) \right)$$

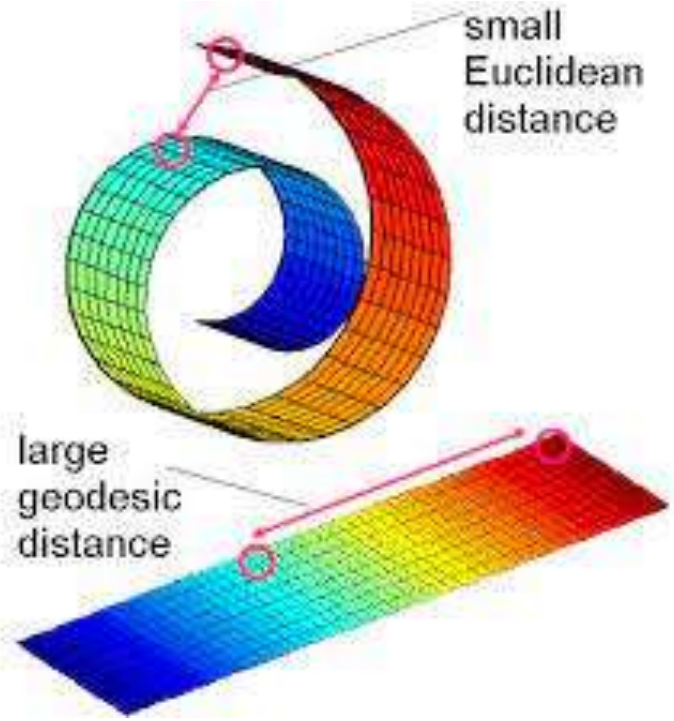
This is called double centering.

2.- Diagonalize it

3.- From the spectrum, decide *d*

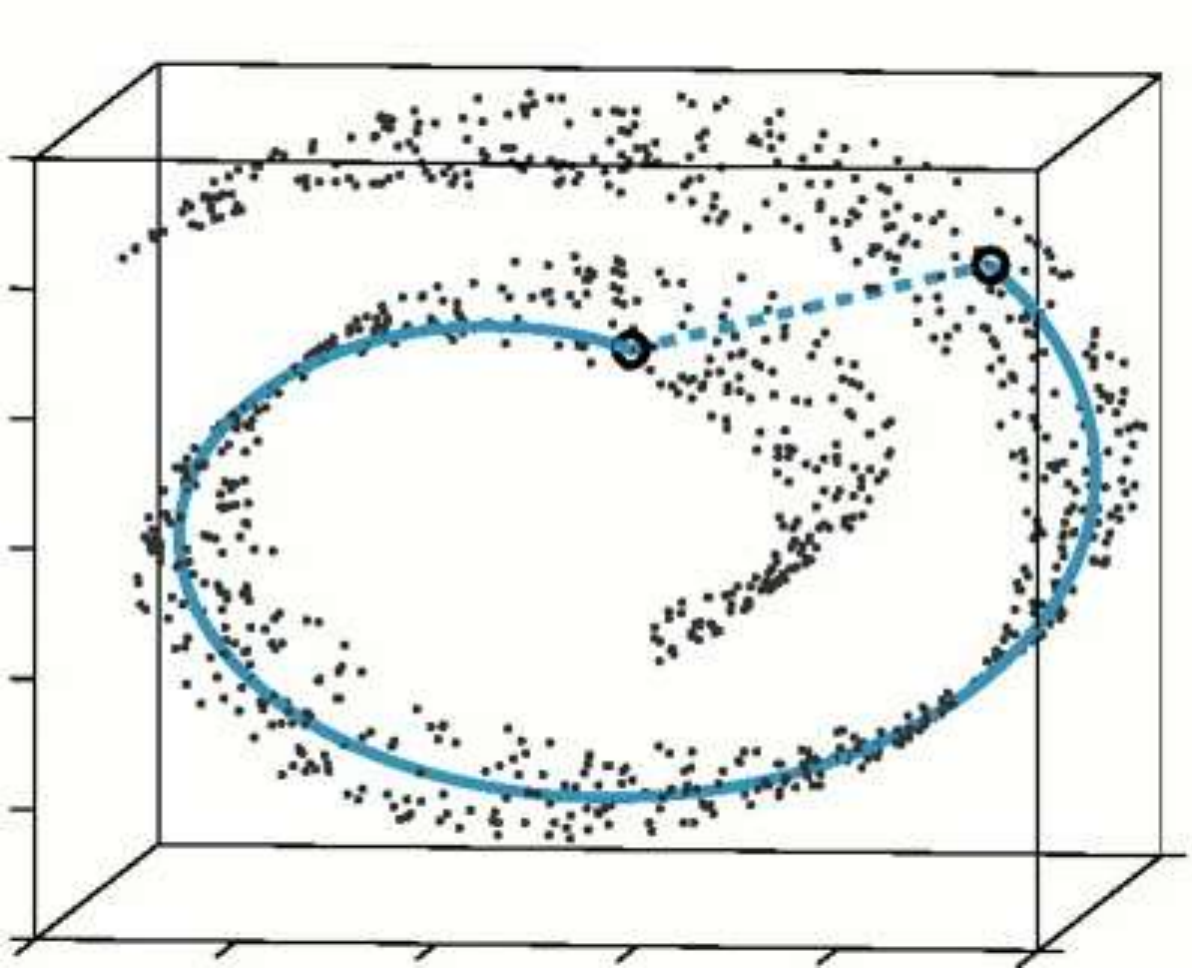
Note that this procedure does not requires Δ^{ij} being computed as the Euclidean distance!!!

ISOMAP



Uses classical MDS setting Δ^{ij} equal to the geodesic distance instead of the Euclidean distance.

Geodesic distance



Geodesic distance

- Determine the neighbors.
  - All points in a fixed radius.
  - K nearest neighbors
- Construct a neighborhood graph.
  - Each point is connected to the other if it is a K nearest neighbor.
  - Edge Length equals the Euclidean distance
- Compute the shortest paths between two nodes

Geodesic distance: Naïve algorithm

1. Assign to every node a tentative distance value: set it to zero for our initial node and to infinity for all other nodes.
2. Set the initial node as current. Mark all other nodes unvisited. Create a set of all the unvisited nodes called the unvisited set.
3. For the current node, consider all of its unvisited neighbors and calculate their tentative distances. Compare the newly calculated tentative distance to the current assigned value and assign the smaller one. For example, if the current node A is marked with a distance of 6, and the edge connecting it with a neighbor B has length 2, then the distance to B (through A) will be 6 + 2 = 8. If B was previously marked with a distance greater than 8 then change it to 8. Otherwise, keep the current value.
4. When we are done considering all of the neighbors of the current node, mark the current node as visited and remove it from the unvisited set. A visited node will never be checked again.
5. If the destination node has been marked visited (when planning a route between two specific nodes) or if the smallest tentative distance among the nodes in the unvisited set is infinity (when planning a complete traversal; occurs when there is no connection between the initial node and remaining unvisited nodes), then stop. The algorithm has finished.
6. Otherwise, select the unvisited node that is marked with the smallest tentative distance, set it as the new "current node", and go back to step 3.

Kernel PCA

- Apply a kernel function (non-linear transformation) to the distances in such a way that we only weight those that we are interested in.
- For instance: You are only interested in distances within a certain radius σ. You can use: $\widetilde{\Delta_{ij}} = 1 - e^{-\frac{\Delta_{ij}}{\sigma}}$

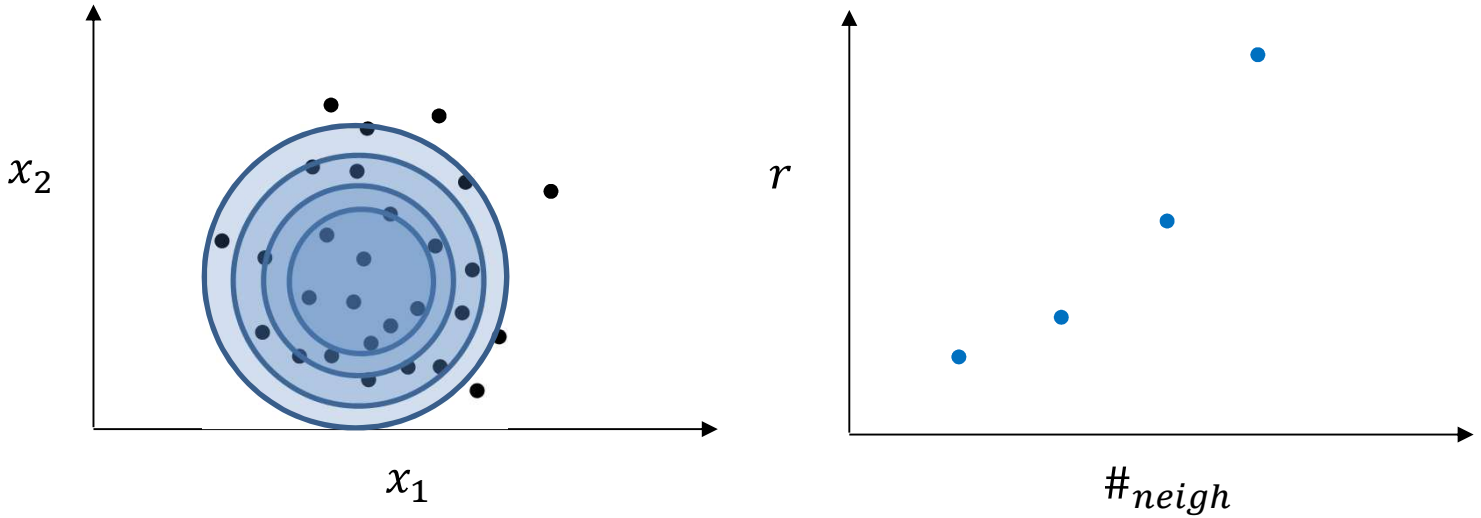
t-SNE or UMAP

- These methods aim to reproduce the probabilities of neighborhood between data points.
- Optimize a loss function that measures the overlap between these probabilities in the ambient and projection spaces.
- Generally excellent performance for visualization, take care when using them for pre-processing.
- Random initialization, not always reproducible results.

Directly estimation of the Intrinsic dimension

Minimum number of parameters required to describe the data while minimizing the information loss.

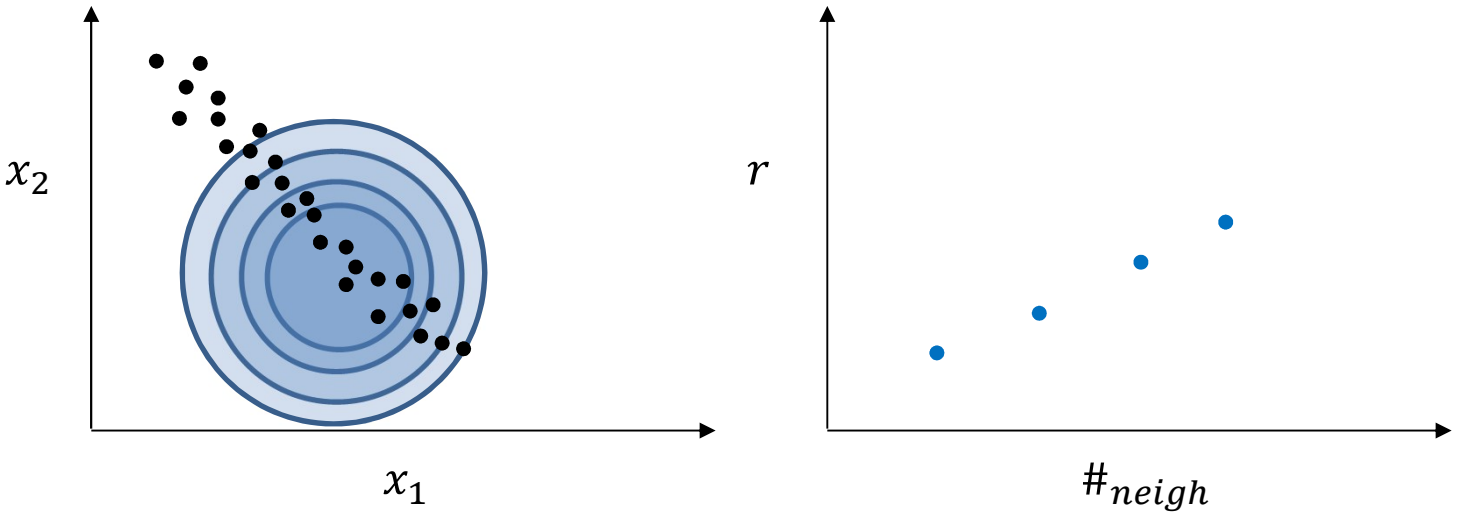
Classical approach $\#_{neigh} = \rho\, r^d$



Directly estimation of the Intrinsic dimension

Minimum number of parameters required to describe the data while minimizing the information loss.

Classical approach $\#_{neigh} = \rho\, r^d$



Directly estimation of the Intrinsic dimension

In practice:

$\#_{neigh} = \rho\, r^d \longrightarrow \log(\#_{neigh}) = \log\rho + d\ \log r$ Linear fit

Only true at constant density:
Modern methods try to
disentangle both measures

TWO-NN

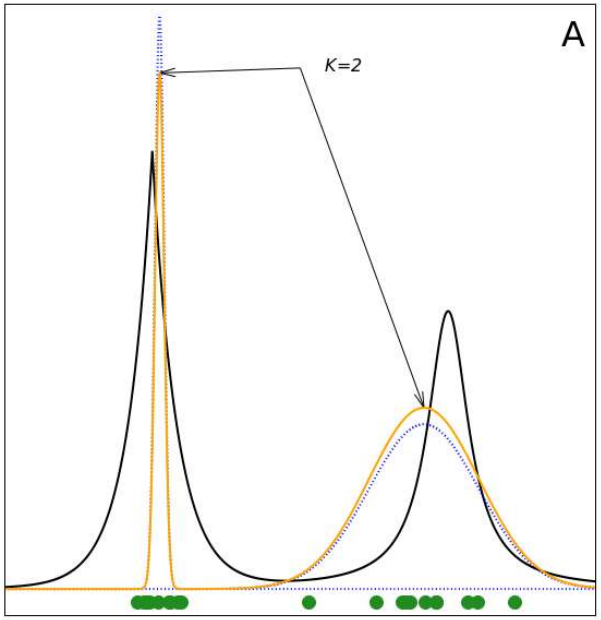
$$\mu_i = \frac{r_{i,2}}{r_{i,1}};$$

$$P(\mu|\rho) = P(\mu) = \frac{d}{\mu^{1+d}}$$

DENSITY ESTIMATION

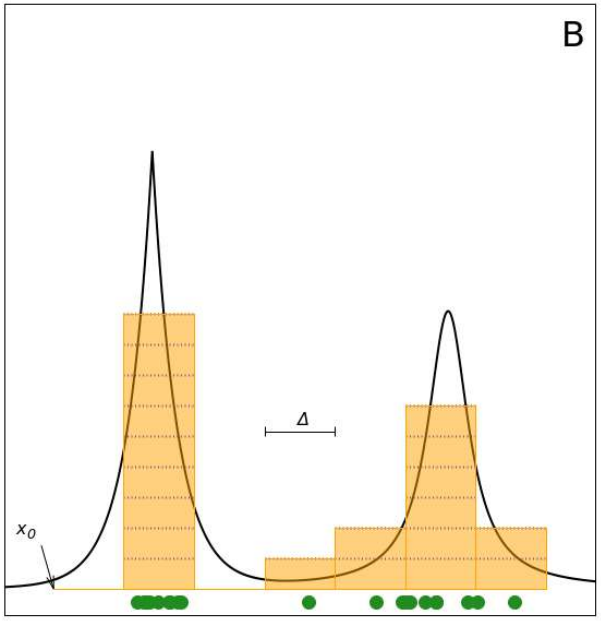
Gaussian mixture models

- Parametric approach (We enforce a functional form to $p(x)$)
- $p(x) = \sum_{i=1}^{K} w_i e^{-\left(\frac{x-\mu_i}{s_i}\right)^2}$
- The only parameter is K



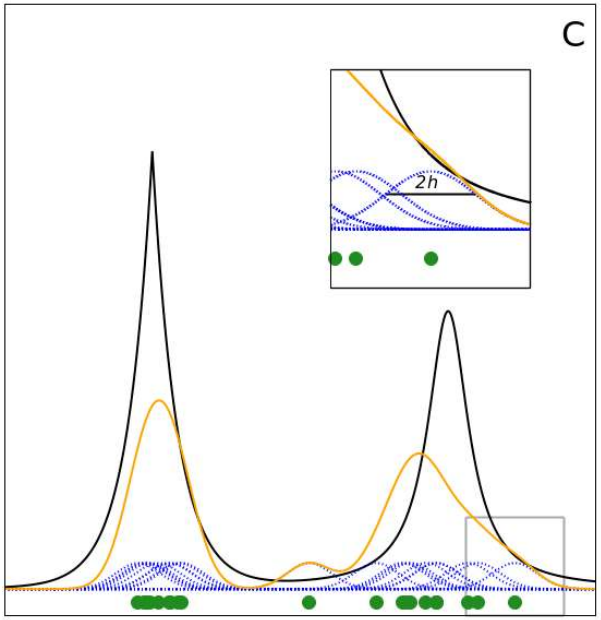
Histograms

- Non-parametric approach (We don't enforce a functional form to $p(x)$)
- Count the number of points within a binwidth Δ
- Two parameters: Δ & x_0
- Poor performance at d>2-3



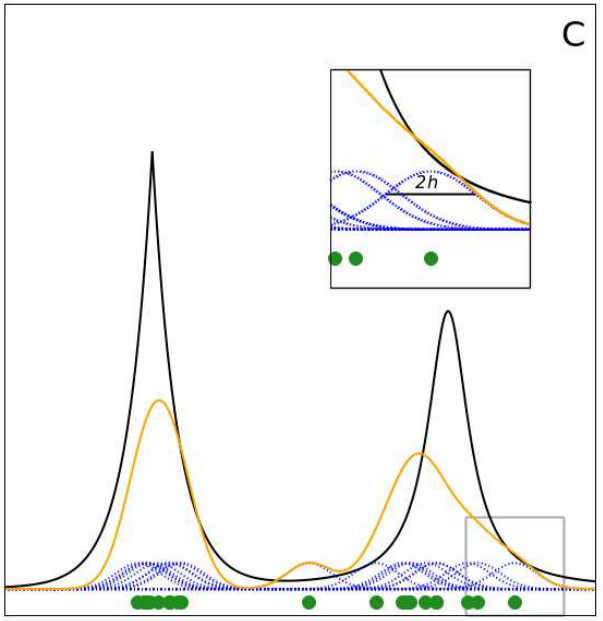
Kernel density estimation

- Non-parametric approach (We don't enforce a functional form to $p(x)$)
- $p(x) = \sum_i \frac{1}{h\sqrt{2\pi}} e^{-\left(\frac{x-x_i}{h}\right)^2}$
- Puts a gaussian in the top of each point, then sum.
- Single parameter: h



K-Nearest neighbor density estimation

- Non-parametric approach (We don't enforce a functional form to $p(x)$)
- $p(x) = \frac{k}{{r_k}^d}$
- The density at a point is the inverse of k times the volume occupied by its k nearest neighbors.
- Single parameter: k



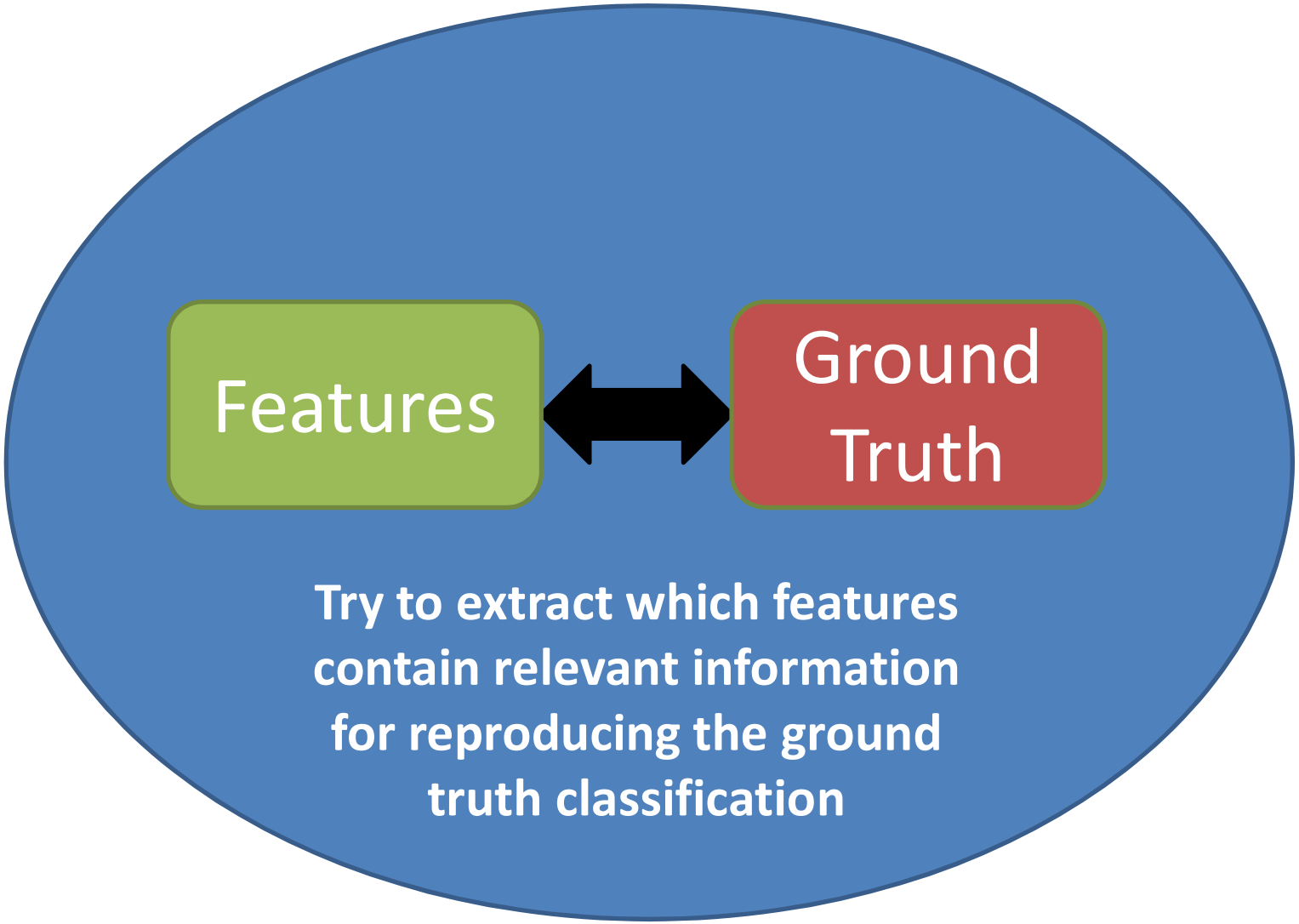
Feature Selection

Master in High Performance Computing

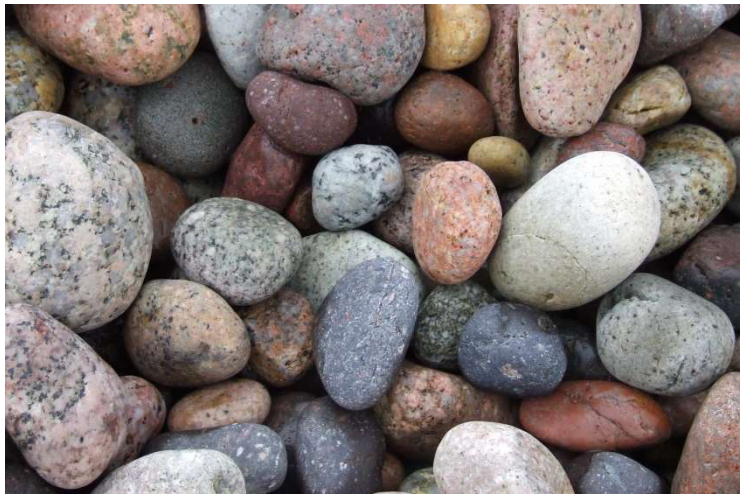
Feature Selection & Dimensionality reduction

- Highly related with the problem that we want to solve
- It may need feedback from the whole process
- Feature selection usually depends on labeled data while dimensionality reduction does not.
- Feature selection can be based on expertise…

Feature selection



My stone collection (3)



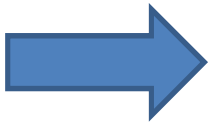
- Weight
- Light wavelength
- Shape
- Volume
- Rugosity
- …

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 and sort by the score

Var ID	Score
A	0.9
B	0.1
C	0.2
D	0.7
E	0.03
F	0.85



Var ID	Score
A	0.9
F	0.85
D	0.7
C	0.2
B	0.1
E	0.03

Techniques for Feature Selection (1) Variable Ranking

- Which variables explain better the labels?
- Quantify it and sort by the score
  - 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).

Techniques for Feature Selection (1) Variable Ranking

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

Techniques for Feature Selection (1) Single variable classifier

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**

- 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
- We will go in deeper detail when talking about external validation

Techniques for Feature Selection (1) Variable Ranking

- Which variables explain better the labels?
- Quantify it and sort by the score
  - 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

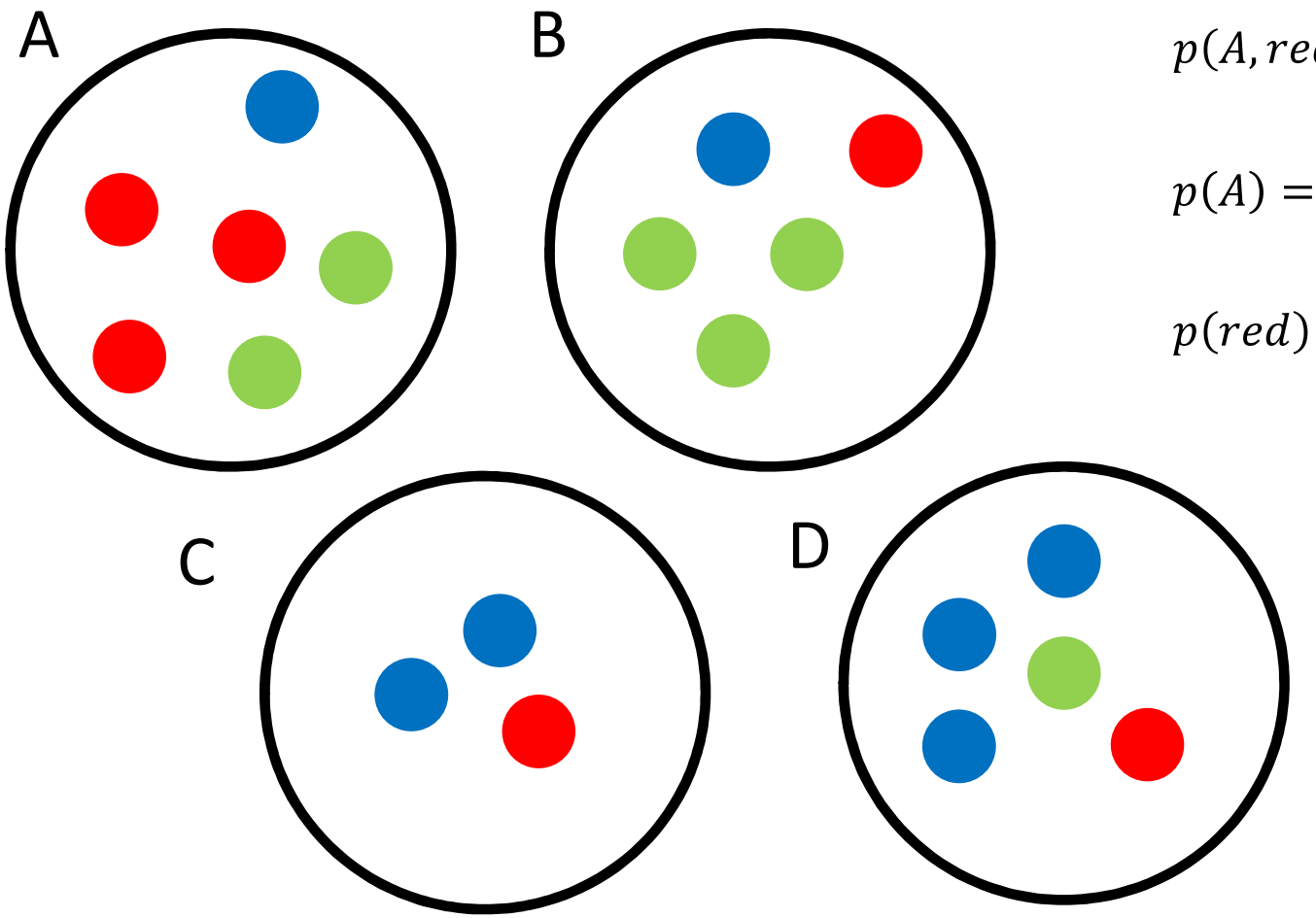
$$I(i) = \int_{x_i} \int_{y} p(x_i, y) \log \frac{p(x_i, y)}{p(x_i)p(y)}$$

- 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.

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



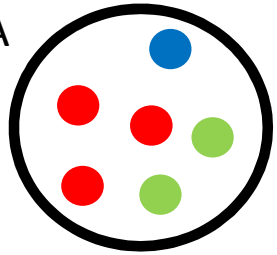
$$p(A, red) = \frac{3}{19}$$

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

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

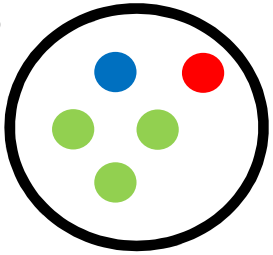
Techniques for Feature Selection (1)
Information Theoretic Ranking

A



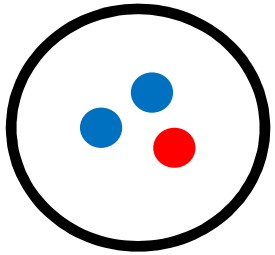
$$I = \frac{3}{19}\log\left(\frac{{}^{3}/_{19}}{{}^{6}/_{19}{}^{6}/_{19}}\right) + \frac{1}{19}\log\left(\frac{{}^{1}/_{19}}{{}^{6}/_{19}{}^{7}/_{19}}\right)$$

B



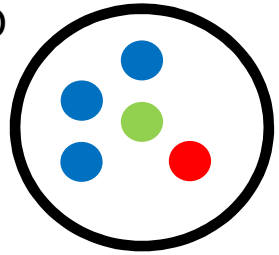
$$+ \frac{2}{19}\log\left(\frac{{}^{2}/_{19}}{{}^{6}/_{19}{}^{6}/_{19}}\right) + \frac{1}{19}\log\left(\frac{{}^{1}/_{19}}{{}^{5}/_{19}{}^{6}/_{19}}\right) +$$

C



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

D



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

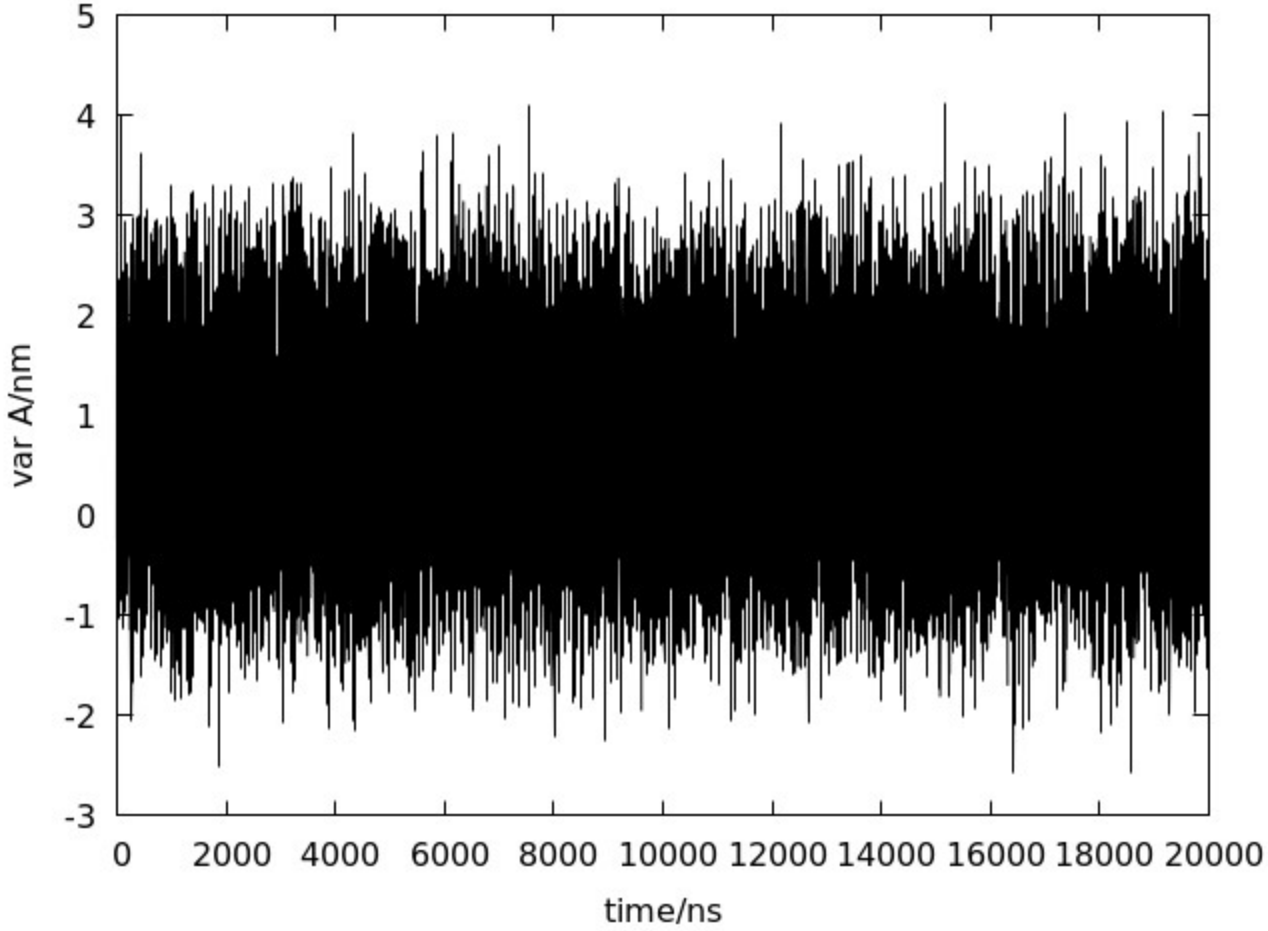
$$\frac{2}{19}\log\left(\frac{{}^{2}/_{19}}{{}^{3}/_{19}{}^{6}/_{19}}\right) + \frac{1}{19}\log\left(\frac{{}^{1}/_{19}}{{}^{5}/_{19}{}^{6}/_{19}}\right) +$$

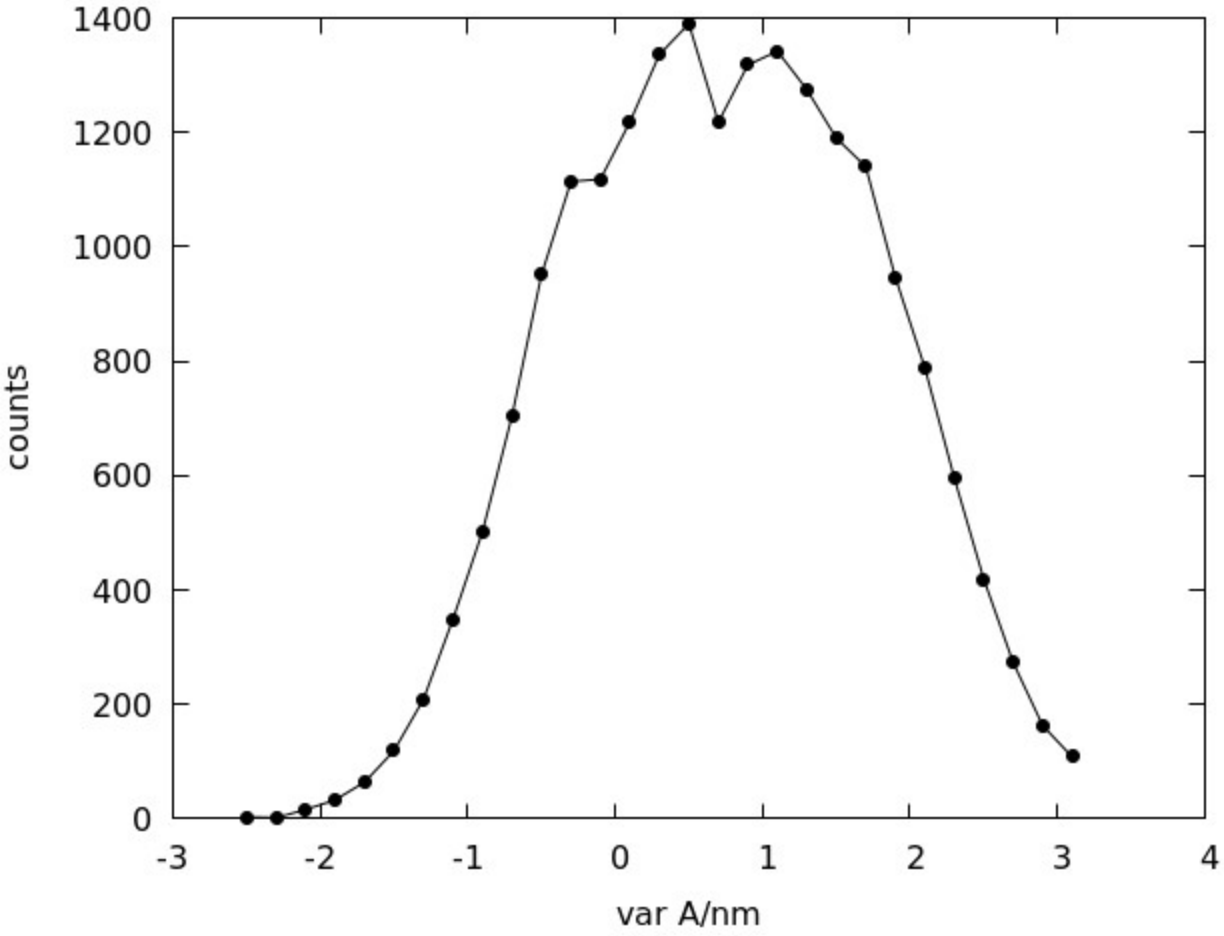
$$\frac{1}{19}\log\left(\frac{{}^{1}/_{19}}{{}^{5}/_{19}{}^{7}/_{19}}\right) + \frac{3}{19}\log\left(\frac{{}^{3}/_{19}}{{}^{5}/_{19}{}^{6}/_{19}}\right) \approx 0.17$$

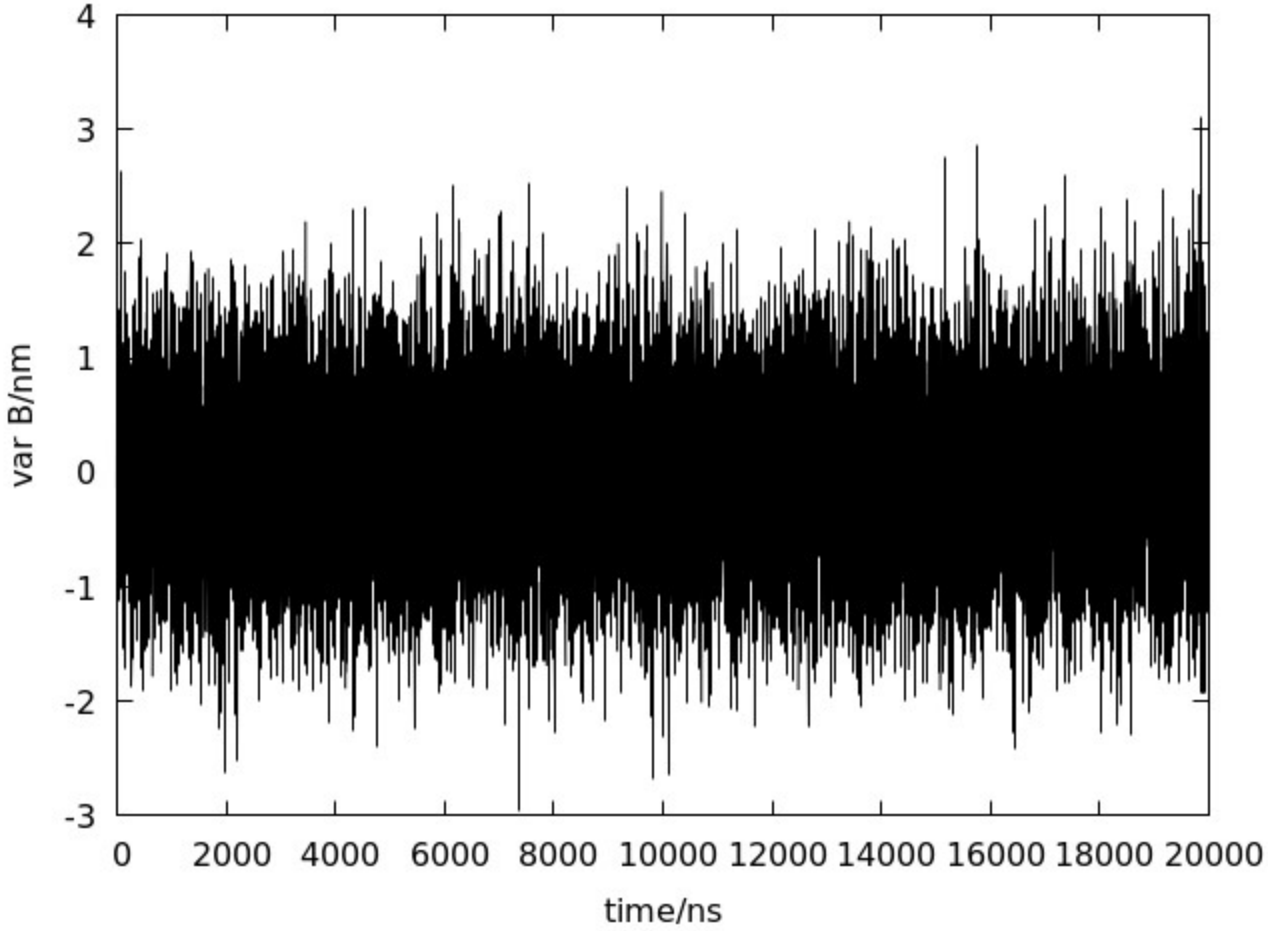
Techniques for Feature Selection (1)

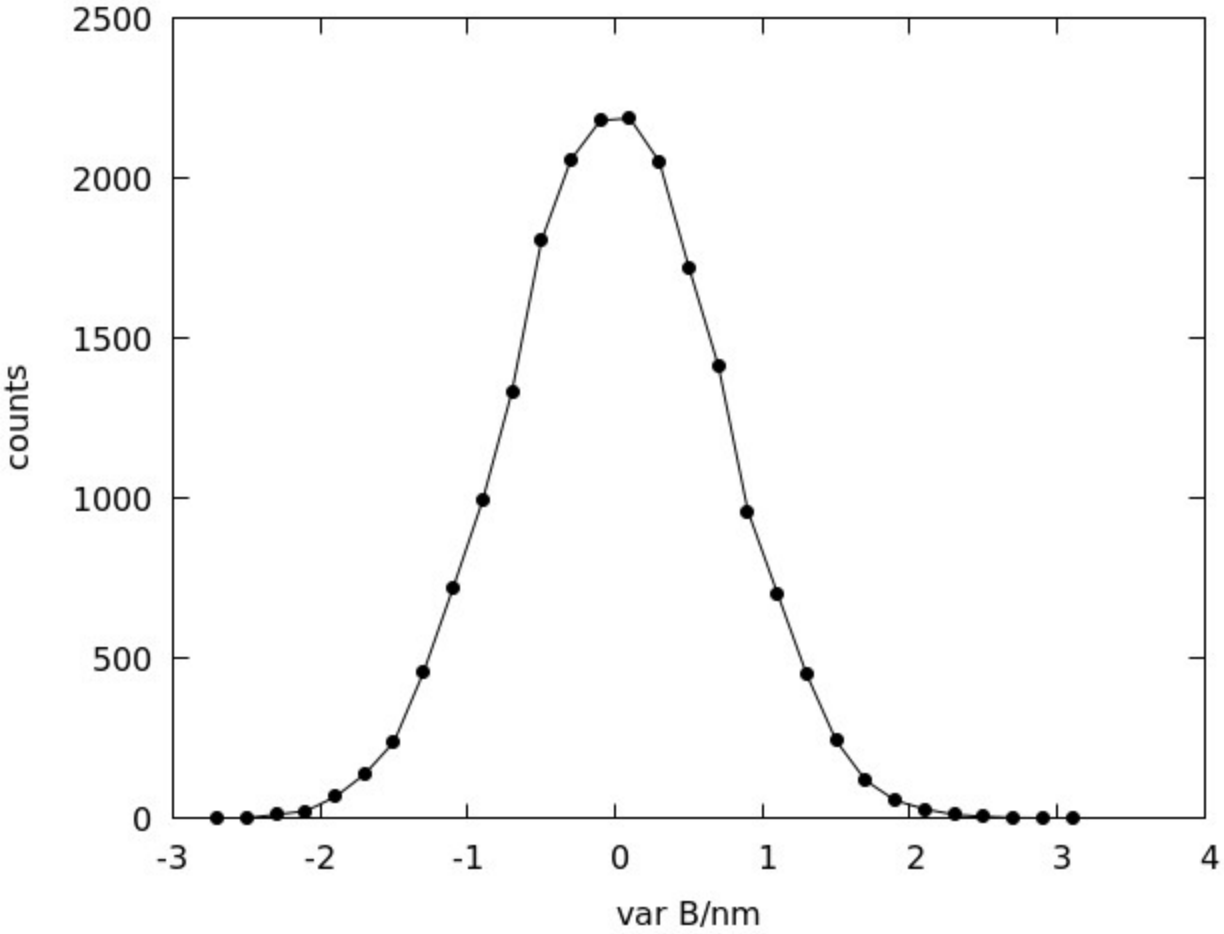
Some questions about Variable Ranking

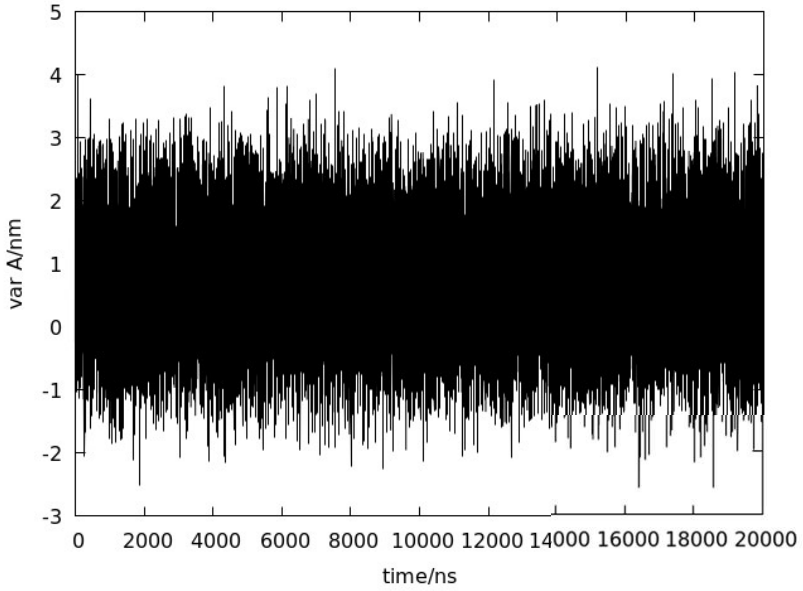
- How to treat redundant variables?
  - Redundant variables get the same information but its combination can lead to noise reduction.
  - Correlation is a measure of redundancy
- A variable useless by itself can be useful together with others

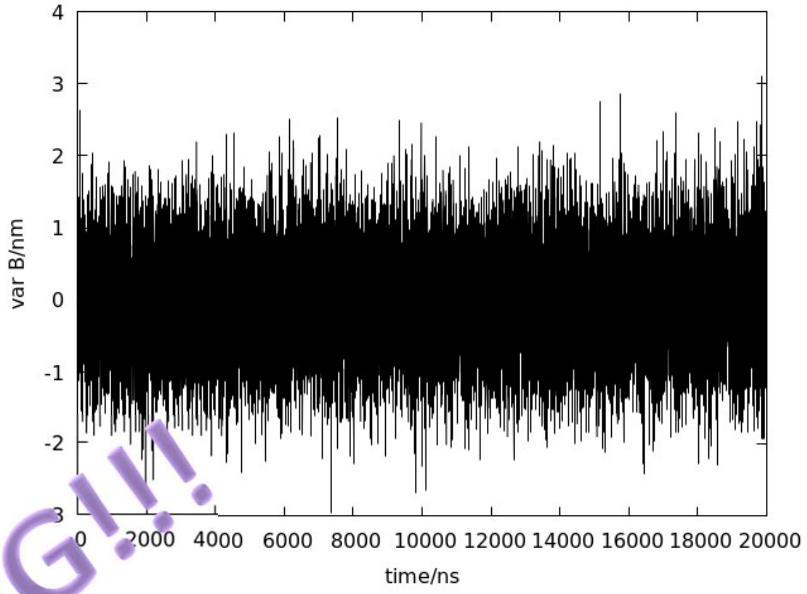

5
4
3
2
1
0
-1
-2
-3
var A/nm
0
2000
4000
6000
8000
10000
12000
14000
16000
18000
20000
time/ns


1400
1200
1000
800
600
400
200
0
counts
-3
-2
-1
0
1
2
3
4
var A/nm

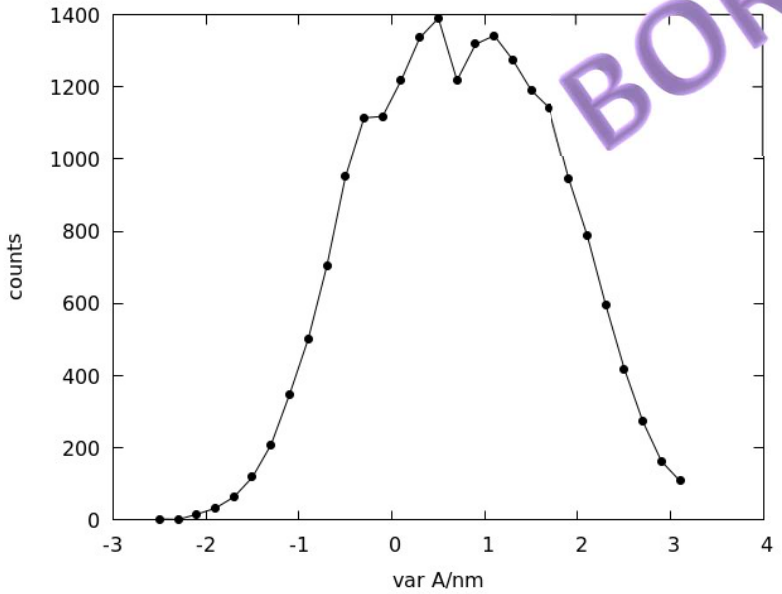

4
3
2
1
0
-1
-2
-3
var B/nm
0
2000
4000
6000
8000
10000
12000
14000
16000
18000
20000
time/ns

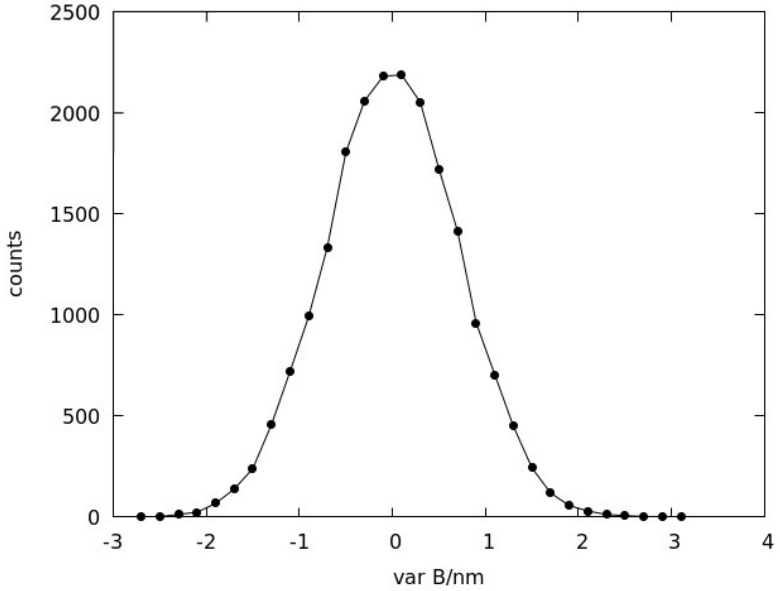

2500
2000
1500
1000
500
0
counts
-3
-2
-1
0
1
2
3
4
var B/nm



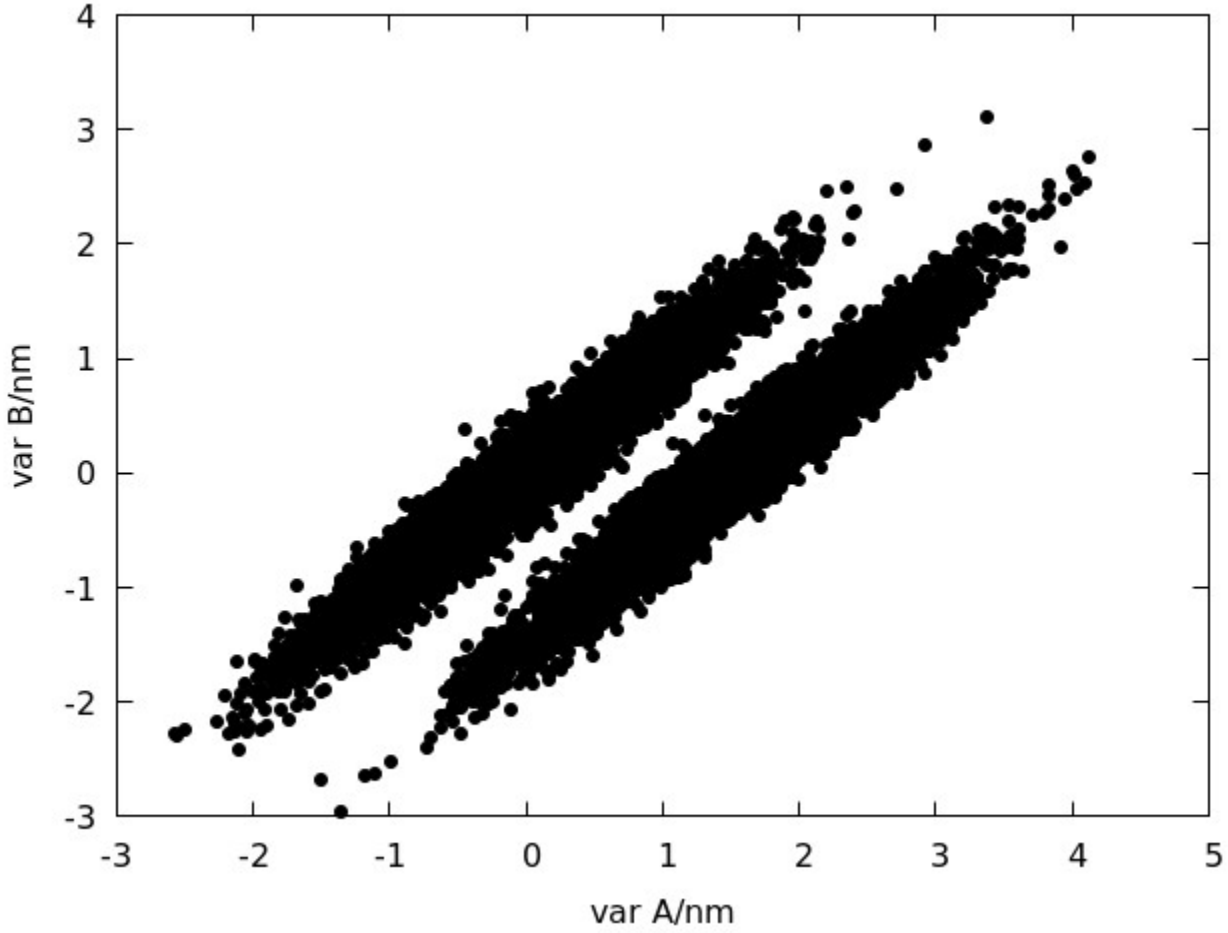


BORING!!!

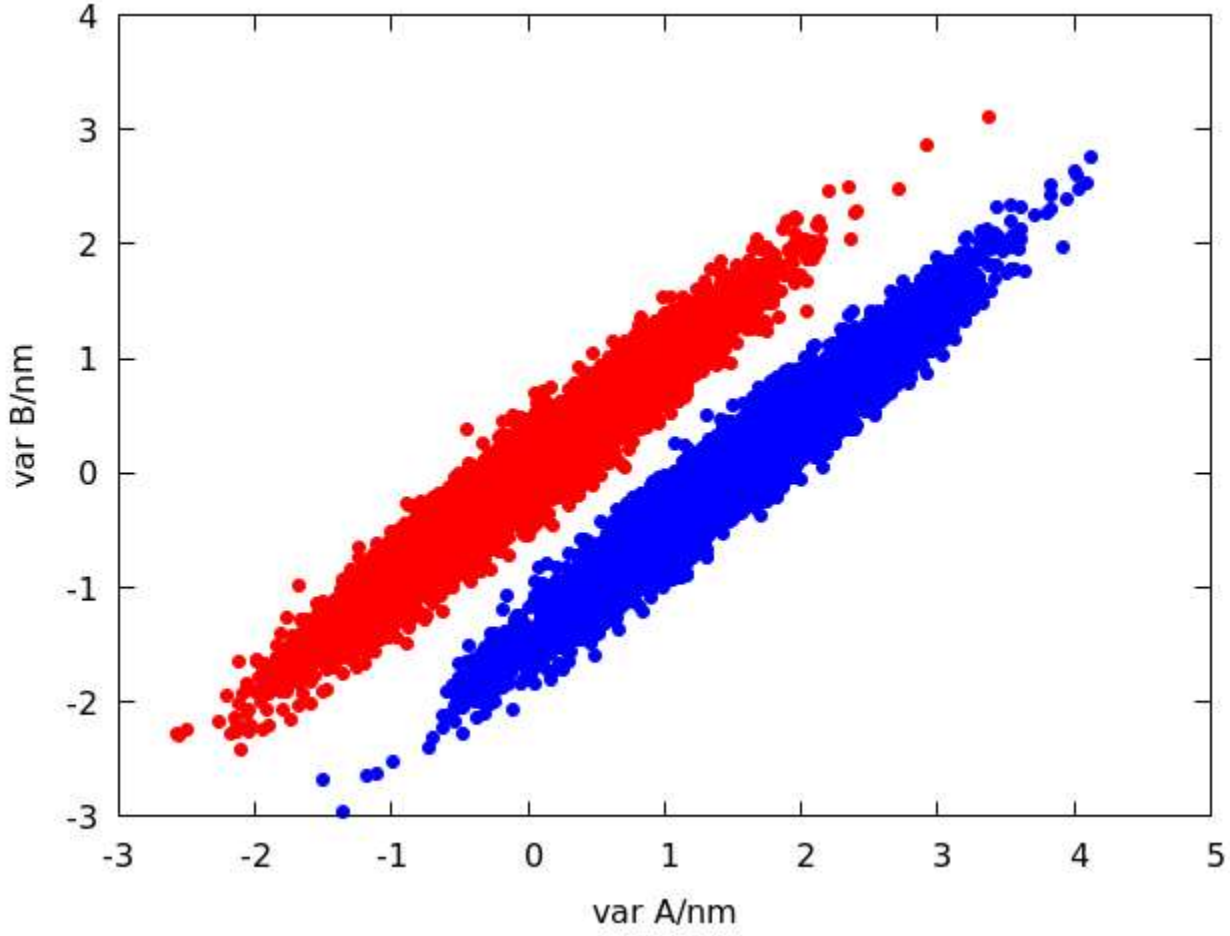




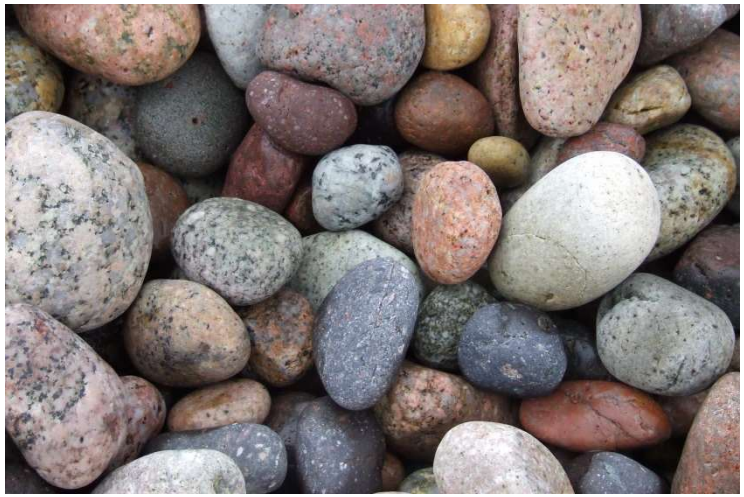
Go Multivariate! Plot 2D



Some structures appear only in multivariate analysis



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
  - How to search the space? (Brute force is NP hard.)
  - How to assess the performance of the prediction.
  - Which learning machine use.

Similarities and distances

Master in High Performance Computing

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:

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

Quantitative Features: *Metric Distances*

- Minkowski distance:

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

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

- Mahalanobis distance

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

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

Quantitative Features: *Not metric distances*

- Pearson correlation:

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

- Point Symmetry distance:

$$d_{ij} = \min_{k \neq i} \frac{\|(x_i - x_j) + (x_k - x_j)\|}{\|(x_i - x_j)\| + \|(x_k - x_j)\|}$$

- Cosine similarity:

$$S_{ij} = \frac{{x_i}^T \cdot x_j}{\|x_j\| \|x_i\|}$$

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

Master in
High Performance Computing

Theory and applications of Clustering algorithms.

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

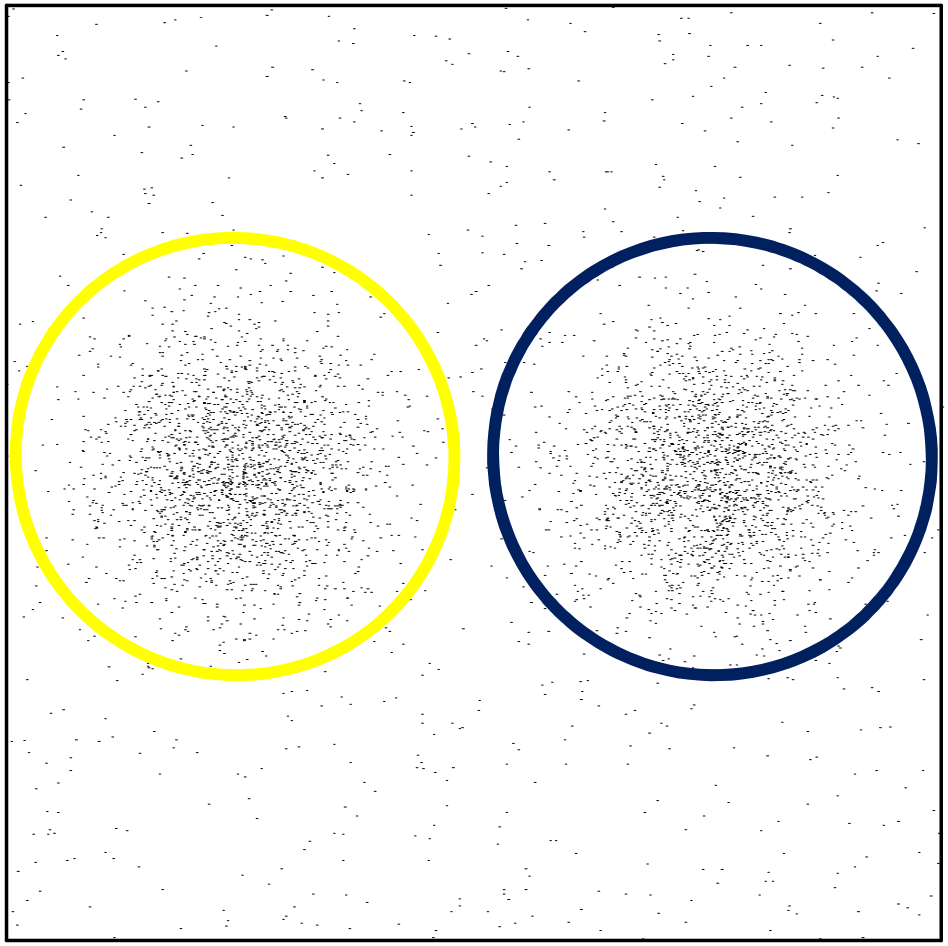
Theory and applications of Clustering algorithms.

- Motivation for Clustering
- Flat, fuzzy and Hierarchical clustering methods
- Clustering methods examples
- External and Internal Validation

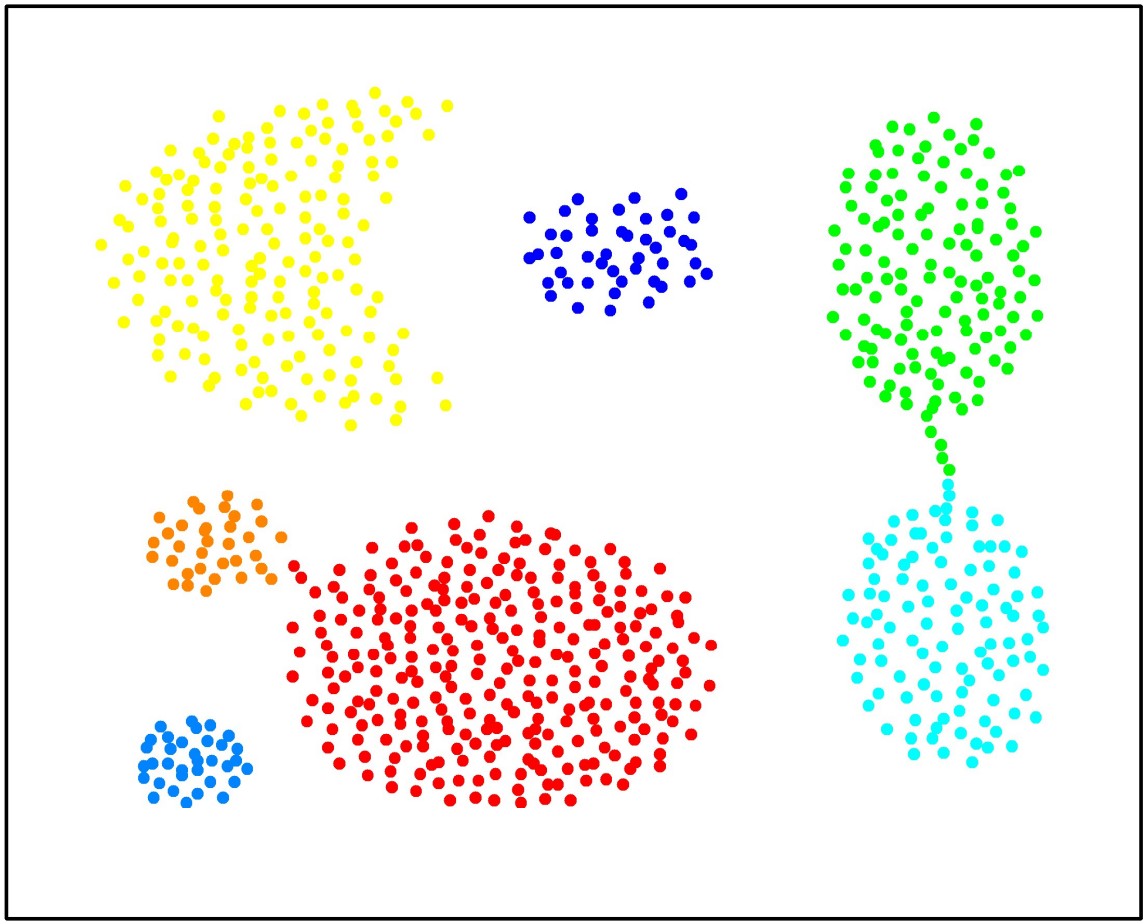
Motivation for clustering

Master in High Performance Computing

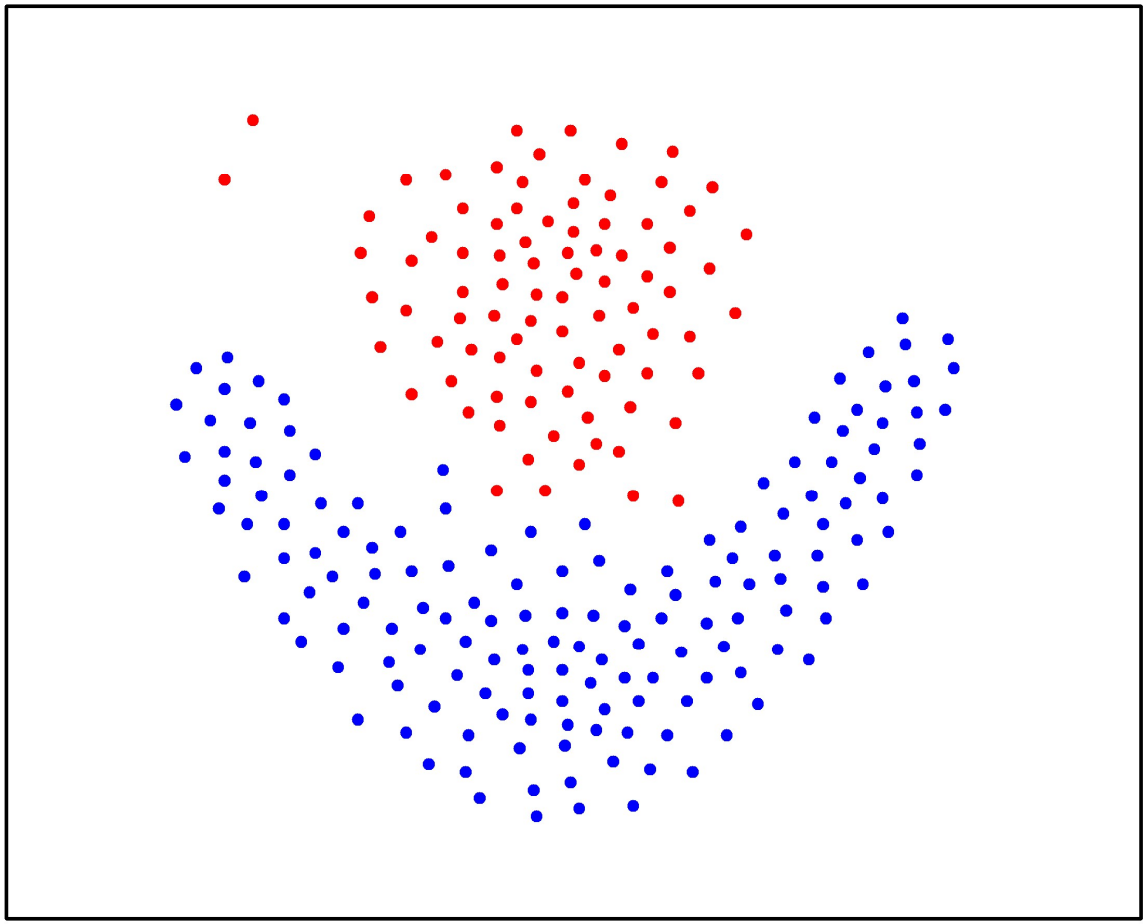
What is a cluster???



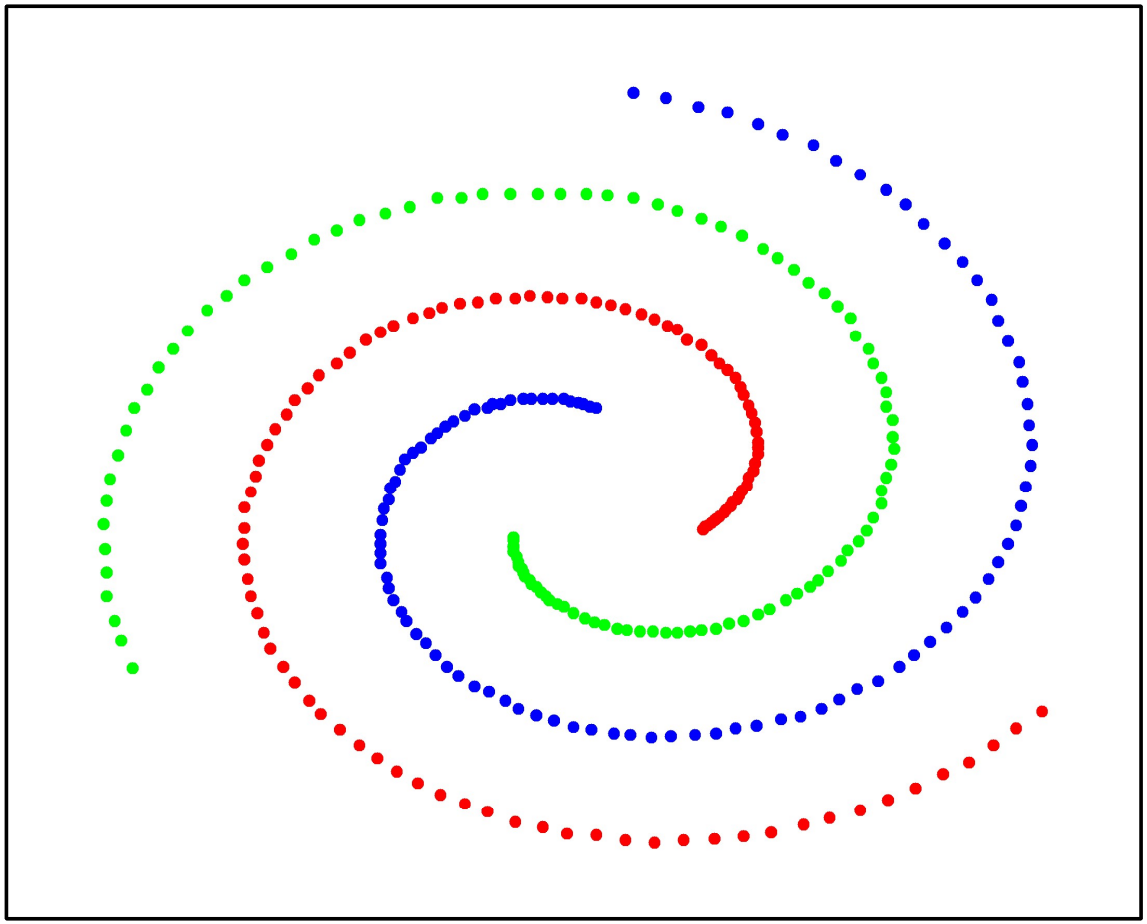
Other cluster examples…



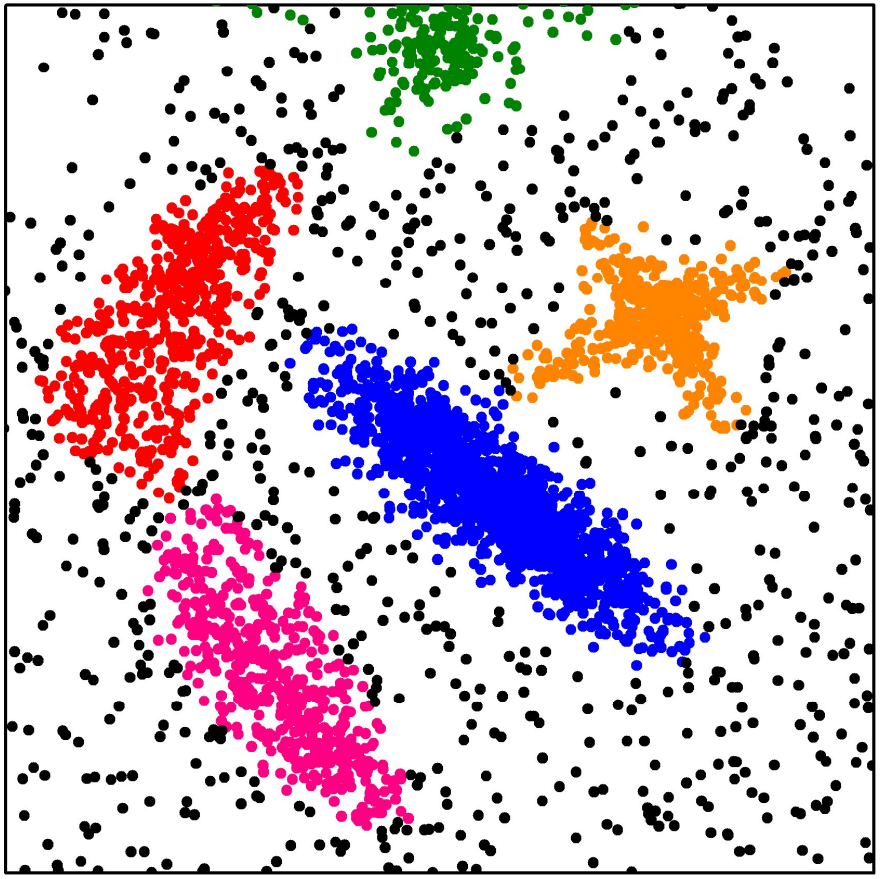
Other cluster examples…



Other cluster examples…

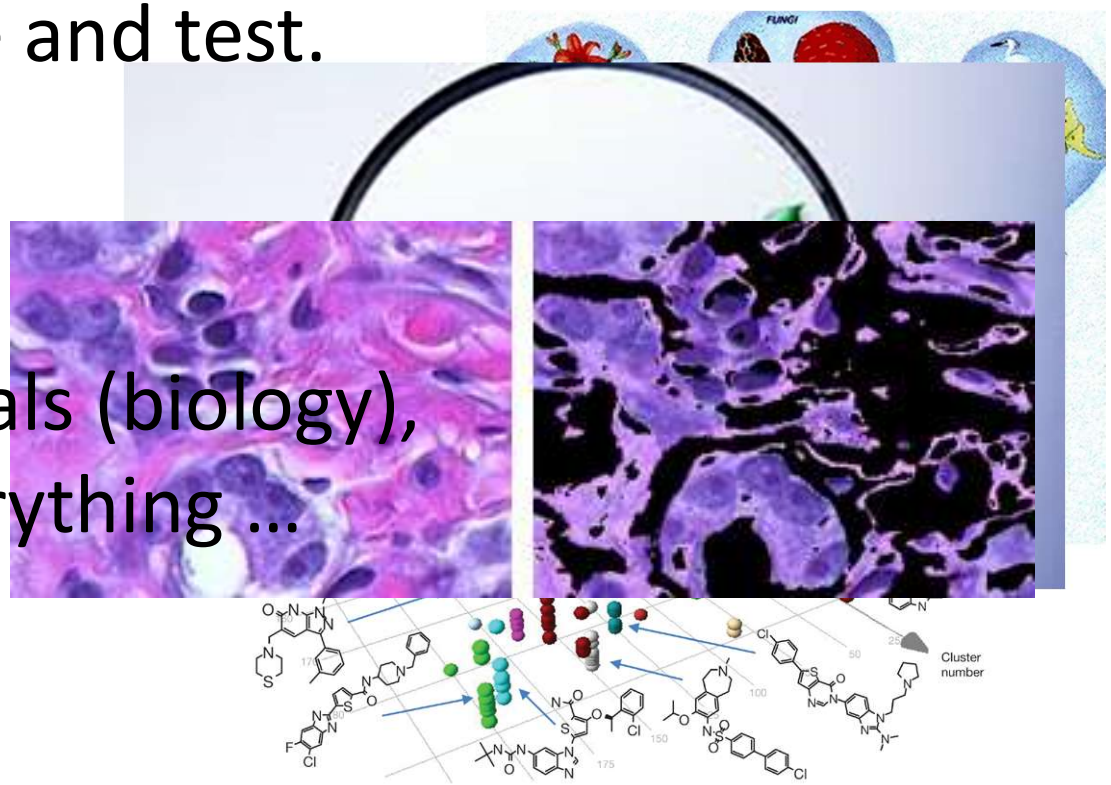


Other cluster examples…



Data mining technique that can be used for:

- 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 …



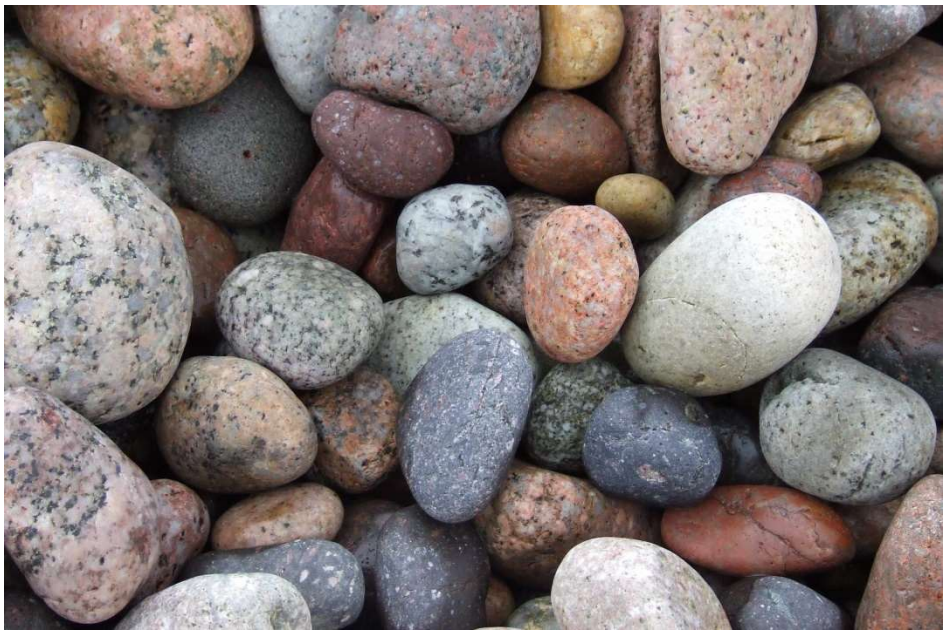
Why Clustering?

"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>

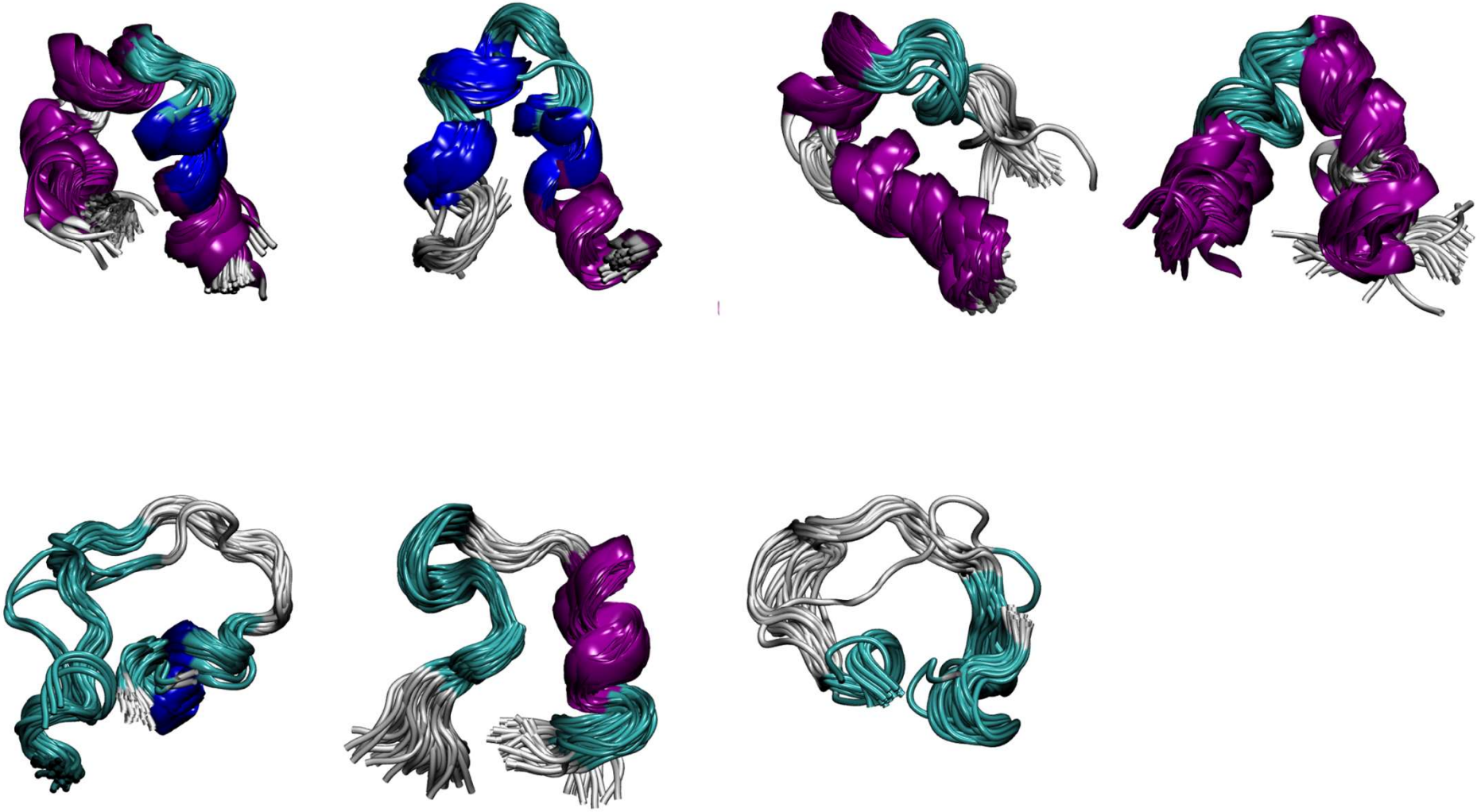
My stone collection

Cluster by light wavelength

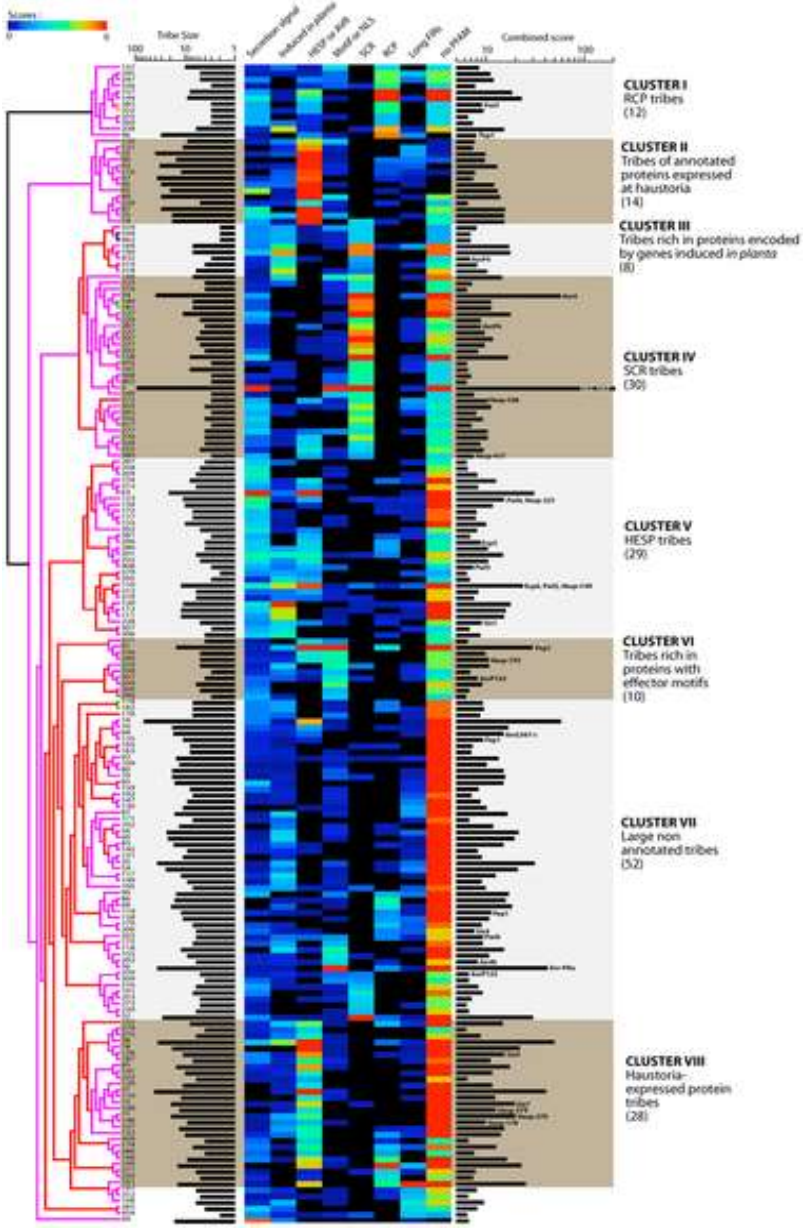


Are they nice or ugly?
Color classification?

Some clustering examples (1)



Some clustering examples (2)



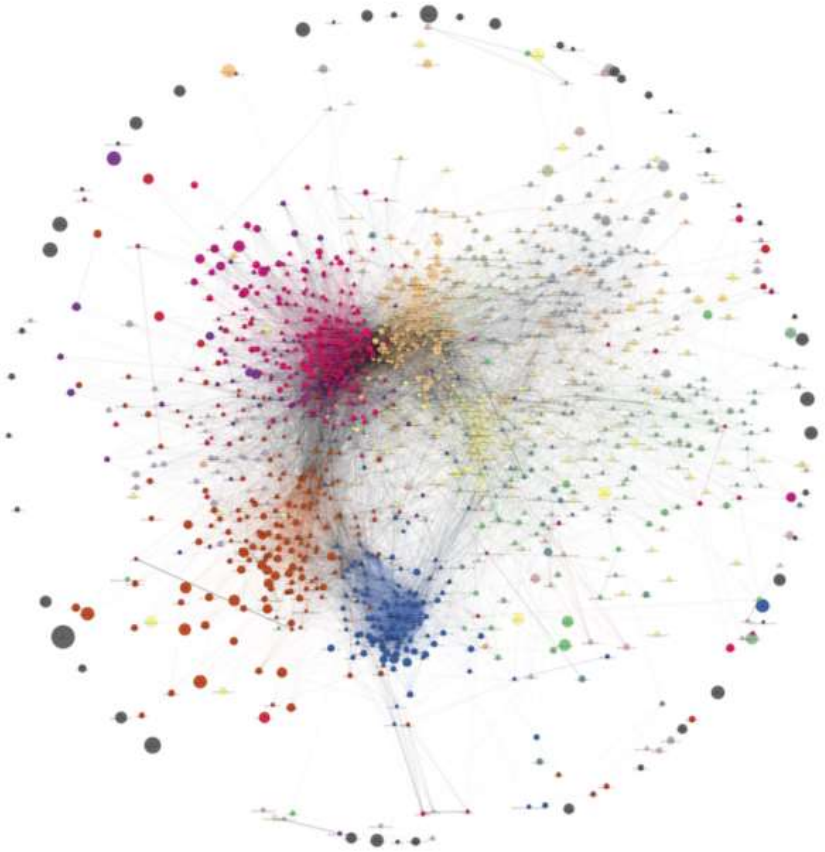
Hierarchical clustering of the secretome reveals clusters of secreted protein families as high priority effector candidates.

Saunders DGO, Win J, Cano LM, Szabo LJ, Kamoun S, et al. (2012) Using Hierarchical Clustering of Secreted Protein Families to Classify and Rank Candidate Effectors of Rust Fungi. PLoS ONE 7(1): e29847. doi:10.1371/journal.pone.0029847 http://journals.plos.org/plosone/article?id=info:doi/10.1371/journal.pone.0029847

By clustering and classifying plant proteins detect protein groups (families) that will probably act against mold

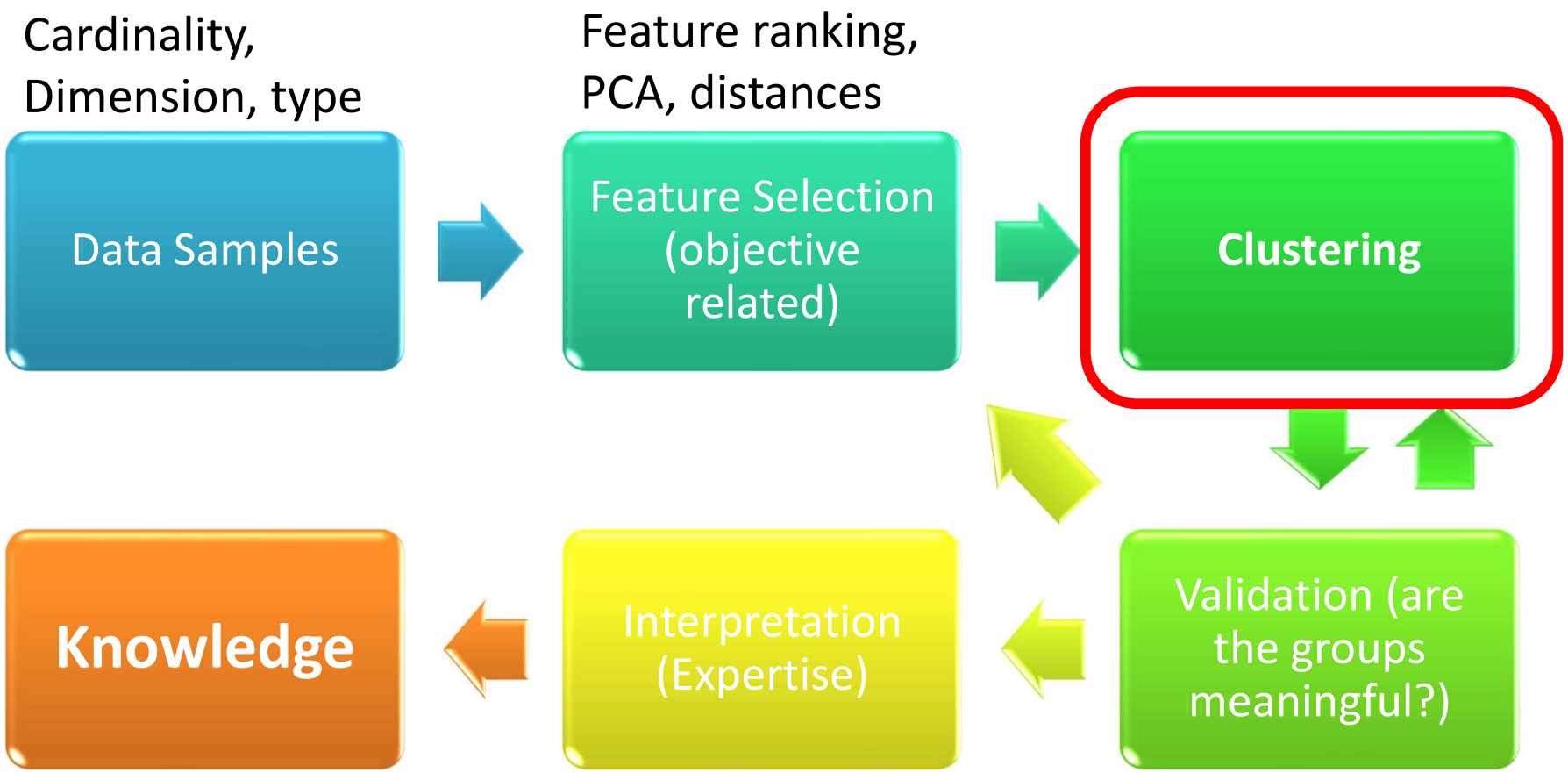
PLOS | ONE

Some clustering examples (3)

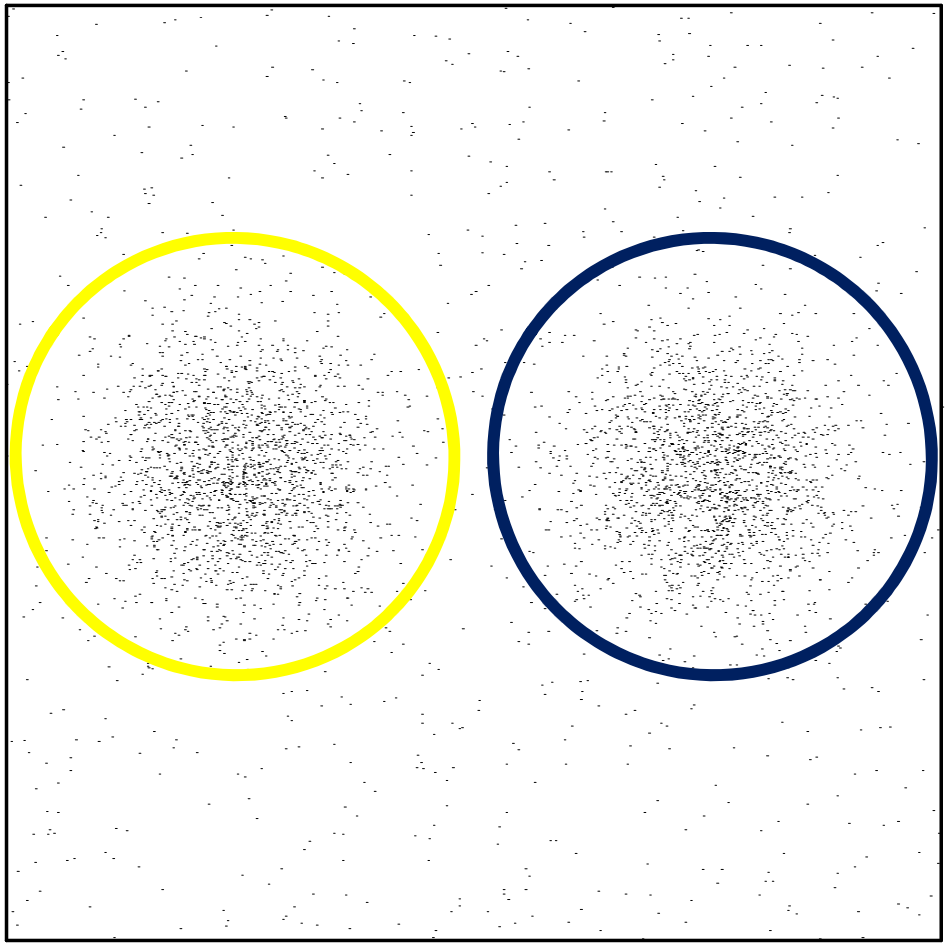


- Web search algorithms need to cluster.
- Not always known
- Cluster can be done in many ways (cluster the net by connections, cluster the results by text mining…)

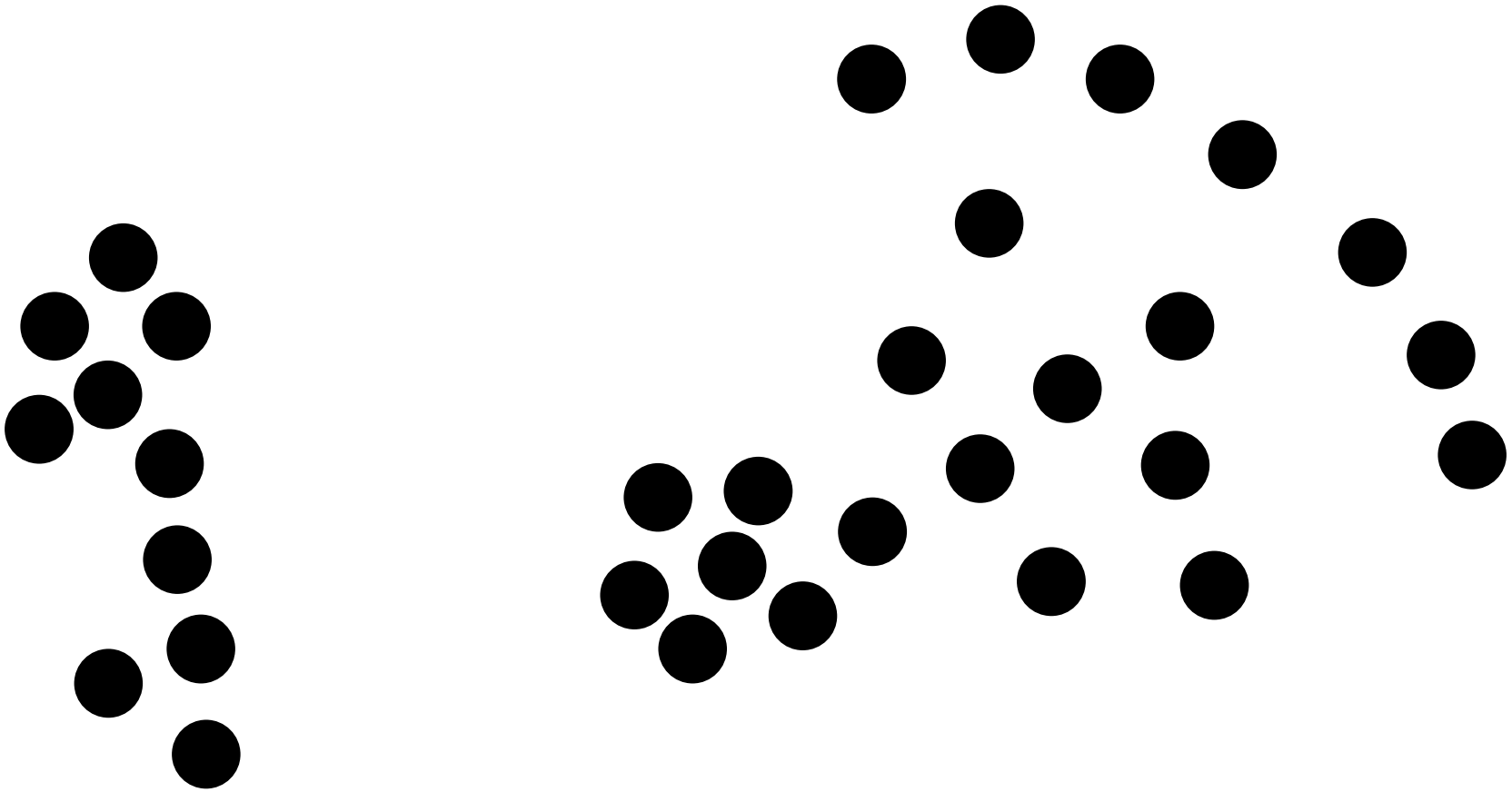
Clustering procedure



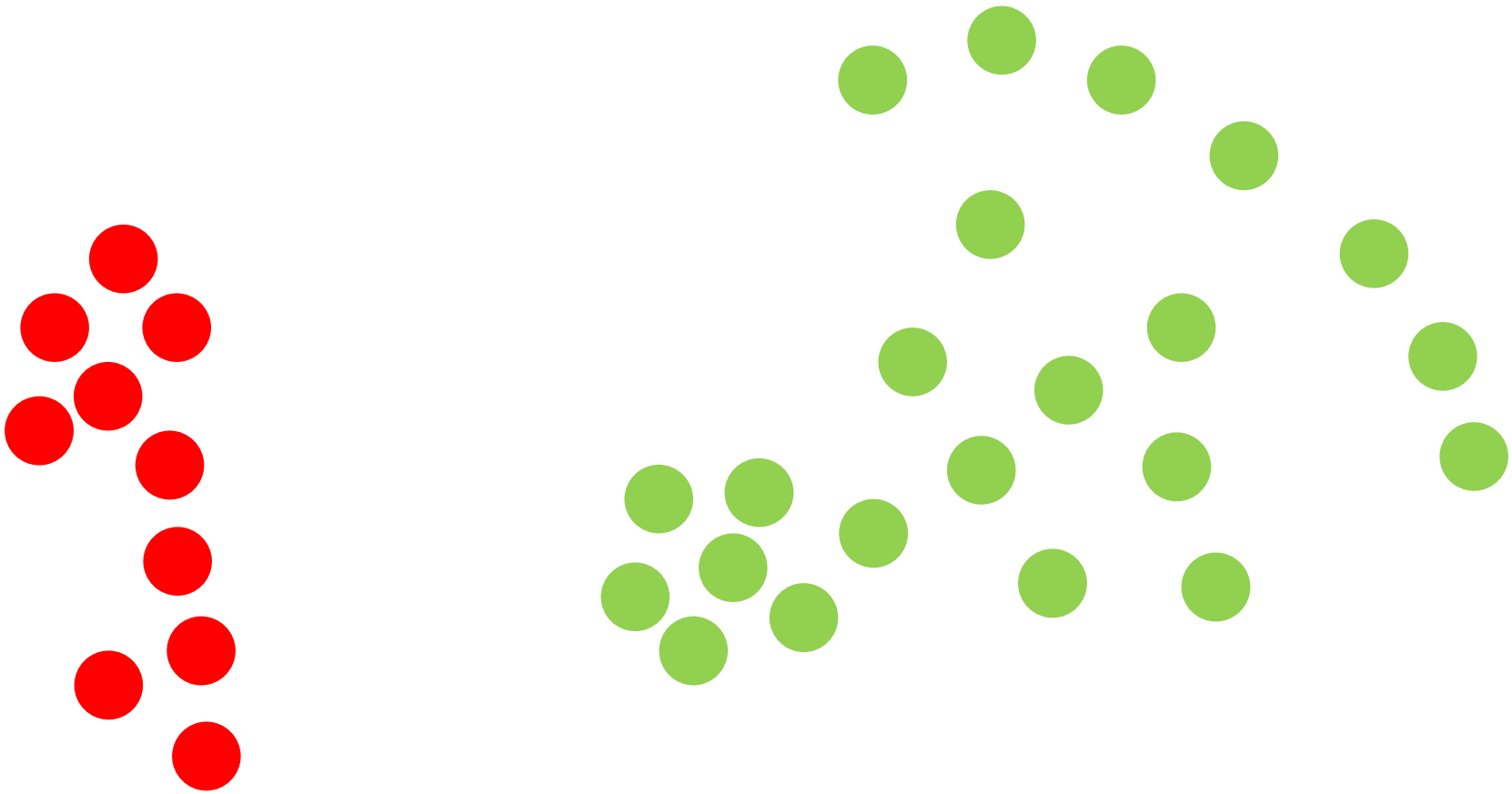
What is a cluster (revisited)



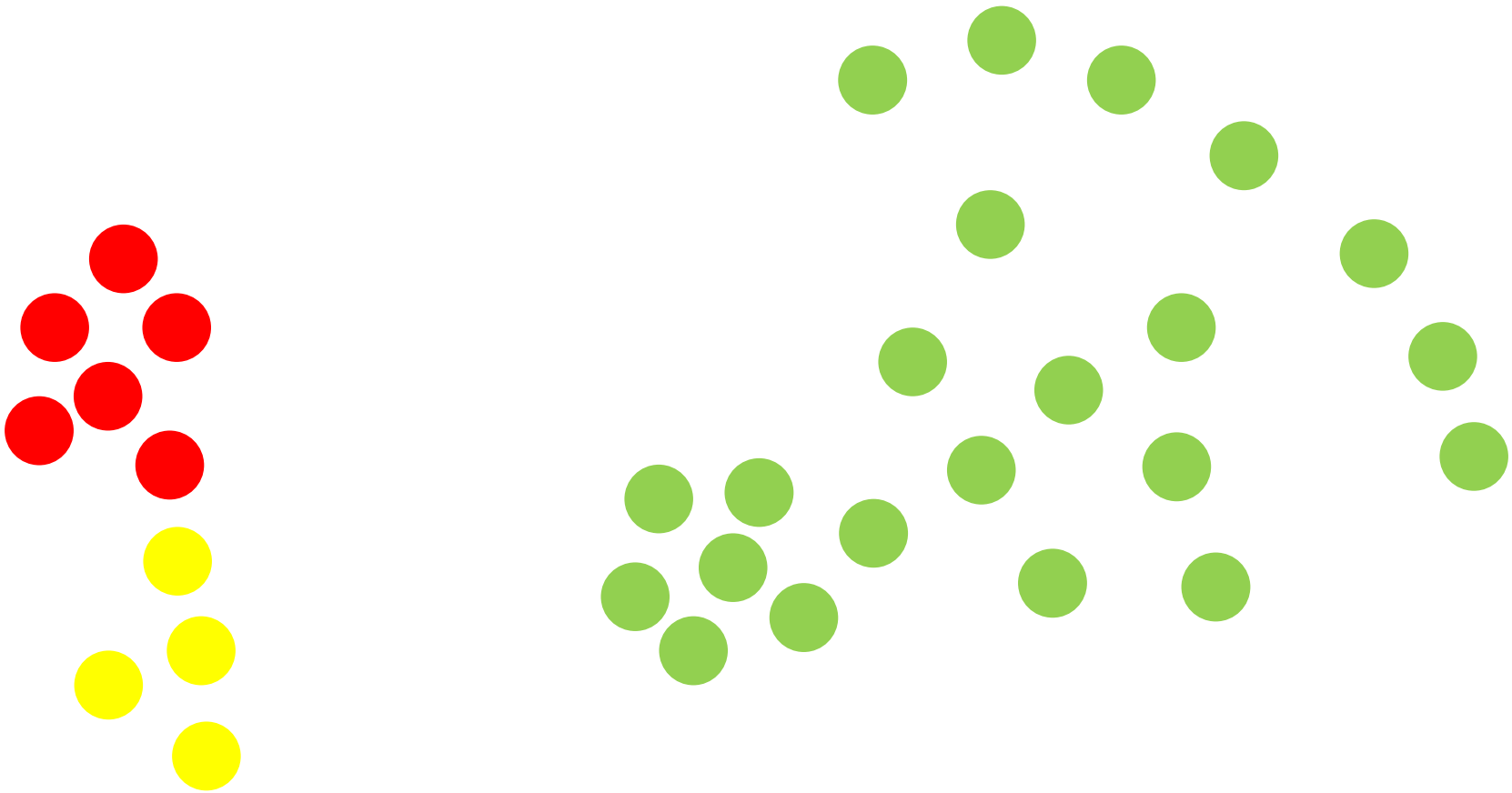
What is a cluster (revisited)



What is a cluster (revisited)



What is a cluster (revisited)



What is a cluster (revisited)

What is a cluster (revisited)

What is a cluster (revisited)

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
- Hierarchical clustering generates a tree instead of a single partition. Can be agglomerative (joining elements) or divisive (dividing the dataset)

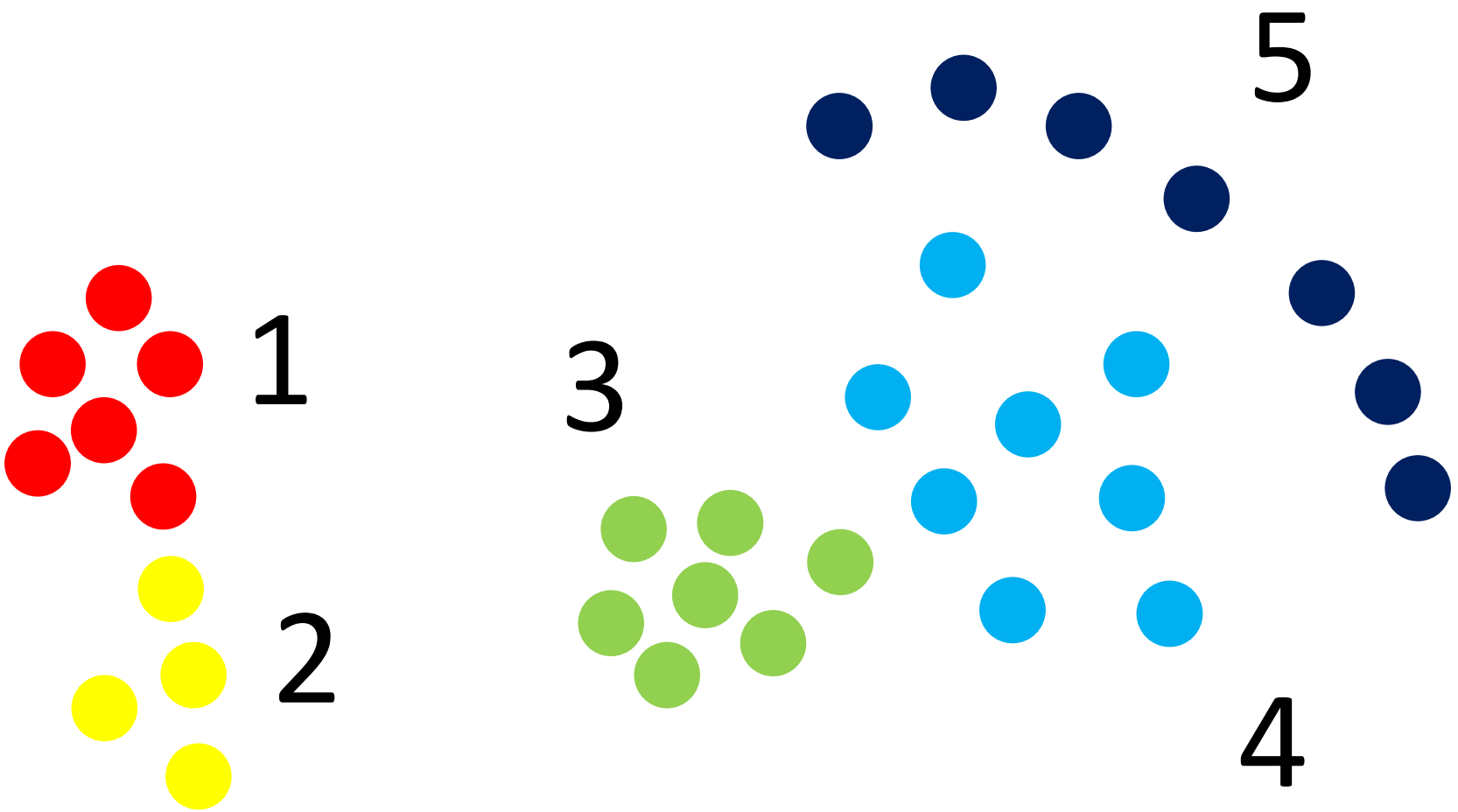
Flat clustering

- 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.

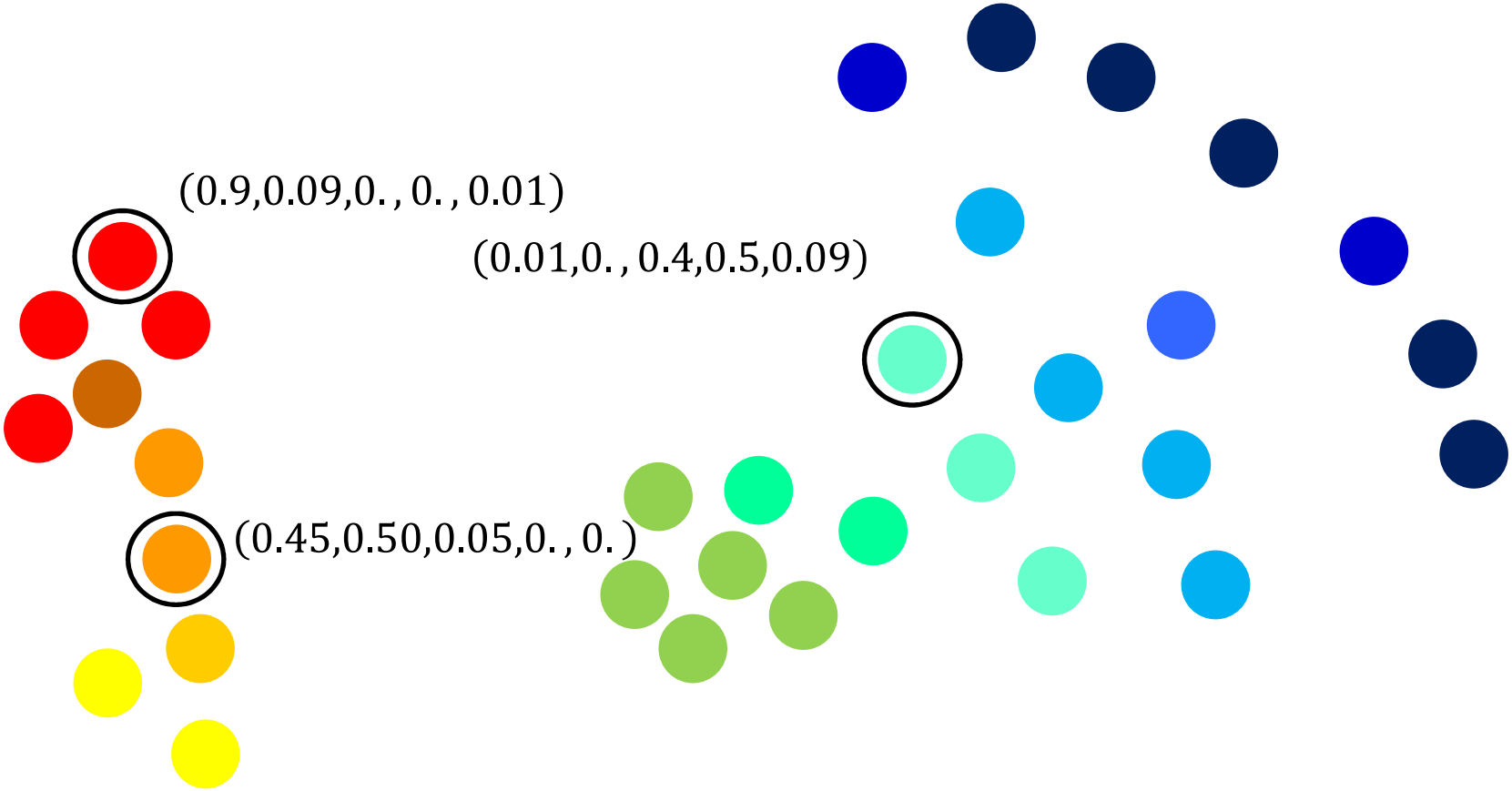
$$\overrightarrow{Cl}(i) = (u_1, u_2, \dots u_l \dots, u_k)$$

- 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

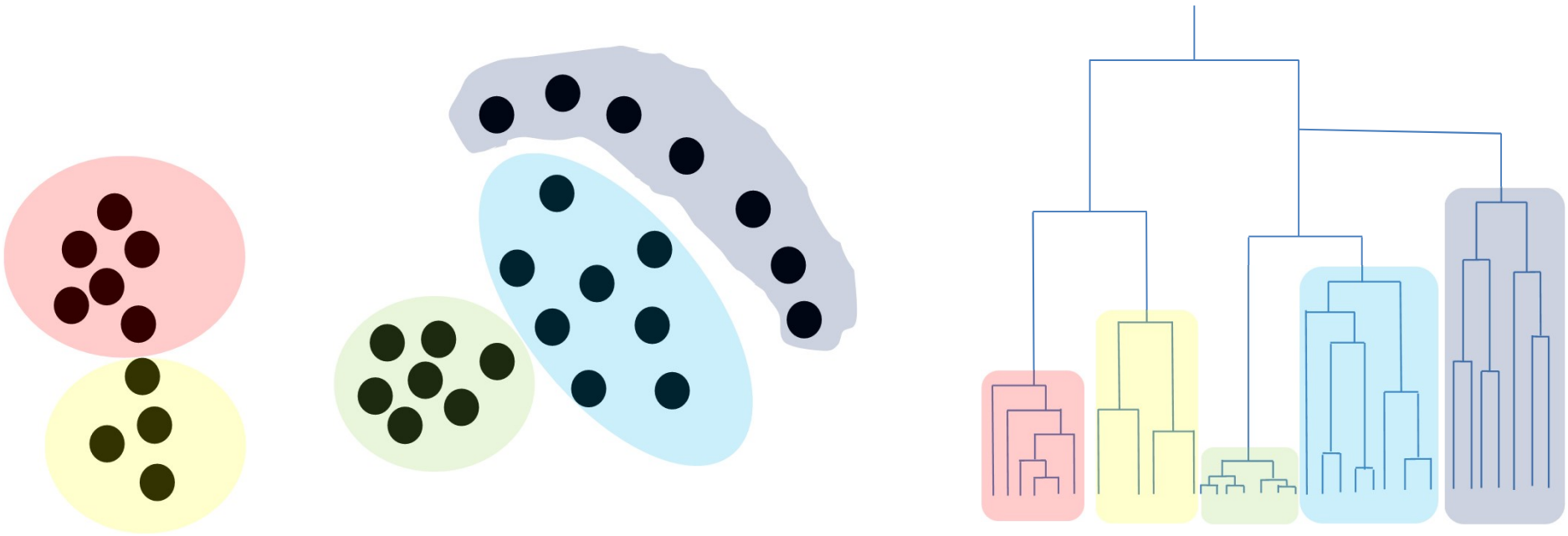
Fuzzy clustering



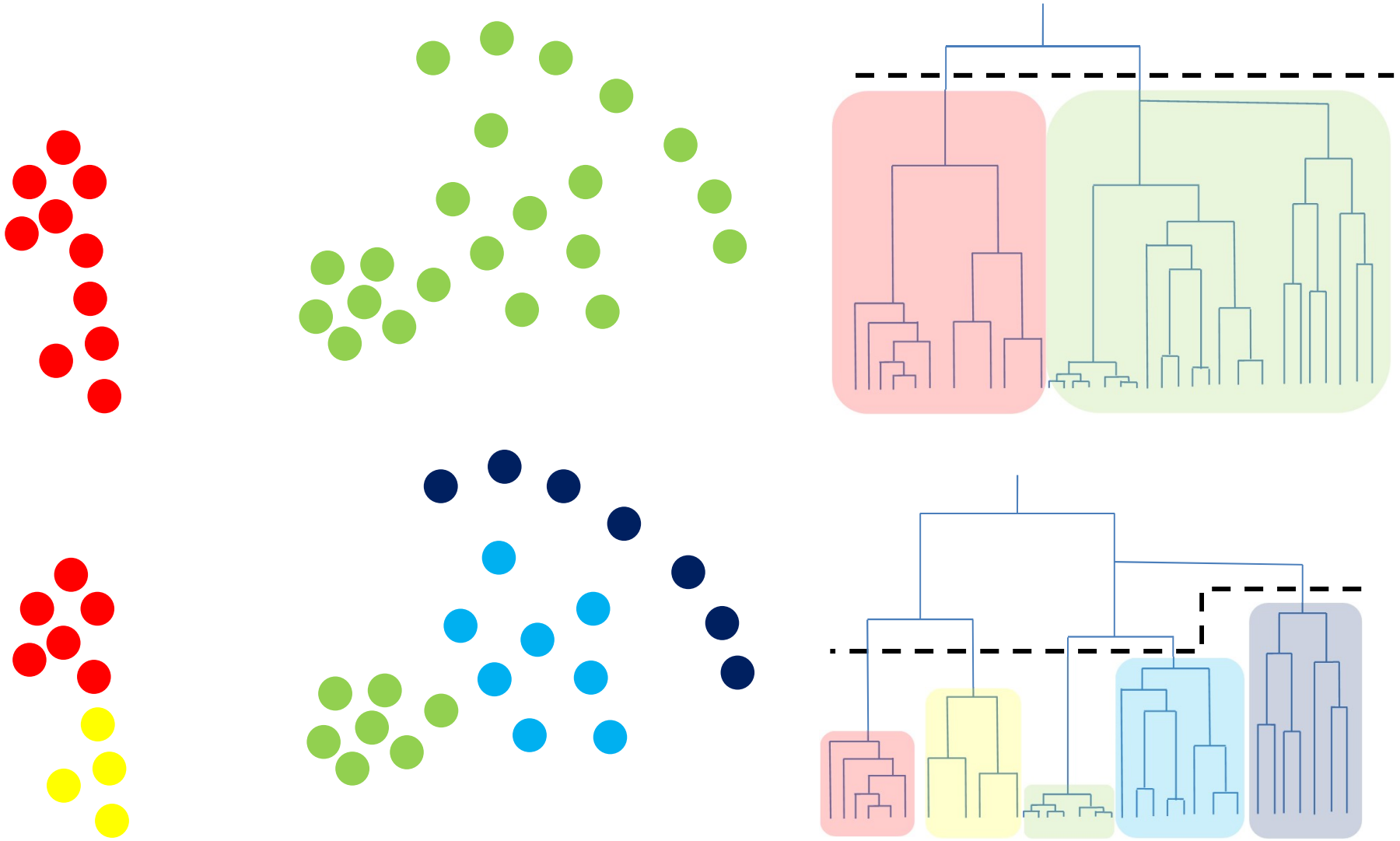
Hierarchical clustering

- 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
- 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

- Loss function:

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

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

k-means algorithm

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

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

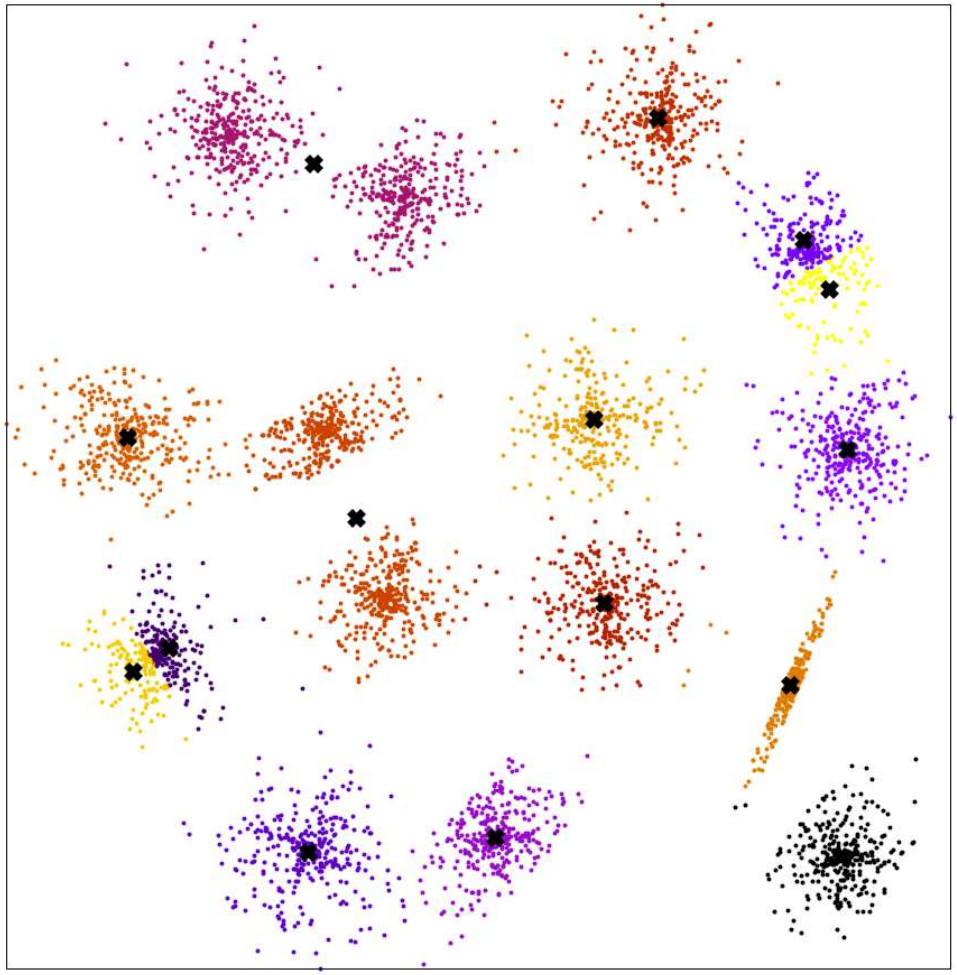
1. Initialize *k* cluster centroids $\{\overrightarrow{c_l}\}_{l=1}^{k}$ (randomly from the data set).
2. Repeat until no instance changes its cluster membership:
   - Decide the cluster membership of instances by assigning them to the nearest cluster centroid

     $$z_i = argmin_l(d(\overrightarrow{c_l}, \overrightarrow{x_i}))$$ *Minimize intra distance*
   - Update the *k* cluster centroids based on the assigned cluster membership

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

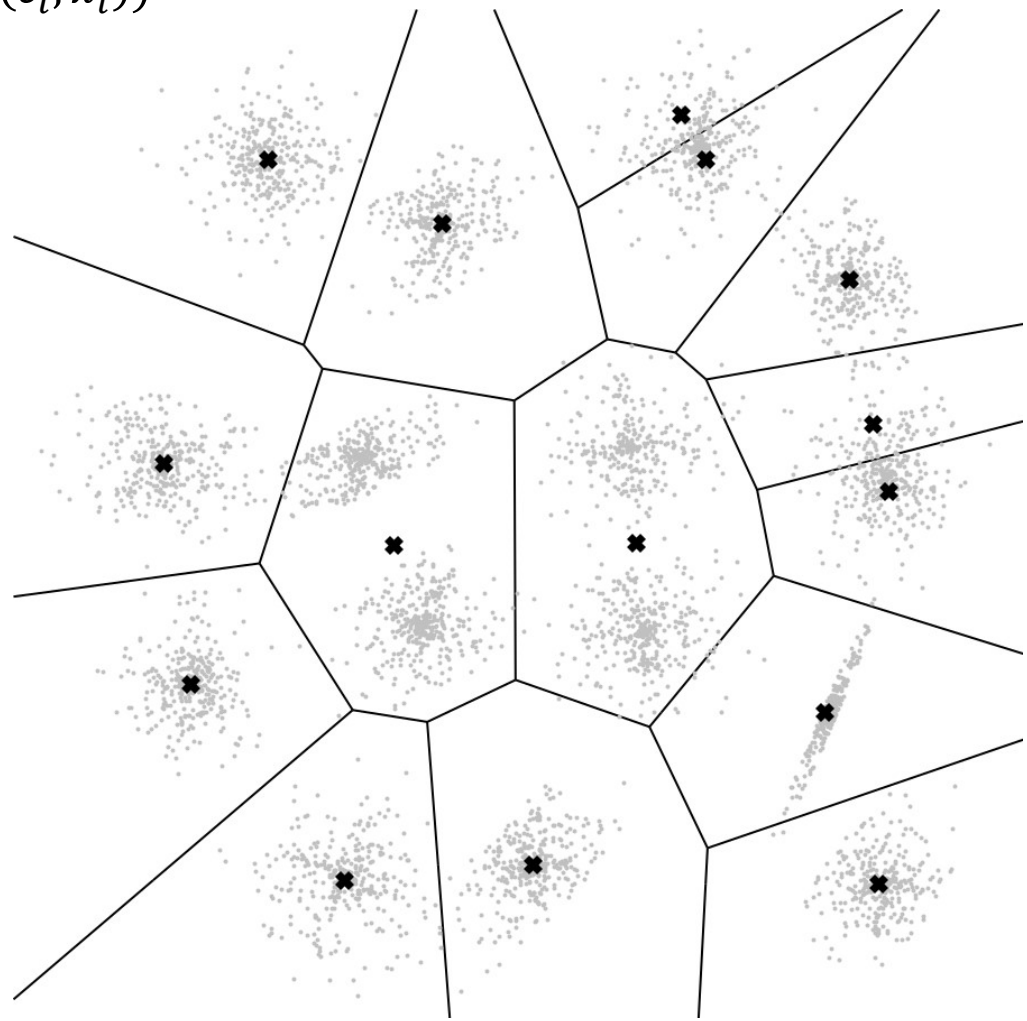
k-means illustration

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



Partitioning the space

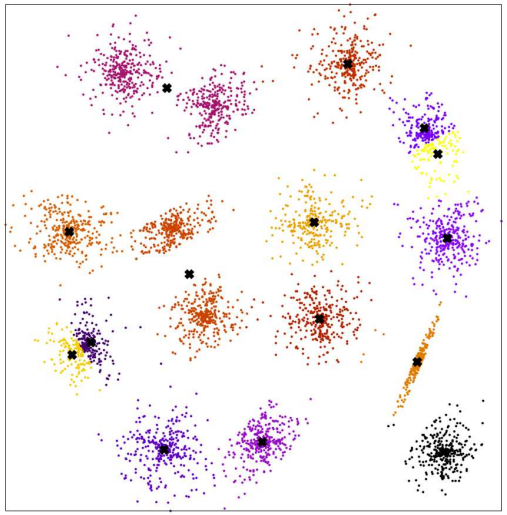
$z_i = argmin_l(d(\overrightarrow{c_l}, \overrightarrow{x_i}))$

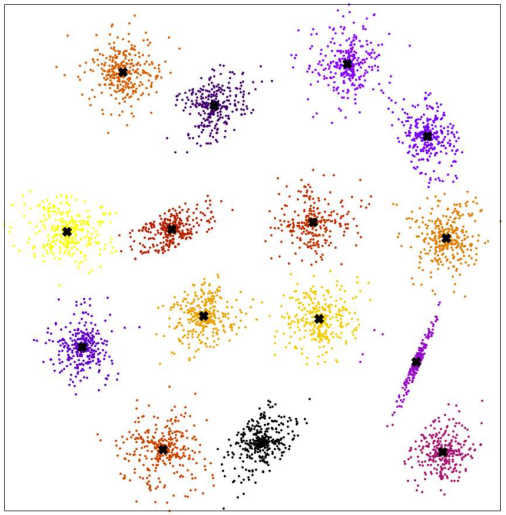


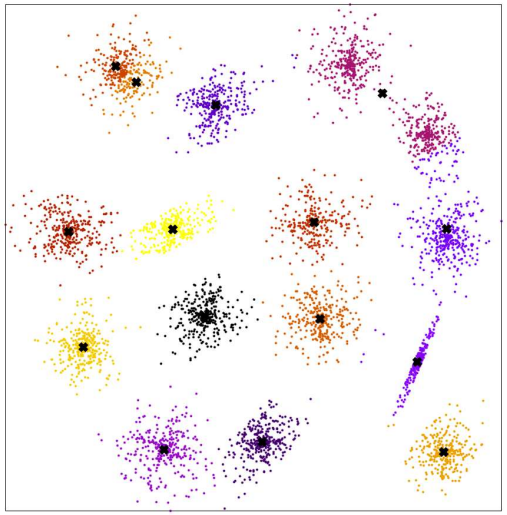
K-means weakness

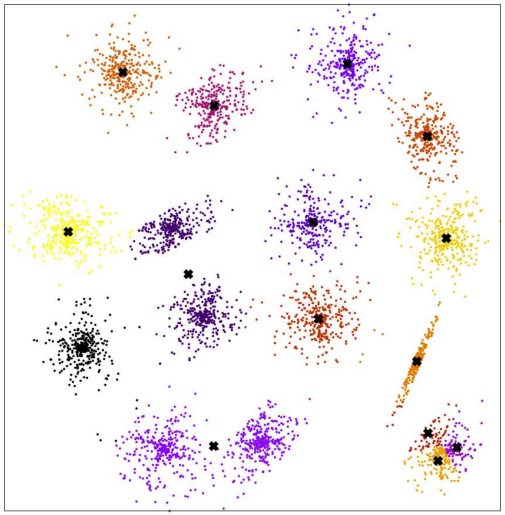
- Initialization sensitive (local optimization) →k-means++
- Which k-employing →Scree test
- Sensitive to outliers →k-medoids
- employs Euclidean distance →k-medoids
- Only for spherical clusters →kernel k-means (advanced)

Different initializations





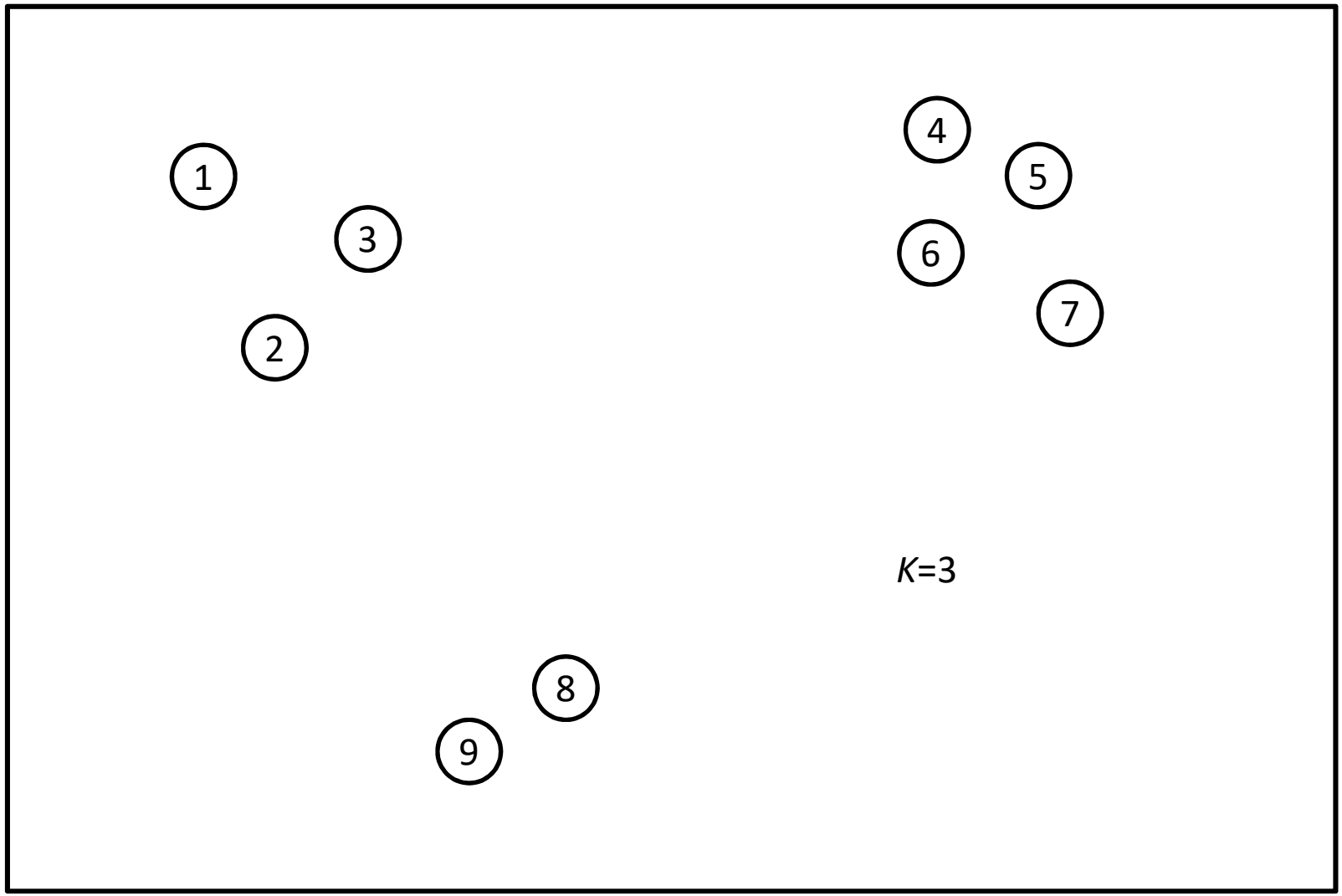




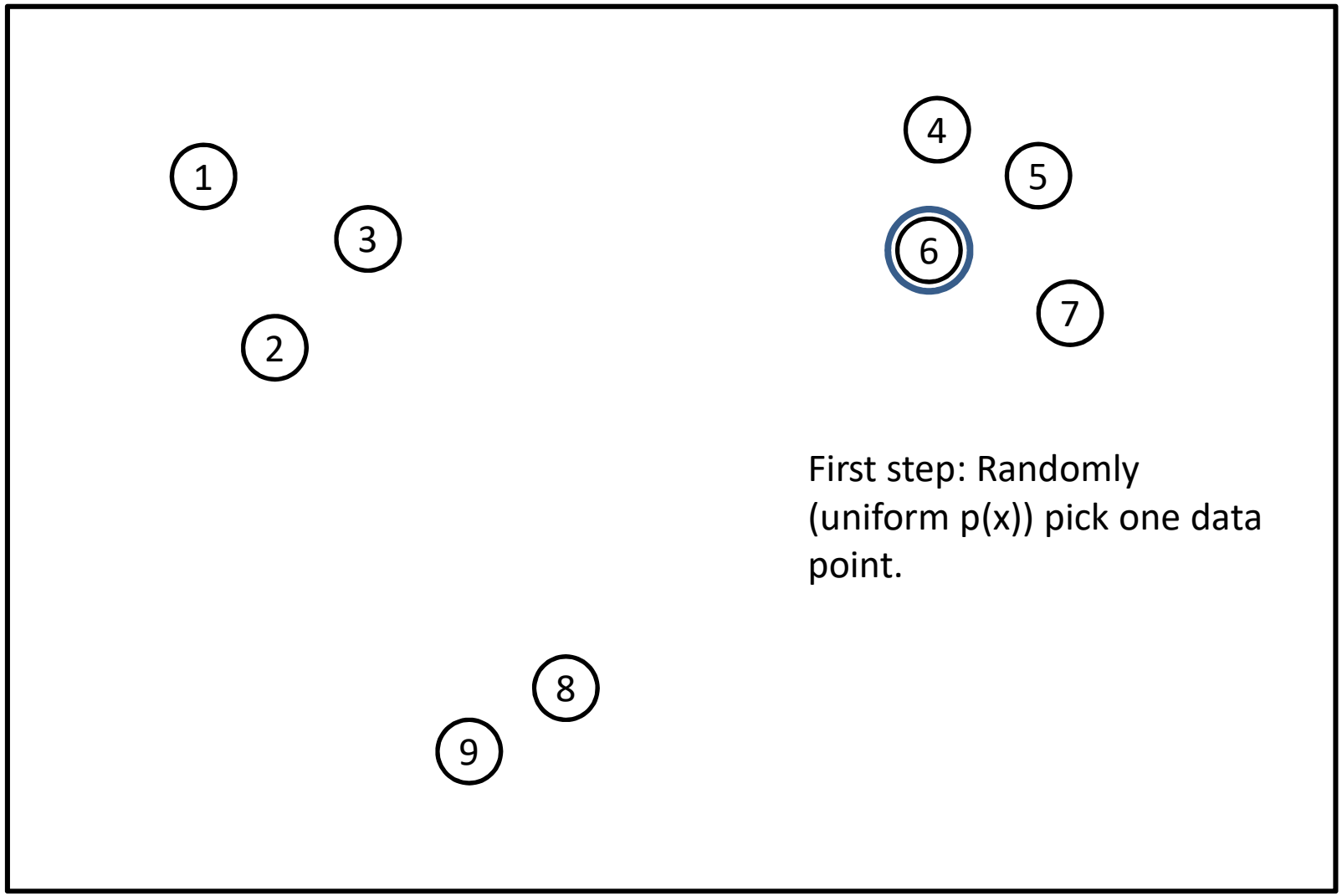
Better initialization: *k*-means++

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

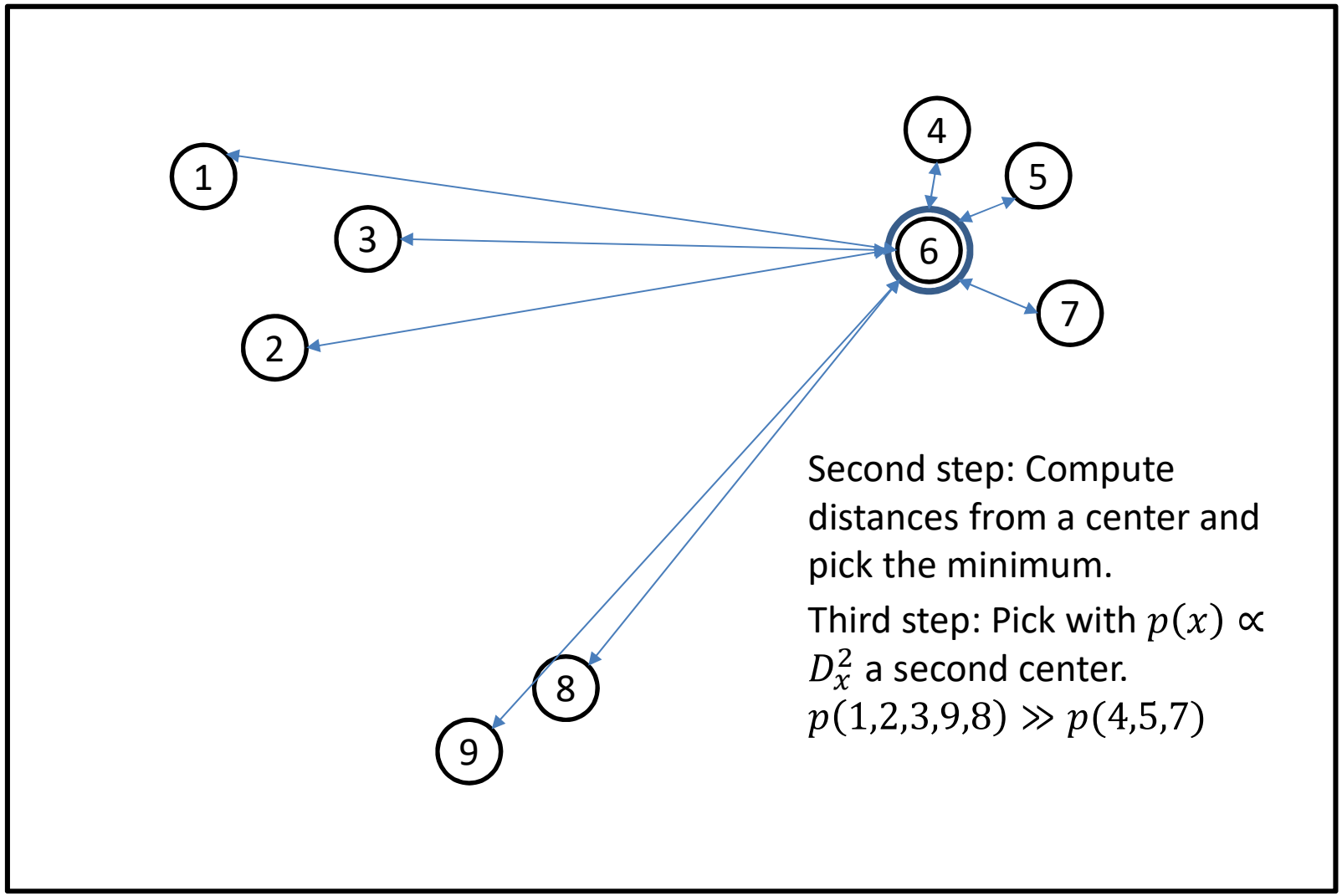
K-means ++



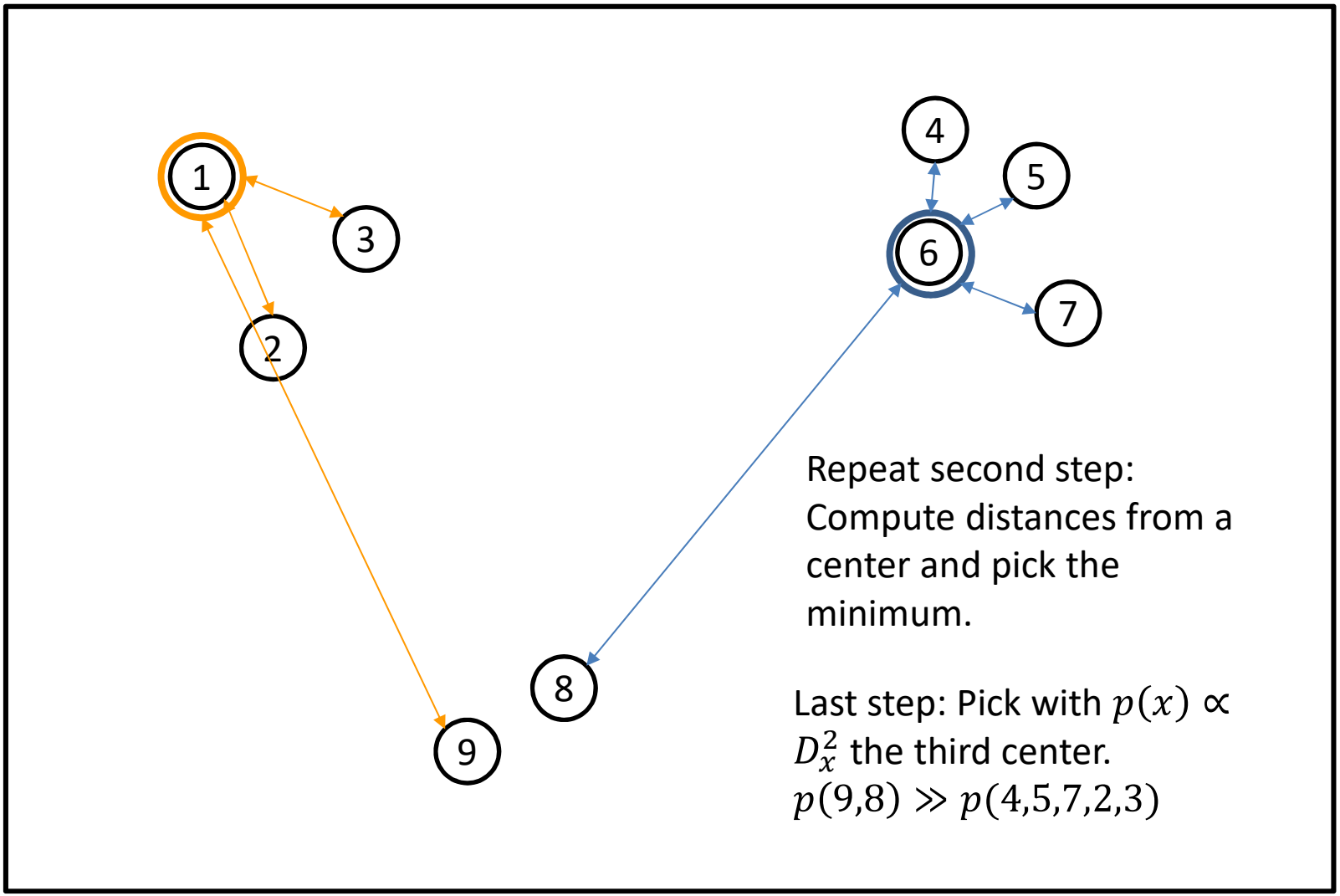
K-means ++


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

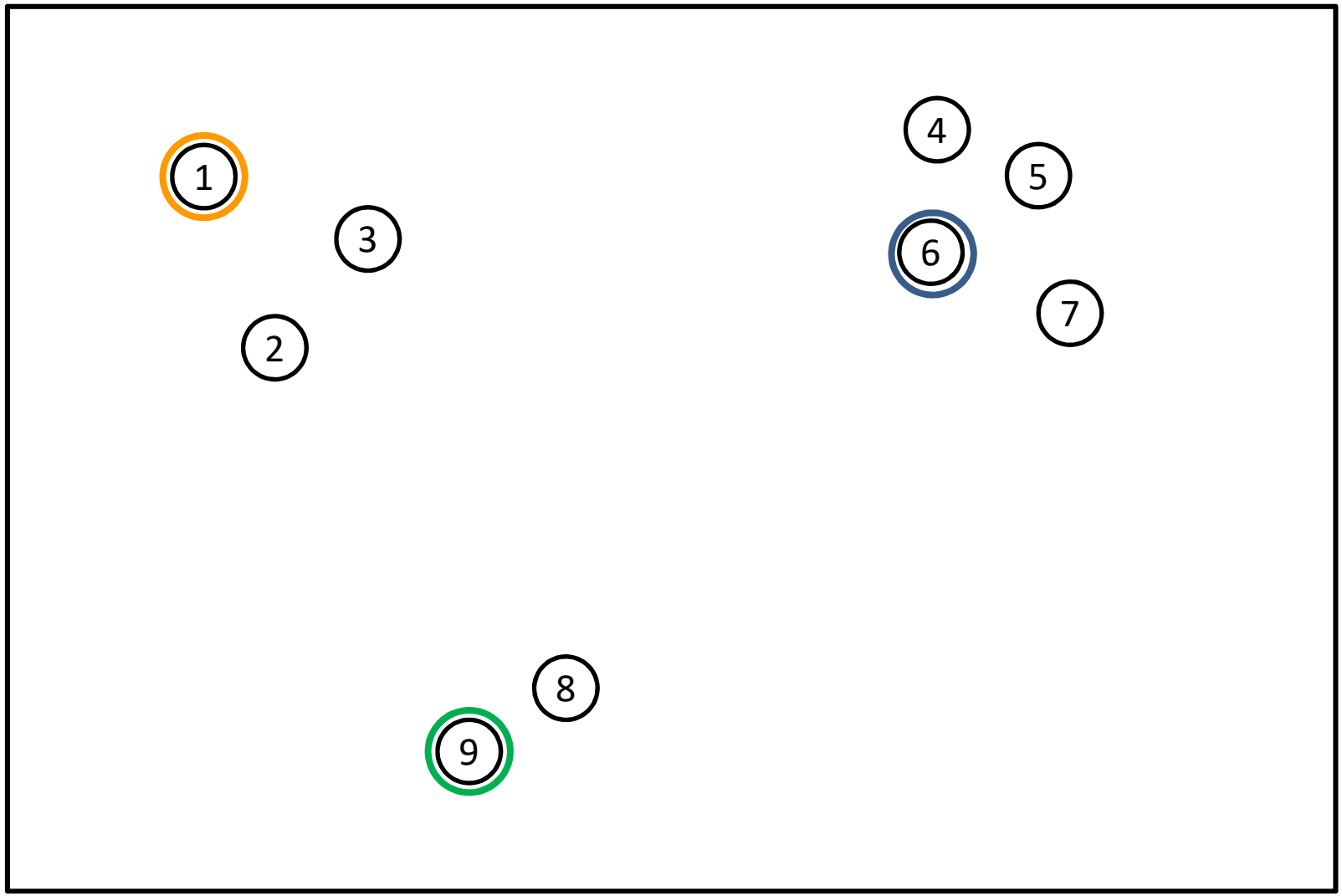
K-means ++


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

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

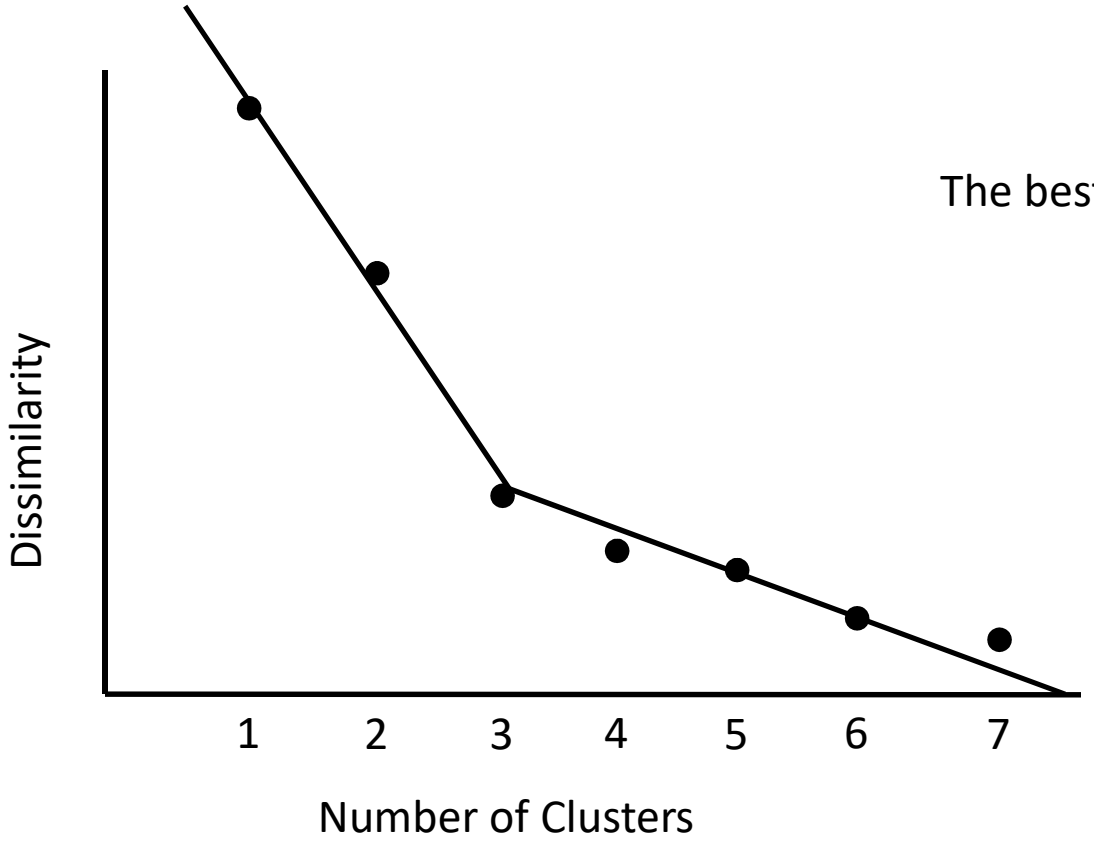
K-means ++



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)

Scree test for k-means

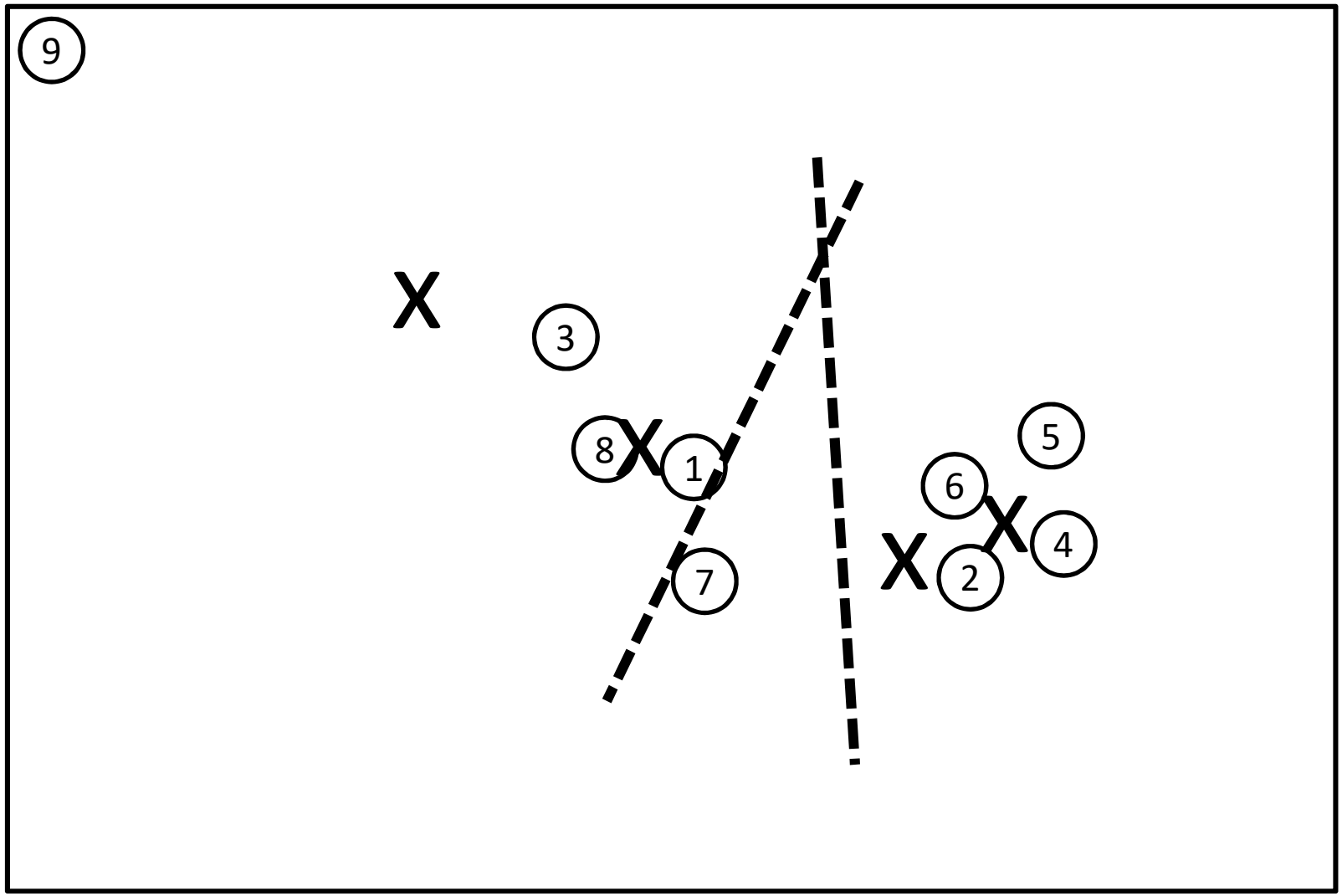


The best K is 3.

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

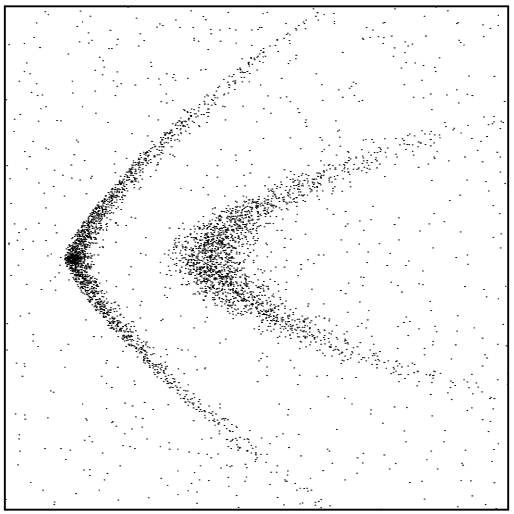


K-medoids

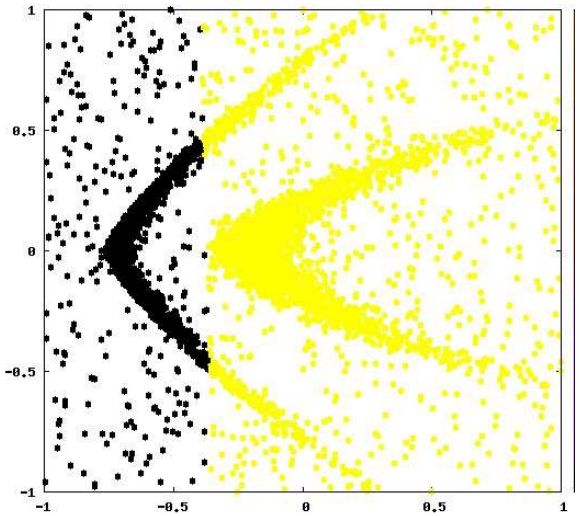
- It can be used with all kind of distances.
- Reduces the impact of outliers.
- Instead of working with centroids, it works with medoids, i.e. the most central element of the cluster.
- The cluster medoid is the element with minimum sum of distances to the rest of the elements of the cluster

K-means weakness

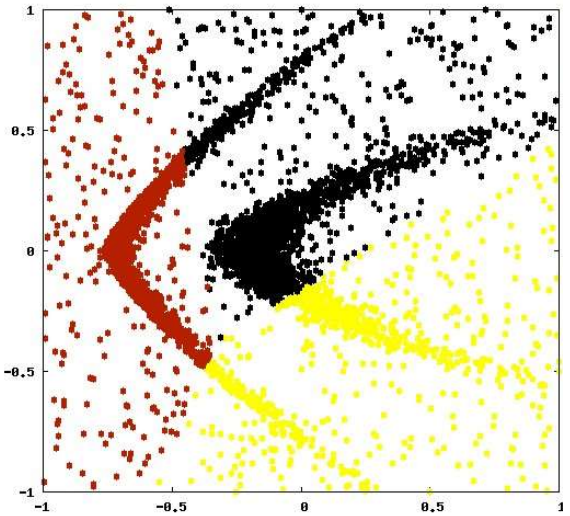
- Initialization sensitive (local optimization) →k-means++
- Which k-employing →Scree test
- Sensitive to outliers →k-medoids
- employs Euclidean distance →k-medoids
- Only for spherical clusters →kernel k-means (advanced)



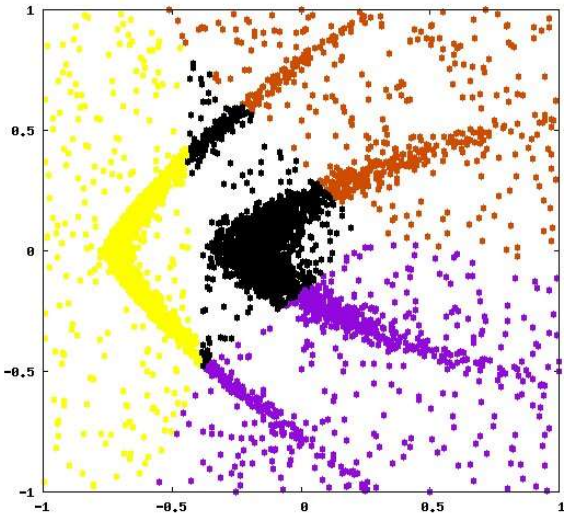
K=2



K=3



K=4



Kernel k-means

Observation: The k-means algorithm only needs to compute the distances from the centers of the clusters to the data points.

Let's use the kernel trick!

- Project the data set into a feature space by means of a nonlinear mapping $\overrightarrow{x_i} \rightarrow \overrightarrow{\phi_i}$.
- Distances in the feature space can be easily computed with the kernel trick:

$$\left\|\overrightarrow{\phi_i} - \overrightarrow{\phi_j}\right\|^2 = K^{ii} + K^{jj} - 2K^{ij}$$

- The loss function and the centroids are analogous to the ones in standard k-means, but in the feature space.

$$O(z) = \sum_{l=1}^{k}\sum_{i=1}^{n} \delta(z_{i,}l)\left\|\overrightarrow{\phi_i} - \overrightarrow{v_l}\right\|^2$$

$$\overrightarrow{v_l} = \frac{\sum_{i=1}^{n} \delta(z_{i,}l)\overrightarrow{\phi_i}}{\sum_{i=1}^{n} \delta(z_{i,}l)}$$

Kernel k-means

- We can compute the Gramm matrix as in the case of kernel PCA.

$$G^{ij} = -\frac{1}{2}\left(K^{ij} - \frac{1}{n}\sum_{i=1}^{n} K^{ij} - \frac{1}{n}\sum_{j=1}^{n} K^{ij} + \frac{1}{n^2}\sum_{j=1}^{n}\sum_{i=1}^{n} K^{ij} \right)$$

- How to compute the distance from each element to a cluster center?
  - We cannot compute directly $\overrightarrow{v_l} = \frac{\sum_{i=1}^{n} \delta(z_i, l)\overrightarrow{\phi_i}}{\sum_{i=1}^{n} \delta(z_i, l)}$
  - But we can compute the value of $\left\|\overrightarrow{\phi_i} - \overrightarrow{v_l}\right\|^2$

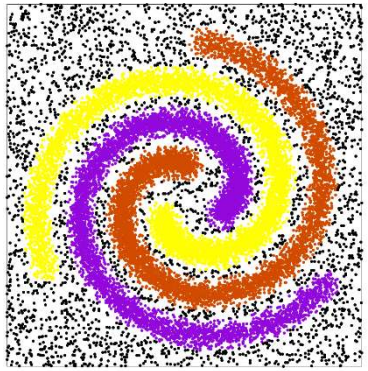
Kernel k-means

$$d_{il}{}^2 = \left\|\overrightarrow{\phi_i} - \overrightarrow{v_l}\right\|^2 = \left(\overrightarrow{\phi_i} - \overrightarrow{v_l}\right) \cdot \left(\overrightarrow{\phi_i} - \overrightarrow{v_l}\right) = \overrightarrow{\phi_i} \cdot \overrightarrow{\phi_i} - 2\overrightarrow{\phi_i} \cdot \overrightarrow{v_l} + \overrightarrow{v_l} \cdot \overrightarrow{v_l}$$
$$= G^{ii} - 2\frac{\sum_{j=1}^{n} \delta\left(z_{i,} l\right) G^{ij}}{\sum_{j=1}^{n} \delta\left(z_{j,} l\right)} + \frac{\sum_{j=1}^{n} \sum_{k=1}^{n} \delta(z_k, l) \delta\left(z_{j,} l\right) G^{kj}}{\left(\sum_{j=1}^{n} \delta\left(z_{j,} l\right)\right)^2}$$

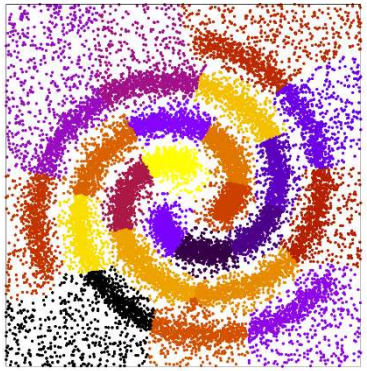
The algorithm:

1. Compute the Gramm matrix.
2. Randomly pick k centers.
3. Iterate the equation $z_i = argmin_l\left(d_{il}{}^2\right)$ until convergence

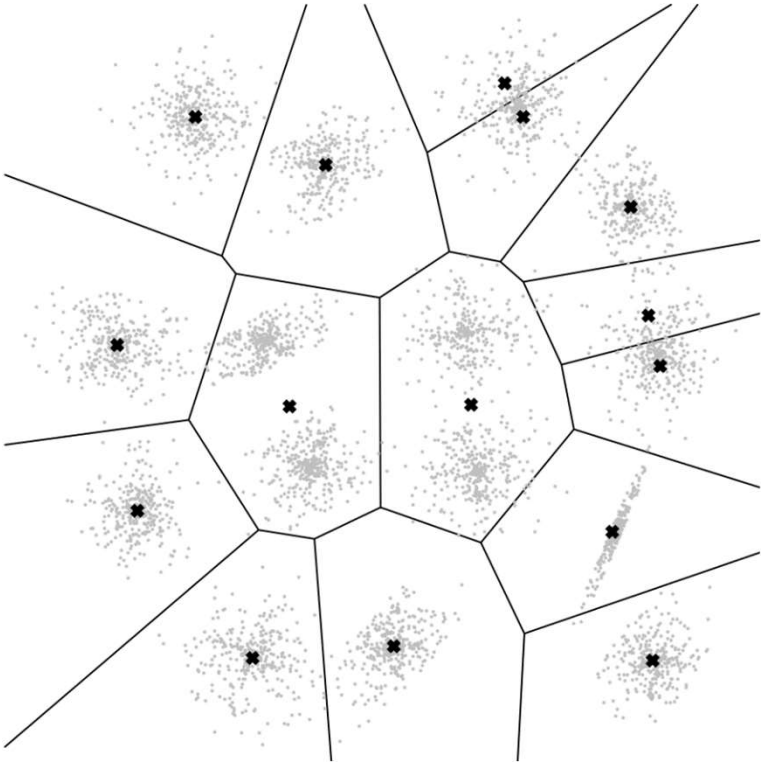
Why *k*-means?



Expected result



K-means result



- **K-means provides an optimal Voronoi partition of the data.**
- **Many other methods are somehow related to k-means.**

Fuzzy c-means: A fuzzy clustering algorithm

Bezdek, J. C., Ehrlich, R., & Full, W. (1984). FCM: The fuzzy c-means clustering algorithm. *Computers & Geosciences*, *10*(2), 191-203.

Fuzzy c-means

- It can be considered a version of k-means with a soft assignation criteria.

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

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

- Therefore, the objective function and the optimization algorithm must be adapted to fulfill these conditions.

Fuzzy c-means algorithm

- Objective function:

$$O(\mathbb{U}) = \sum_{l=1}^{k} \sum_{i=1}^{n} (u_{il})^m \|\overrightarrow{x_i} - \overrightarrow{c_l}\|^2$$

- $\mathbb{U}$ is a $n \times k$ matrix with the membership of the n elements in the k clusters.
- m is the fuzzification parameter. Usually m=2.
- $\overrightarrow{c_l}$ is the vector with the coordinates of the l cluster center. As in the case of k-means, it is the average value of the cluster coordinates. $\overrightarrow{c_l} = \frac{\sum_{i=1}^{n} (u_{il})^m \overrightarrow{x_i}}{\sum_{i=1}^{n} (u_{il})^m}$

Fuzzy c-means algorithm

- Random initialization of $\mathbb{U}$
- Iterative optimization:
  - Given $\mathbb{U}$ compute the centers:

$$\overrightarrow{c_l} = \frac{\sum_{i=1}^{n} (u_{il})^m \overrightarrow{x_i}}{\sum_{i=1}^{n} (u_{il})^m}$$

  - Given the centers, update $\mathbb{U}$:

$$u_{ij} = \frac{1}{\sum_{l=1}^{k} \left(\frac{\left\| \overrightarrow{x_i} - \overrightarrow{c_j} \right\|}{\left\| \overrightarrow{x_i} - \overrightarrow{c_l} \right\|} \right)^{2/m-1}}$$

- Repeat until the change in $\mathbb{U}$ is smaller than a given threshold

Fuzzy c-means algorithm

- Similar problems to k-means:
  - Initialization
  - Number of clusters
  - Sensitive to outliers
  - Spherical clusters

Hierarchical clustering algorithms

Based on slides by Evimaria Terzi
(http://www.cs.bu.edu/~evimaria/)

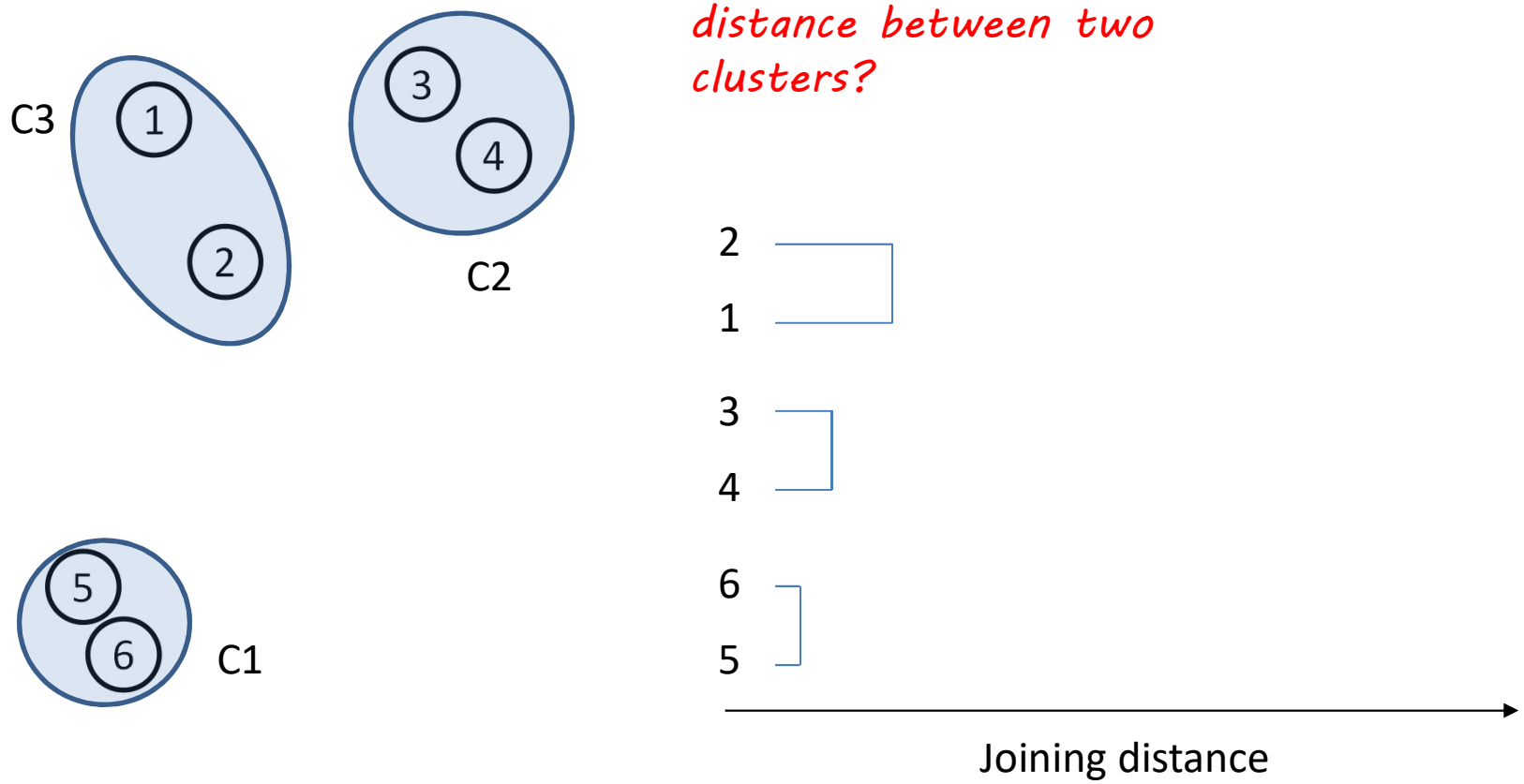
Hierarchical Clustering

- Two main types of hierarchical clustering
  - **Agglomerative:**
    - Start with the points as individual clusters
    - At each step, merge the closest pair of clusters until only one cluster (or **k** clusters) are left
  - **Divisive:**
    - Start with one, all-inclusive cluster
    - At each step, split a cluster until each cluster contains a point (or there are **k** clusters)
- Traditional hierarchical algorithms use a similarity or distance matrix
  - Merge or split one cluster at a time

Agglomerative clustering algorithm

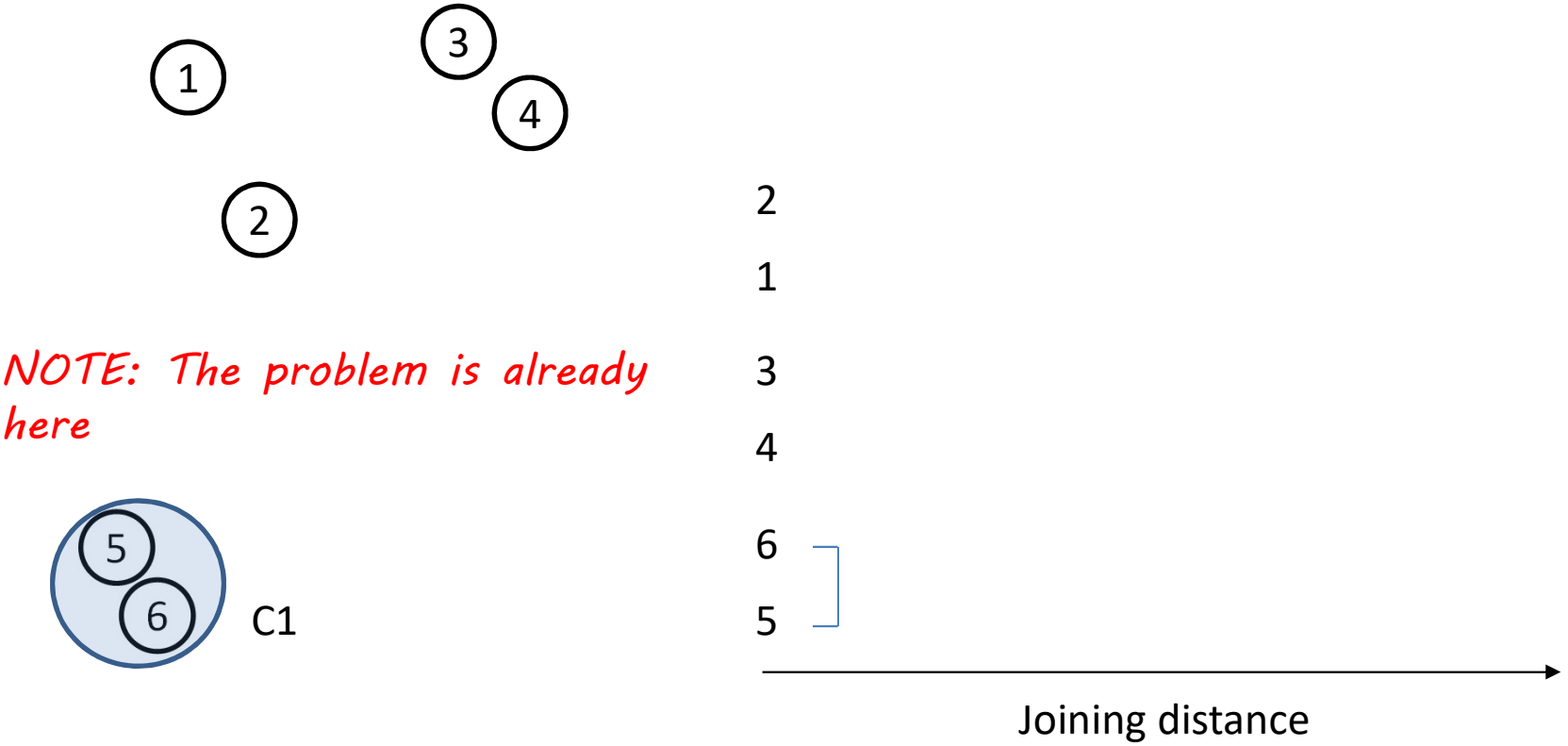
- Most popular hierarchical clustering technique
- Basic algorithm
  1. Compute the distance matrix between the input data points
  2. Let each data point be a cluster
  3. **Repeat**
  4. Merge the two closest clusters
  5. Update the distance matrix
  6. **Until** only a single cluster remains

Hierarchical clustering at glance



At each step: Pick the minimum distance and merge these elements

Hierarchical clustering at glance



At each step: Pick the minimum distance and merge these elements

Agglomerative clustering algorithm

- Most popular hierarchical clustering technique
- Basic algorithm
  1. Compute the distance matrix between the input data points
  2. Let each data point be a cluster
  3. **Repeat**
  4. Merge the two closest clusters
  5. Update the distance matrix
  6. **Until** only a single cluster remains
- Key operation is the computation of the distance between two clusters
  - Different definitions of the distance between clusters lead to different algorithms

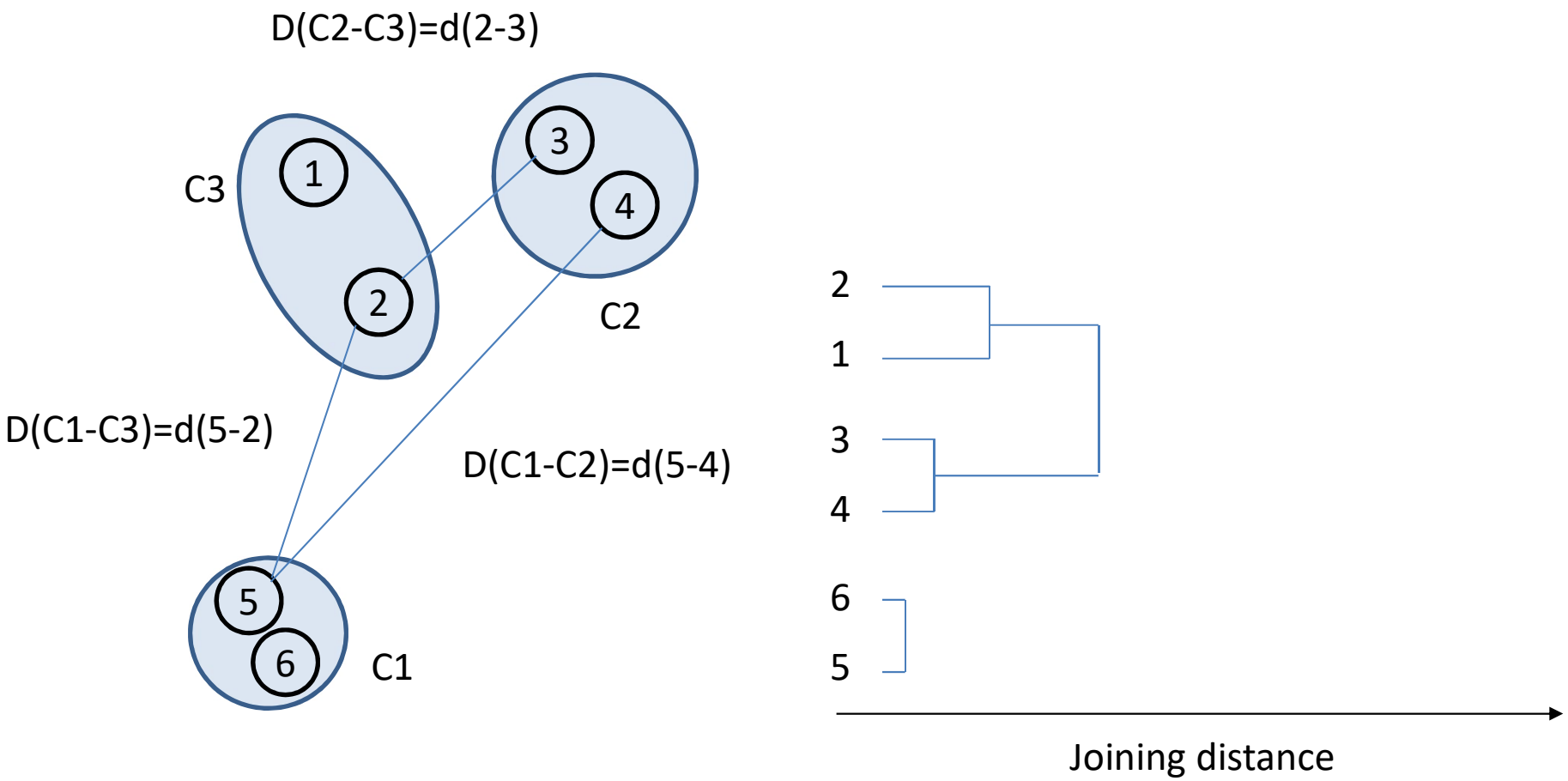
Single-link distance

- **Single-link distance** between clusters C_i and C_j is the ***minimum distance*** between any object in C_i and any object in C_j

- The distance is **defined by the two most similar (i.e nearest) objects**

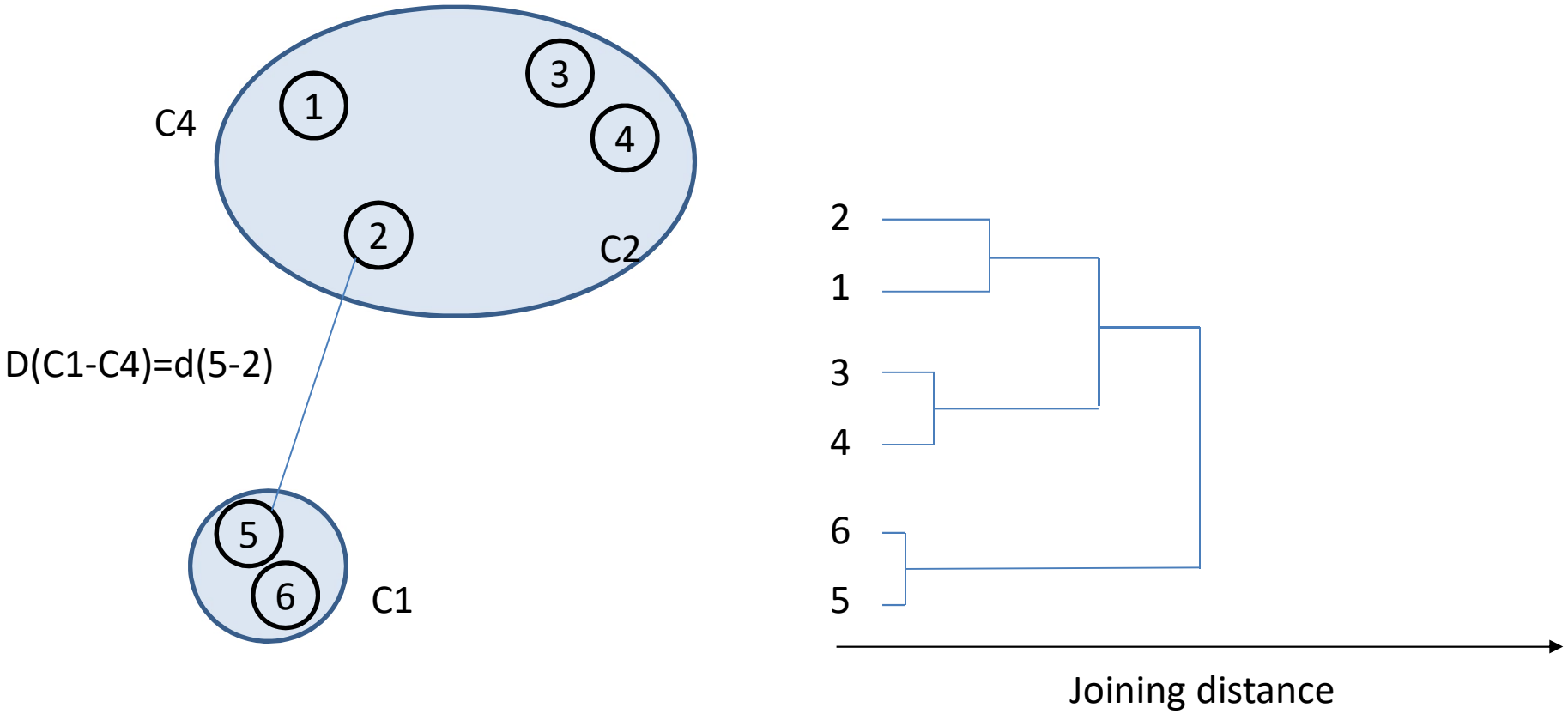
$$D_{sl}(C_i, C_j) = \min_{x,y} \left\{ d(x, y) \middle| x \in C_i, y \in C_j \right\}$$

$$D_{sl}(C_i, C_j) = \min_{x,y} \left\{ d(x,y) \middle| x \in C_i, y \in C_j \right\}$$



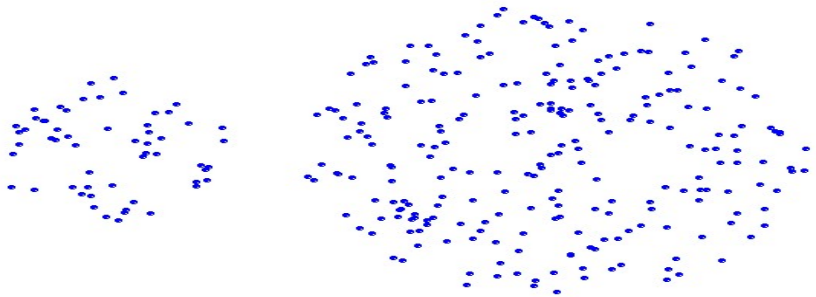
At each step: Pick the minimum distance and merge these elements

$$D_{sl}(C_i, C_j) = \min_{x,y} \left\{ d(x,y) \middle| x \in C_i, y \in C_j \right\}$$

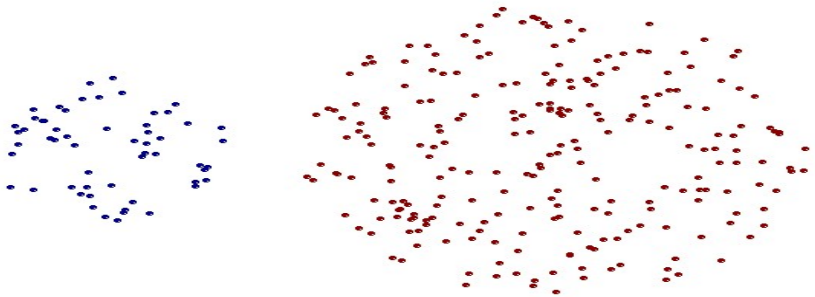


At each step: Pick the minimum distance and merge these elements

Strengths of single-link clustering



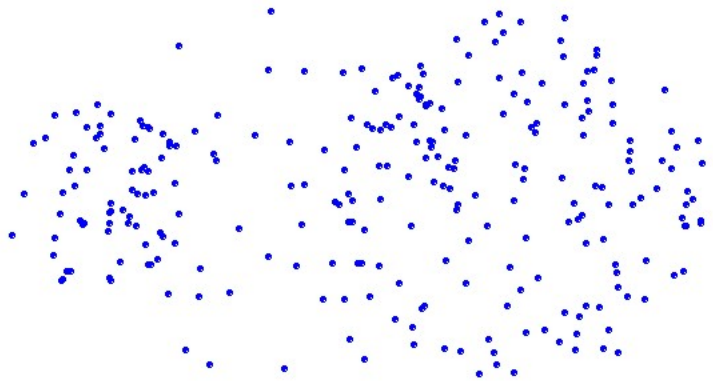
Original Points



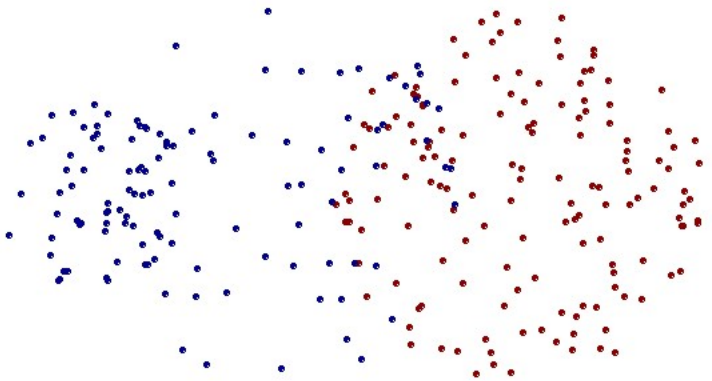
Two Clusters

- **Can handle non-elliptical shapes**

Limitations of single-link clustering



Original Points



Two Clusters

- **Sensitive to noise and outliers**
- **It produces long, elongated clusters**

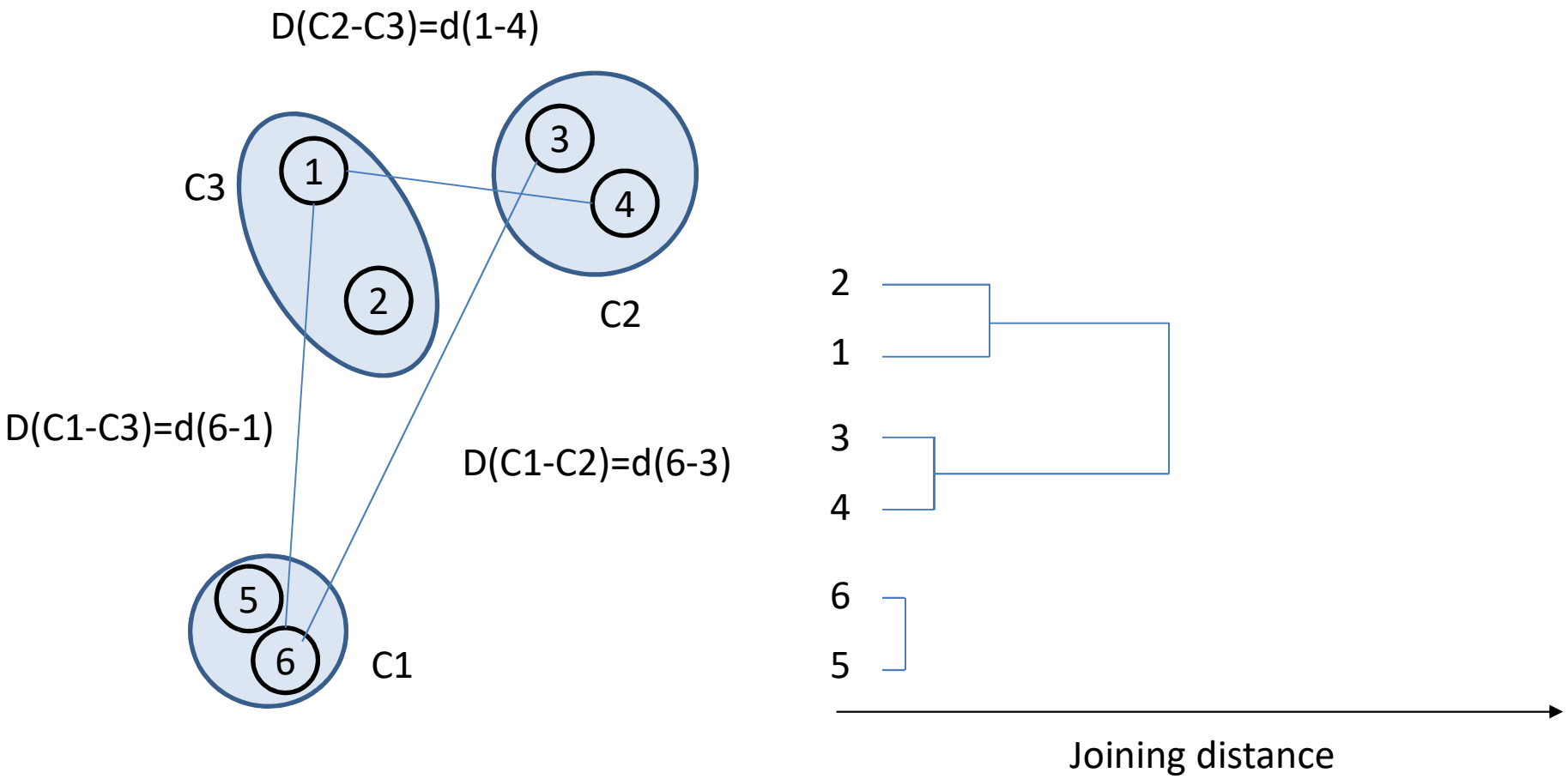
Complete-link distance

- **Complete-link distance** between clusters C_i and C_j is the ***maximum distance*** between any object in C_i and any object in C_j

- The distance is **defined by the two most dissimilar (i.e. furthest) objects**

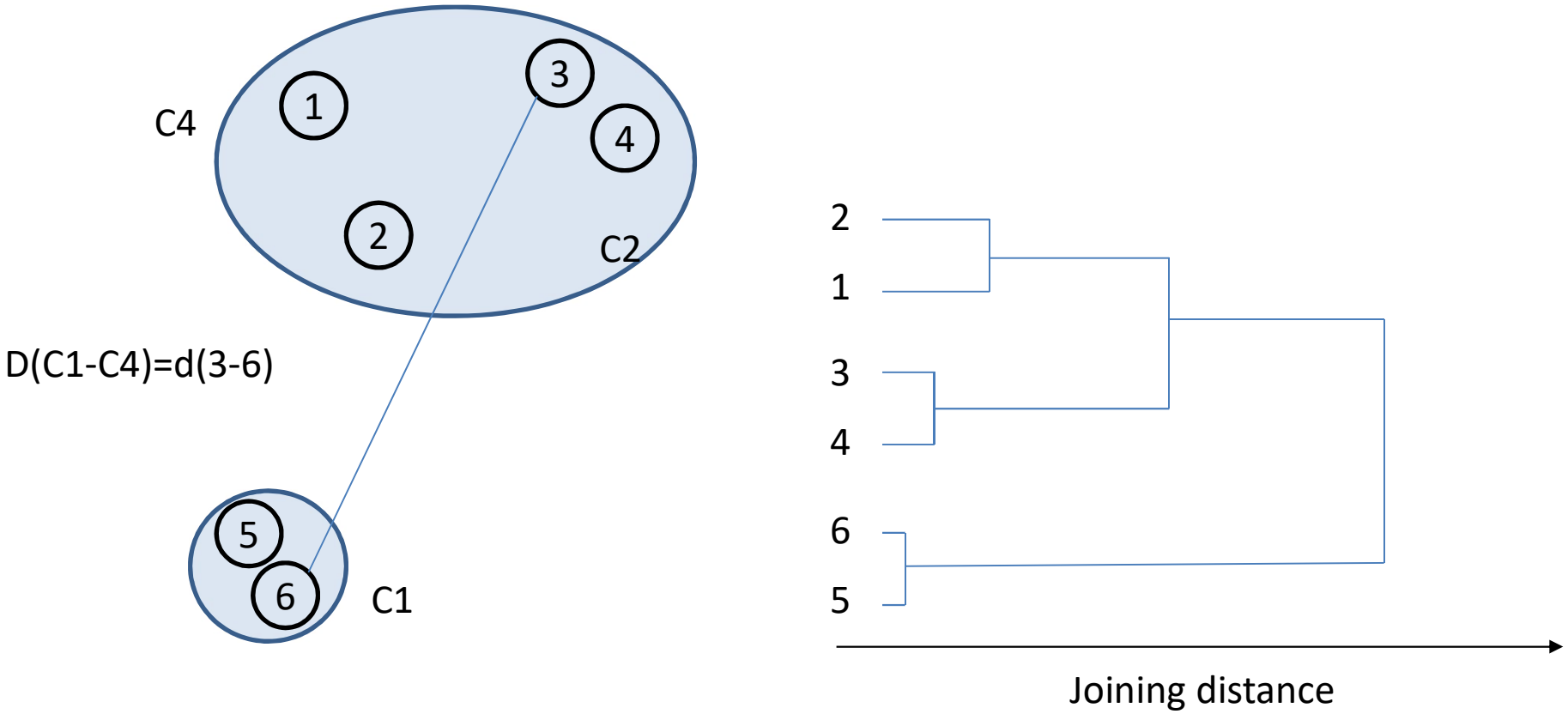
$$D_{cl}(C_i, C_j) = \max_{x,y} \left\{ d(x, y) \middle| x \in C_i, y \in C_j \right\}$$

$$D_{cl}(C_i, C_j) = \max_{x,y} \left\{ d(x,y) \middle| x \in C_i, y \in C_j \right\}$$



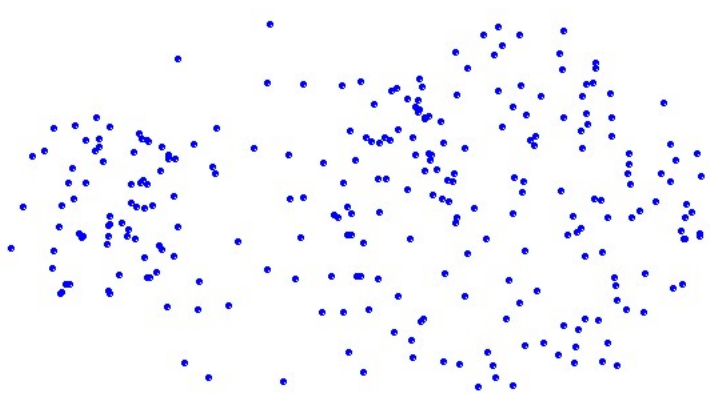
At each step: Pick the minimum distance and merge these elements

$$D_{cl}(C_i, C_j) = \max_{x,y} \left\{ d(x,y) \middle| x \in C_i, y \in C_j \right\}$$

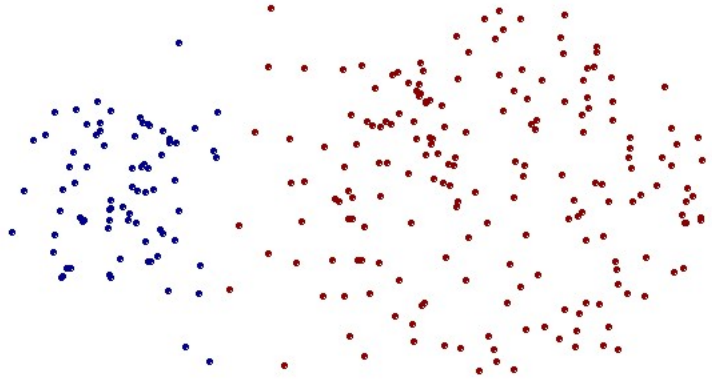


At each step: Pick the minimum distance and merge these elements

Strengths of complete-link clustering



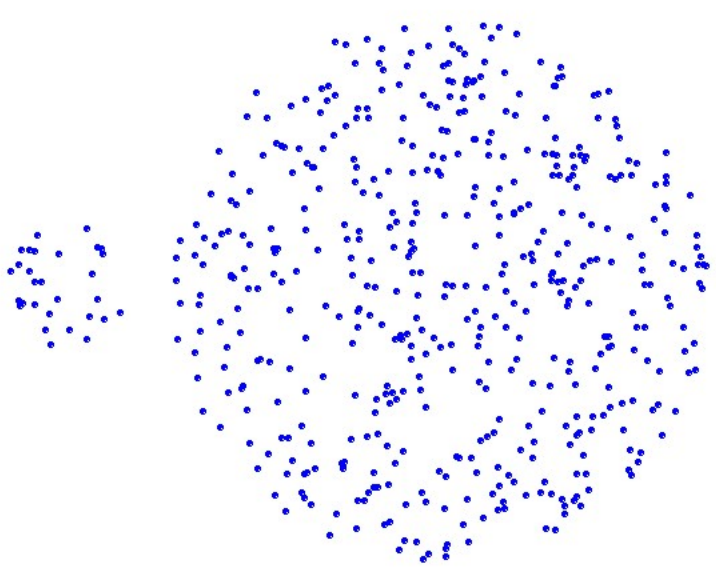
Original Points



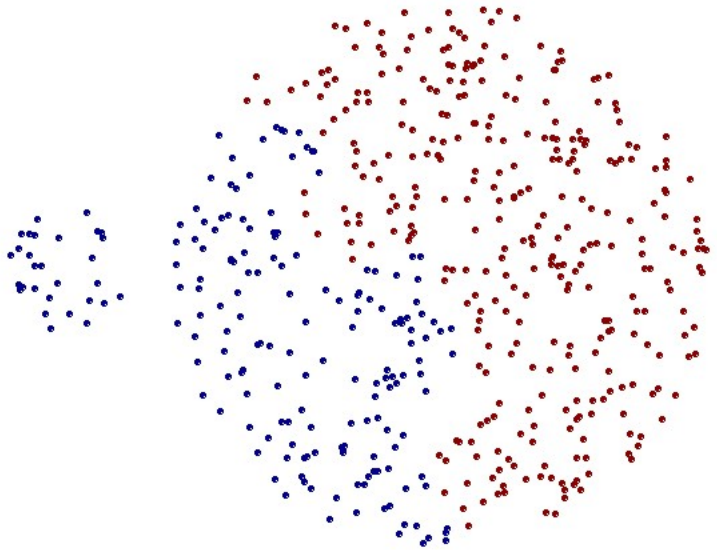
Two Clusters

- **More balanced clusters (with equal diameter)**
- **Less susceptible to noise**

Limitations of complete-link clustering



Original Points



Two Clusters

- **Tends to break large clusters**
- **All clusters tend to have the same diameter – small clusters are merged with larger ones**

Group average distance

- **Group average distance** between clusters C_i and C_j is the ***average distance*** between any object in C_i and any object in C_j

$$D_{avg}(C_i, C_j) = \frac{1}{|C_i| \times |C_j|} \sum_{x \in C_i, y \in C_j} d(x, y)$$

Average-link clustering: discussion

- Compromise between Single and Complete Link
- Strengths
  - Less susceptible to noise and outliers
- Limitations
  - Biased towards globular clusters

Distance between two clusters

- **Centroid distance** between clusters C_i and C_j is the distance between the centroid r_i of C_i and the centroid r_j of C_j

$$D_{centroids}(C_i, C_j) = d(r_i, r_j)$$

Distance between two clusters

- **Ward's distance** between clusters C_i and C_j is the ***difference*** between the ***total within cluster sum of squares for the two clusters separately***, and the ***within cluster sum of squares resulting from merging the two clusters*** in cluster C_{ij}

$$D_w(C_i, C_j) = \sum_{x \in C_i} (x - r_i)^2 + \sum_{x \in C_j} (x - r_j)^2 - \sum_{x \in C_{ij}} (x - r_{ij})^2$$

- r_i: centroid of C_i
- r_j: centroid of C_j
- r_{ij}: centroid of C_{ij}

Ward's distance for clusters

- Similar to group average and centroid distance
- Less susceptible to noise and outliers
- Biased towards globular clusters
- Hierarchical analogue of k-means
  - Can be used to initialize k-means

Master in
High Performance Computing

Modern Clustering algorithms.

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

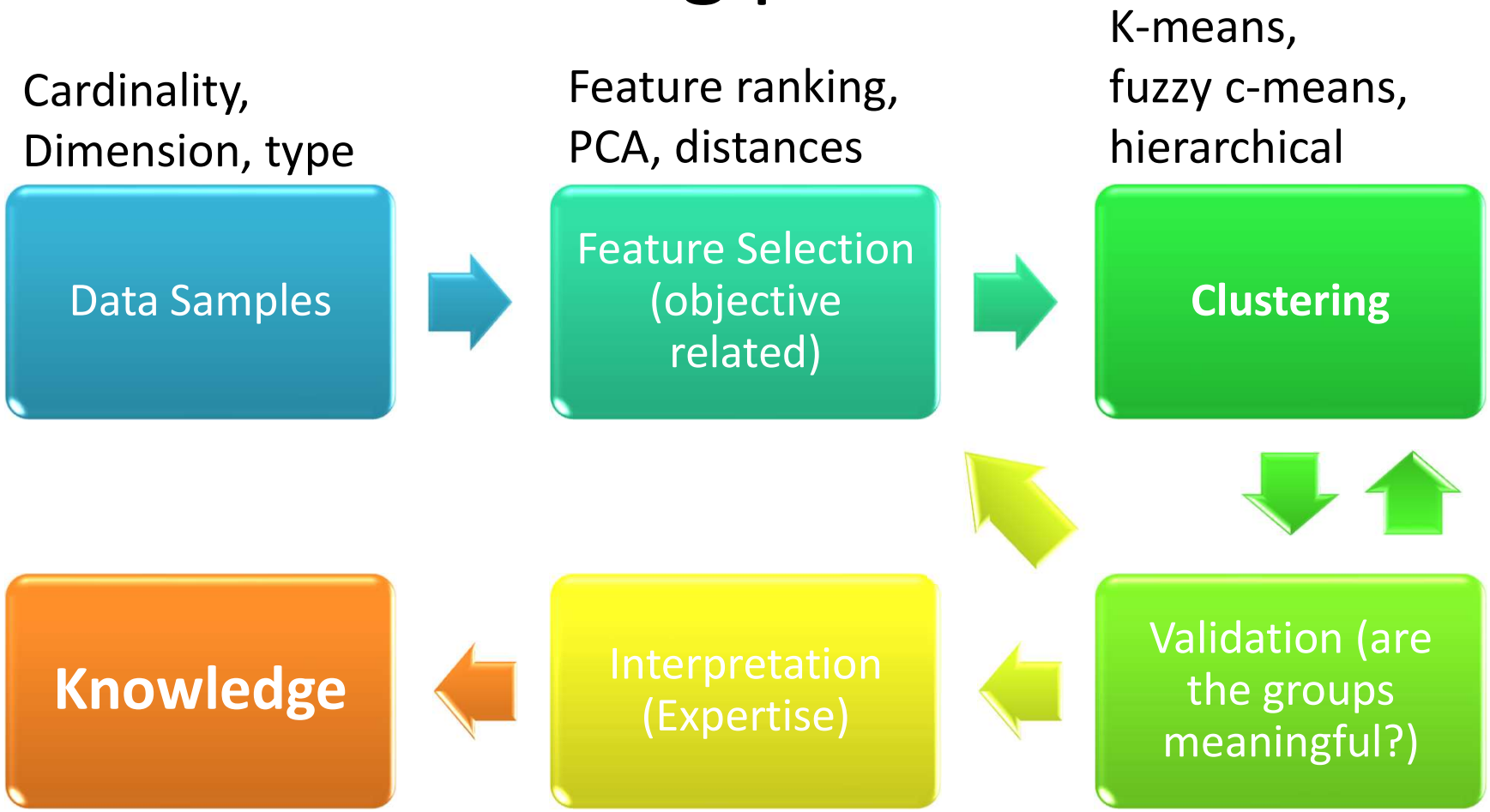
Previously at lesson one…

- **Data samples:** Characterized by the number of elements (cardinality), the number of features (dimensionality) and the type of these features.
- **Feature selection:** We employ it when a ground truth is available. The method of choice depends on the nature of the ground truth and the nature of the features.
- **Dimensionality reduction:** Does not require a ground truth. It usually simplifies the data and allows a more efficient representation or treatment. Although we focused on PCA, there are many alternatives.
- **Similarities and distances:** a pairwise measure of how similar are the elements of the data set. Are the basis for many clustering methods and they should be selected according with the nature of the dataset. More complicated distances (like the geodesic distance) are available and can be useful in difficult analyses.

Previously at lesson two…

- **Flat clustering:** Provides a hard partition of your dataset in k groups. (k-means)
- **Fuzzy clustering:** Provides a soft partition of your dataset in k groups. A soft partition implies that every point belongs to a cluster with a certain degree of membership. (fuzzy c-means)
- **Hierarchical clustering:** It can be done in an agglomerative or in a divisive way. Its output is a dendrogram: a tree-like diagram that records the sequences of merges or splits. Agglomerative cluster algorithms differ in the way they compute the inter-cluster distance (Single-link, Complete-link, Ward's method...).

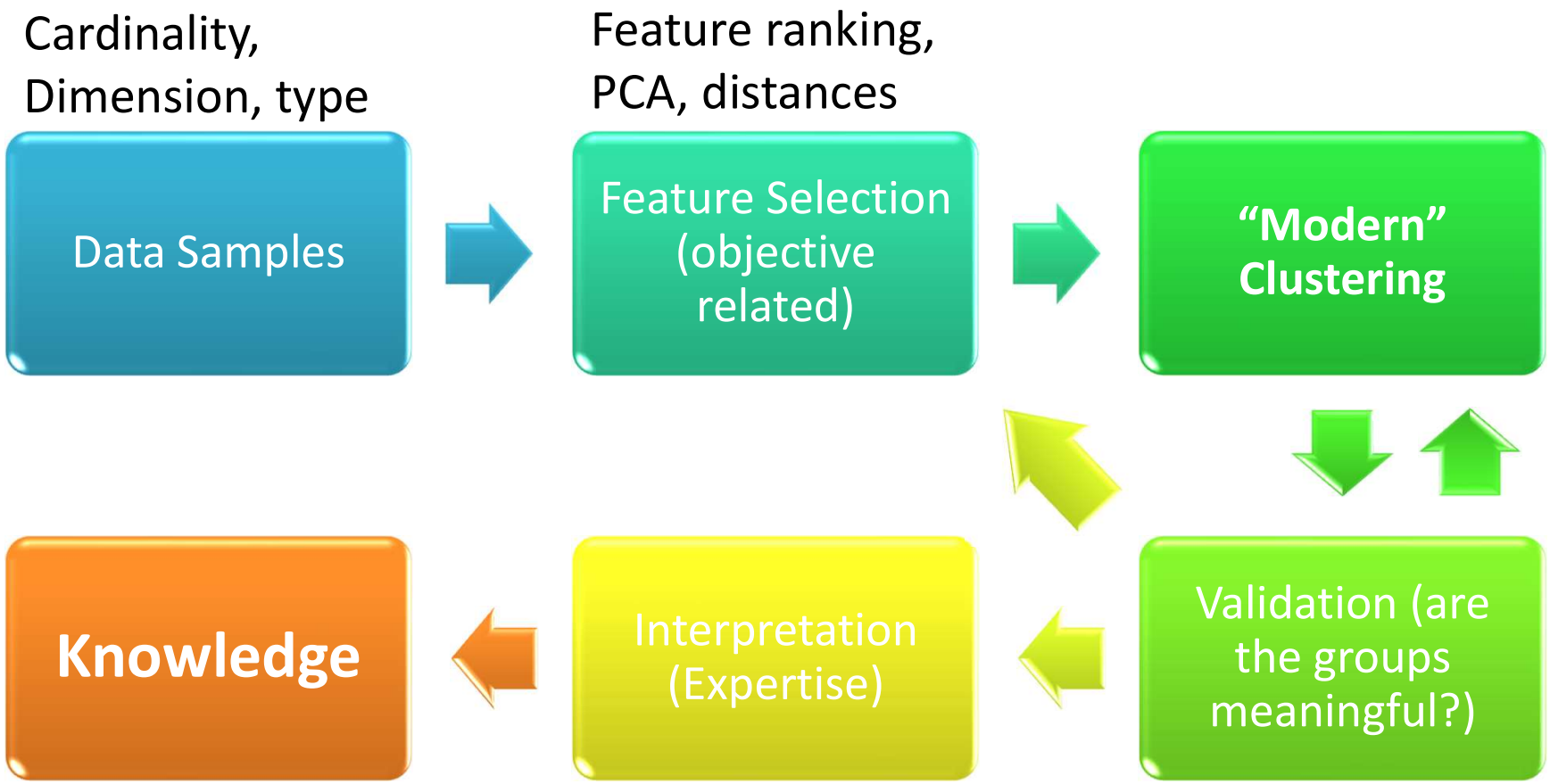
Clustering procedure



Some problems with the discussed algorithms

- K-means and fuzzy c-means are not well suited to deal with arbitrary shape (non-convex) clusters.
- Which intercluster distance shall we use for hierarchical clustering?
- What happens with noisy data sets?
- Which should be the final number of clusters?

Clustering procedure



Reviews on clustering algorithms

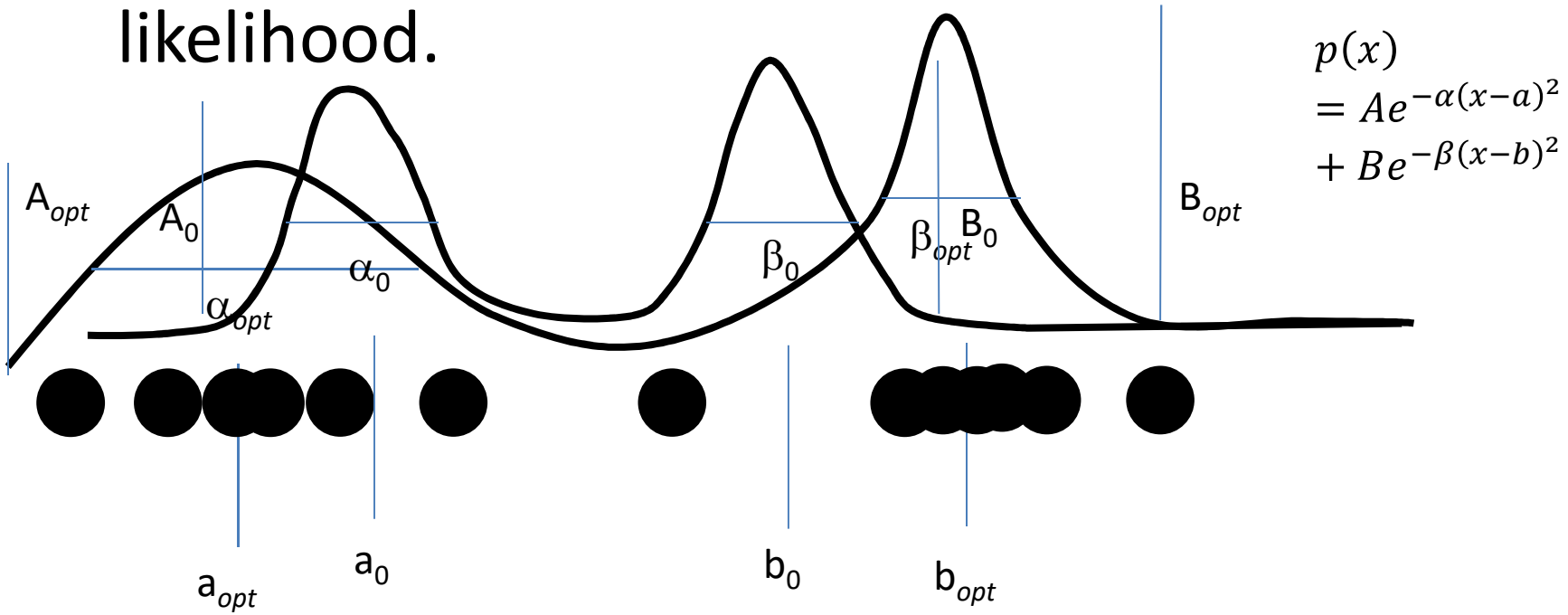
- There are hundreds of clustering algorithms, each of them with their advantages and withdrawals.
  - Jain, A. K., Murty, M. N., & Flynn, P. J. (1999). Data clustering: a review. *ACM computing surveys (CSUR)*, *31*(3), 264-323.
  - Xu, R., & Wunsch, D. (2005). Survey of clustering algorithms. *Neural Networks, IEEE Transactions on*, *16*(3), 645-678.
  - Jain, A. K. (2010). Data clustering: 50 years beyond K-means. *Pattern recognition letters*, *31*(8), 651-666.
  - Xu, D., & Tian, Y. (2015). A Comprehensive Survey of Clustering Algorithms.*Annals of Data Science*, *2*(2), 165-193.

Expectation-Maximization clustering

Modern clustering algorithms (I)

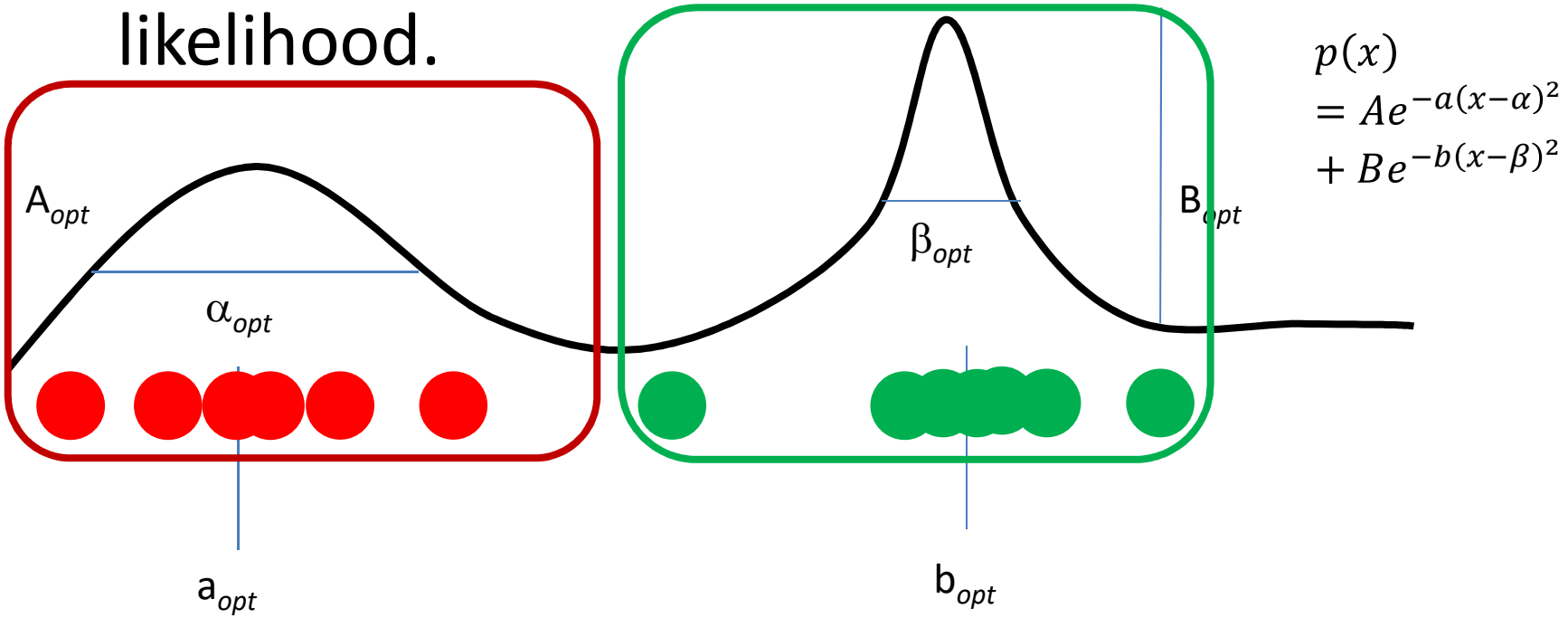
Model based clustering

- Consider your data as a set of realizations of an underlying probability function $p(x)$.
- Assume the functional form of $p(x)$ and estimate its parameters by maximum likelihood.



Model based clustering

- Consider your data as a set of realizations of an underlying probability function $p(x)$.
- Assume the functional form of $p(x)$ and estimate its parameters by maximum likelihood.



Expectation-maximization algorithm

- Iterative procedure to compute the ***Maximum Likelihood (ML)*** estimate – even in the presence of missing or hidden data
- **EM** consists of two steps:
  - **Expectation step:** the (missing) data are estimated given the observed data and current estimates of model parameters
  - **Maximization step:** The likelihood function is maximized under the assumption that the (missing) data are known

EM algorithm for mixture of Gaussians

- What is a mixture of **K** Gaussians?

$$p(x) = \sum_{k=1}^{K} \pi_k F(x \mid \Theta_k)$$

with

$$\sum_{k=1}^{K} \pi_k = 1$$

and **F(x|Θ)** is the Gaussian distribution with parameters **Θ = {μ,Σ}**

EM algorithm for mixture of Gaussians

- If all points **xϵX** are mixtures of **K** Gaussians then

$$p(X) = \prod_{i=1}^{n} p(x_i) = \prod_{i=1}^{n} \sum_{k=1}^{K} \pi_k F(x_i \mid \Theta_k)$$

- **Goal:** Find $\pi_1,\ldots, \pi_k$ and $\Theta_1,\ldots, \Theta_k$ such that **P(X)** is maximized
- Or, **ln(P(X))** is maximized:

$$L(\Theta) = \sum_{i=1}^{n} \ln\left\{\sum_{k=1}^{K} \pi_k F(x_i \mid \Theta_k)\right\}$$

Mixtures of Gaussians -- notes

- Every point $\mathbf{x_i}$ is ***probabilistically*** assigned (generated) to (by) the **k**-th Gaussian

- Probability that point $\mathbf{x_i}$ is generated by the **k**-th Gaussian is

$$w_{ik} = \frac{\pi_k F(x_i \mid \Theta_k)}{\sum_{j=1}^{K} \pi_j F(x_i \mid \Theta_j)}$$

Mixtures of Gaussians -- notes

- Every Gaussian (cluster) C_k has an effective number of points assigned to it N_k

$$N_k = \sum_{i=1}^{n} w_{ik}$$

- With mean

$$\mu_k = \frac{1}{N_k} \sum_{i=1}^{n} w_{ik} x_i$$

- And variance

$$\Sigma_k = \frac{1}{N_k} \sum_{i=1}^{n} w_{ik} \left(x_i - \mu_k\right) x_i \left(x_i - \mu_k\right)^T$$

EM for Gaussian Mixtures

- Initialize the means μ_k, variances Σ_k ($\Theta_k=(\mu_k,\Sigma_k)$) and mixing coefficients π_k, and evaluate the initial value of the loglikelihood

- **Expectation step**: Evaluate weights

$$w_{ik} = \frac{\pi_k F(x_i \mid \Theta_k)}{\sum_{j=1}^{K} \pi_j F(x_i \mid \Theta_j)}$$

EM for Gaussian Mixtures

- **Maximization step**: Re-evaluate parameters

$$\mu_k^{new} = \frac{1}{N_k} \sum_{i=1}^{n} w_{ik} x_i$$

$$\Sigma_k^{new} = \frac{1}{N_k} \sum_{i=1}^{n} w_{ik} \left(x_i - \mu_k^{new} \right) x_i \left(x_i - \mu_k^{new} \right)^T$$

$$\pi_k^{new} = \frac{N_k}{N}$$

- Evaluate $L(\Theta^{new})$ and stop if converged

EM characteristics

- Notice the similarity between EM for Normal mixtures and K-means: The expectation step is the assignment, while the maximization step is the update of the centers.
- If the model (k, functional form) is not realistic, neither will the results.
- It is not guaranteed to reach the global optimum.

Density-Based Spatial Clustering of Applications with Noise (DBSCAN)

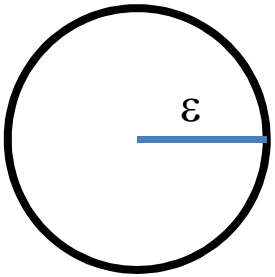
Modern clustering algorithms (II)

DBSCAN

Density-based Clustering locates regions of high density that are separated from one another by regions of low density.

- Density = number of points within a specified radius (Eps)

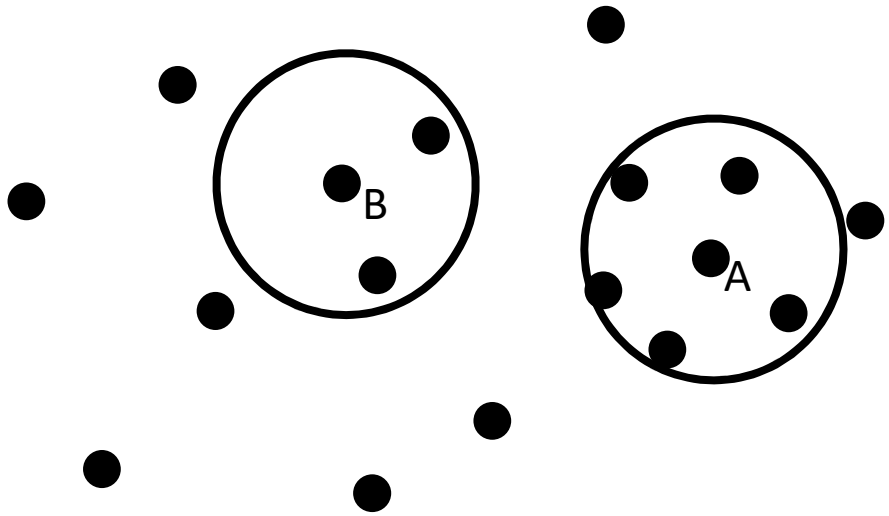
A route for this explanation



ε / Eps is a method parameter.

$\rho_A = 6$

$\rho_B = 3$



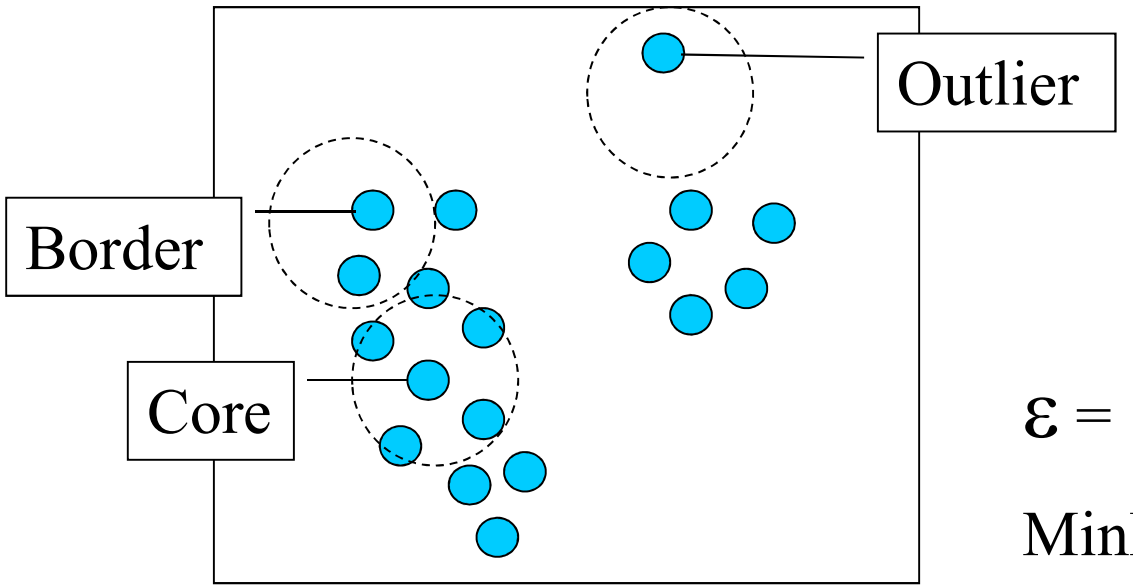
- We will classify the points according with their density.
- We will define a set of rules that exploit this classification in order to define the clusters.

DBSCAN: Point classes

- A point is a **core point** if it has more than a specified number of points (MinPts) within Eps. These are points that are at the interior of a cluster
- A **border point** has fewer than MinPts within Eps, but is in the neighborhood of a core point
- A **noise point** is any point that is not a core point or a border point.

Note that we are adding a new method parameter: MinPts

Border & Core

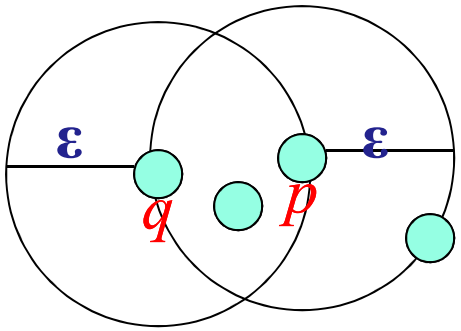


DBSCAN: Ideas behind the rules

- Any two core points are close enough– within a distance *Eps* of one another – are put in the same cluster
- Any border point that is close enough to a core point is put in the same cluster as the core point
- Noise points are discarded

Concepts: ε-Neighborhood

- **ε-Neighborhood** - Objects within a radius of ε from an object. (epsilon-neighborhood)
- **Core objects** - ε-Neighborhood of an object contains at least **MinPts** of objects



ε-Neighborhood of p

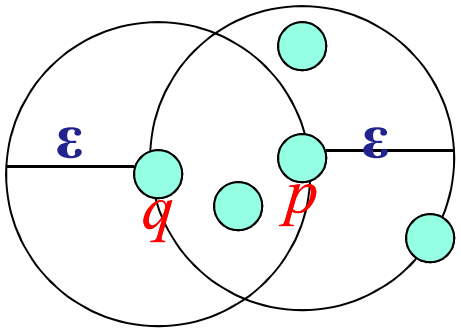
ε-Neighborhood of q

p is a core object (MinPts = 4)

q is not a core object

Concepts: Reachability

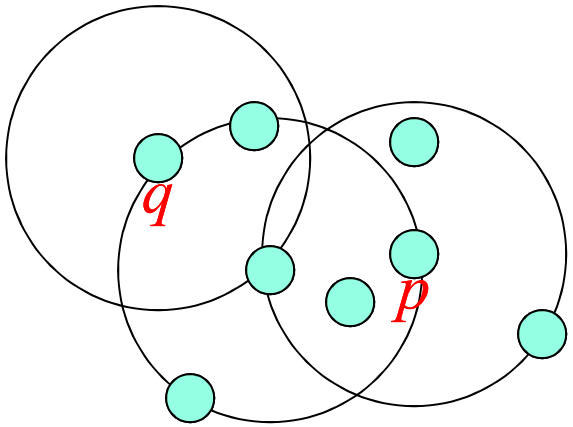
- **Directly density-reachable**
  - An object q is directly density-reachable from object p if q is within the ε-Neighborhood of p and p is a core object.



- q is directly density-reachable from p
- p is not directly density-reachable from q?

Concepts: Reachability

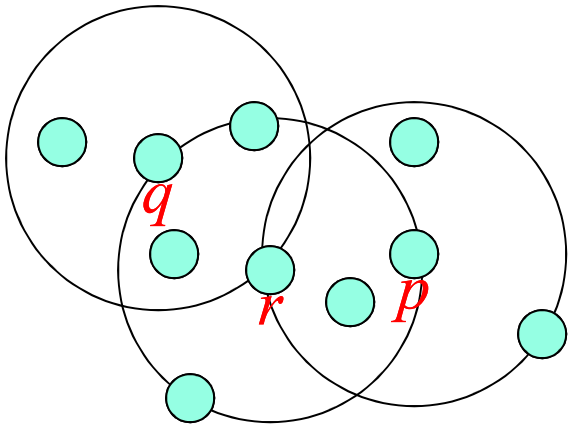
- **Density-reachable:**
  - An object *p* is density-reachable from *q* w.r.t ε and *MinPts* if there is a chain of objects $p_1,\ldots,p_n$, with $p_1=q$, $p_n=p$ such that $p_{i+1}$is directly density-reachable from p_i w.r.t ε and *MinPts* for all 1 <= *i* <= *n*



- *q* is density-reachable from *p*
- *p* is not density- reachable from *q*?
- Transitive closure of direct density-Reachability, asymmetric

Concepts: Connectivity

- **Density-connectivity**
  - Object *p* is density-connected to object *q* w.r.t ε and *MinPts* if there is an object *o* such that both *p* and *q* are density-reachable from *o* w.r.t ε and *MinPts*

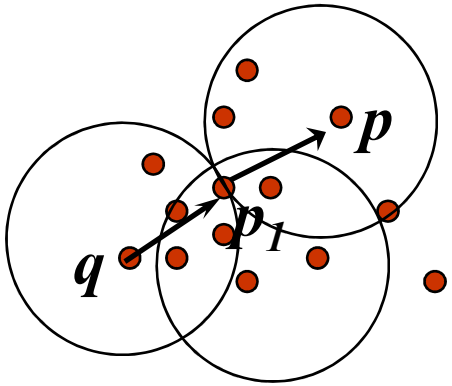


- *P* and *q* are density-connected to each other by *r*
- Density-connectivity is symmetric

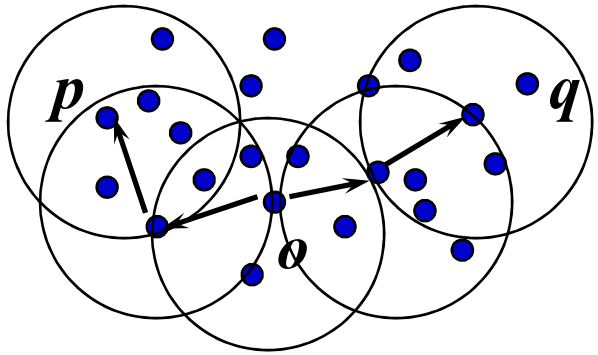
Concepts: cluster & noise

- **Cluster**: a cluster ***C*** in a set of objects ***D*** w.r.t ε and *MinPts* is a non empty subset of ***D*** satisfying
  - Maximality: For all *p*, *q* if $p \in \boldsymbol{C}$ and if *q* is density-reachable from *p* w.r.t ε and *MinPts*, then also $q \in \boldsymbol{C}$.
  - Connectivity: for all *p*, *q* $\in \boldsymbol{C}$, *p* is density-connected to *q* w.r.t ε and *MinPts* in ***D.***
  - ***Note:*** cluster contains *core objects* as well as *border objects*
- **Noise:** objects which are not directly density-reachable from at least one core object.

(Indirectly) Density-reachable:



Density-connected

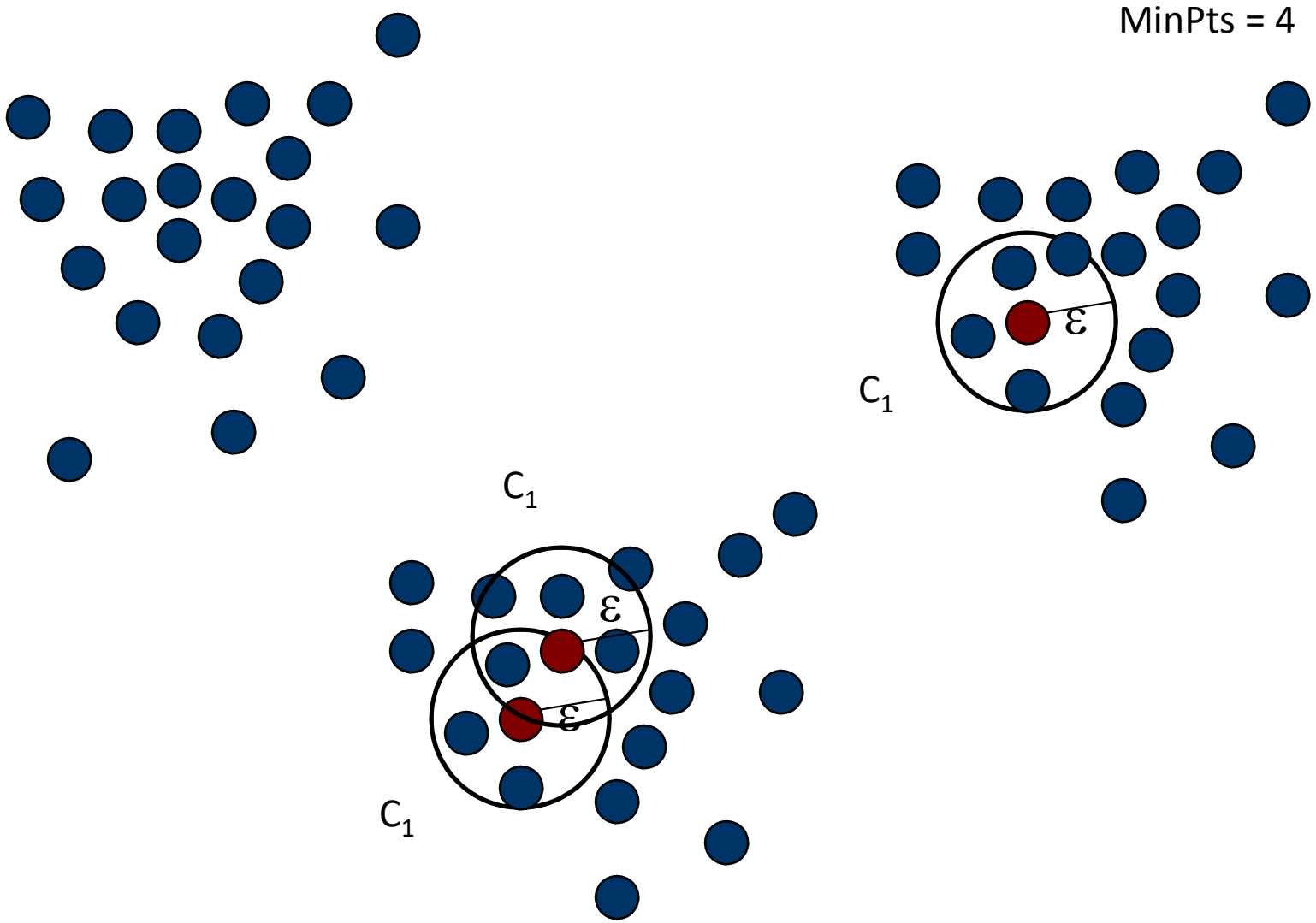


DBSCAN: The Algorithm

- select a point ***p***
- Retrieve all points density-reachable from ***p*** wrt ε and ***MinPts***.
- If ***p*** is a core point, a cluster is formed.
- If ***p*** is a border point, no points are density-reachable from ***p*** and DBSCAN visits the next point of the database.
- Continue the process until all of the points have been processed.

Result is independent of the order of processing the points

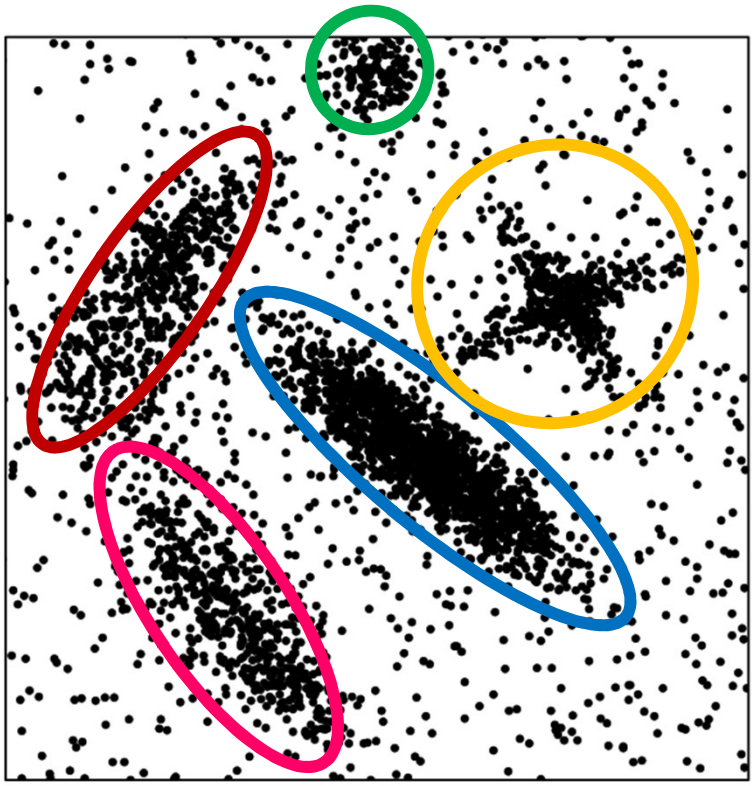
An Example

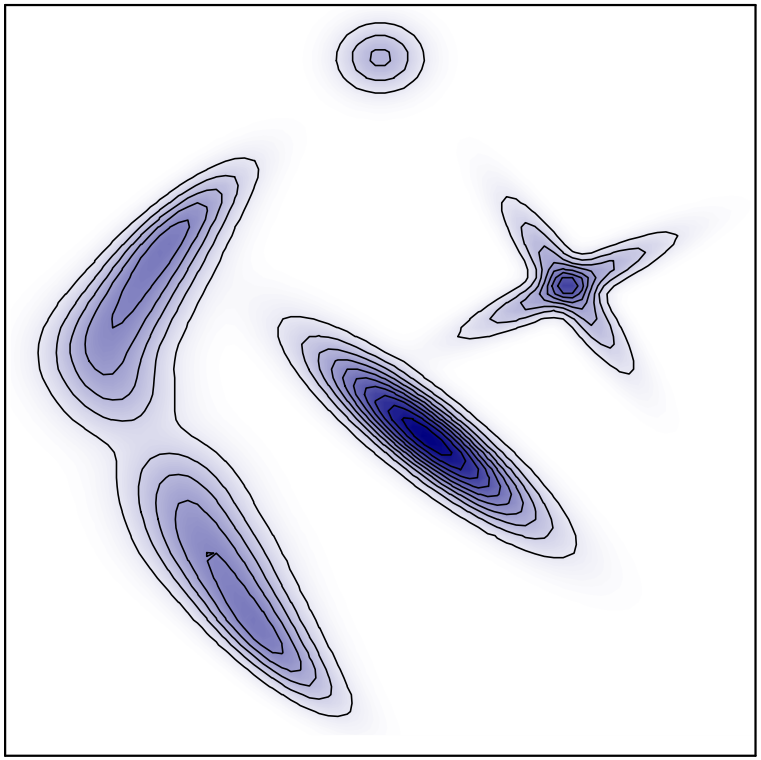


Fast Search and Find of Density Peaks

Modern clustering algorithms (III)

Density Peaks clustering





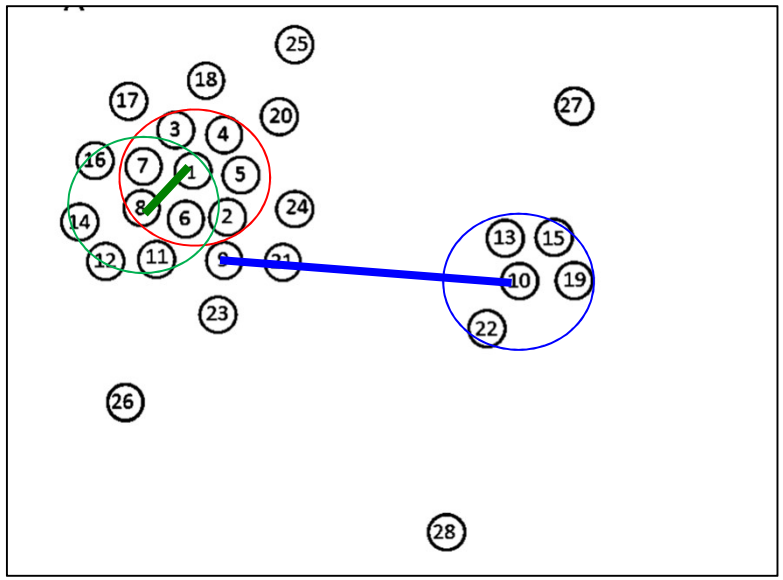
Clusters = peaks in the density of points*

= peaks in the “mother” probability distribution

*Idea already present in DB-SCAN

Density Peaks: δ concept

Clustering in a 2-dimensional space



1) Compute the local density (ρ) around each point

$\rho(1)=7$

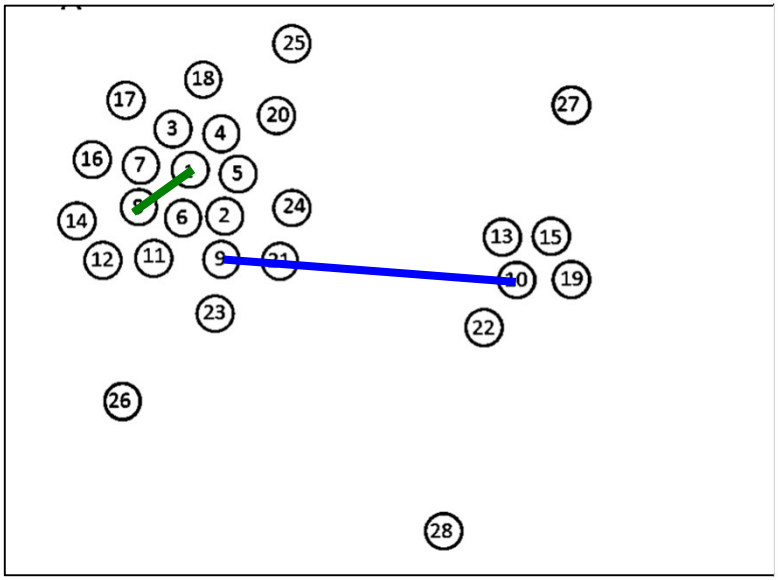
$\rho(8)=5$

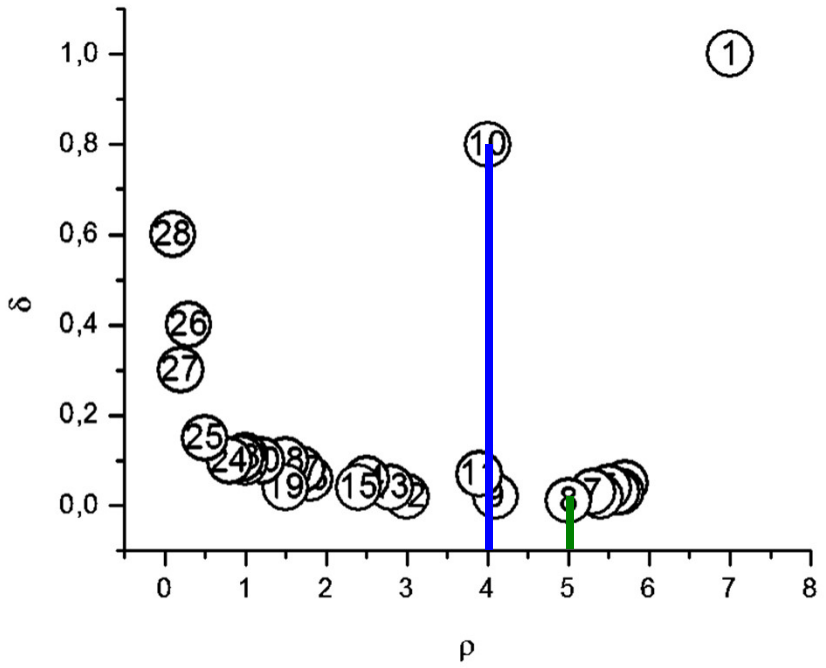
$\rho(10)=4$

2) For each point compute the distance from all the points with higher density. Take the minimum value.

Density Peaks decision graph

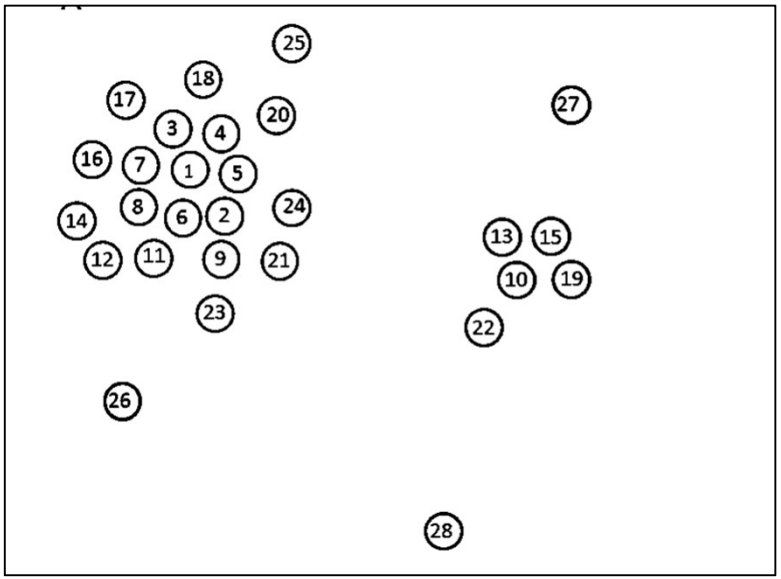
3) For each point, plot the minimum distance as a function of the density.

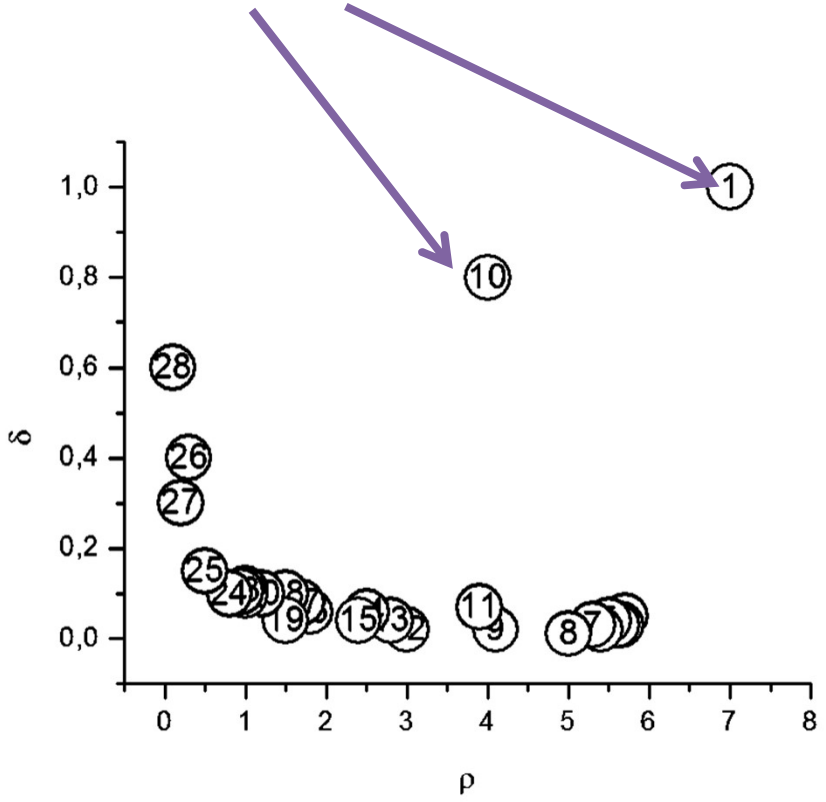




Density Peaks decision graph

4) the “outliers” in this graph are the cluster centers

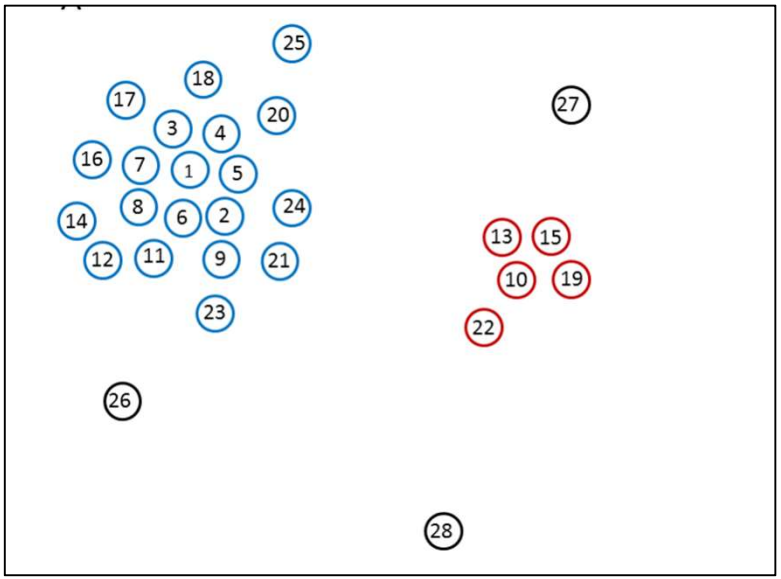


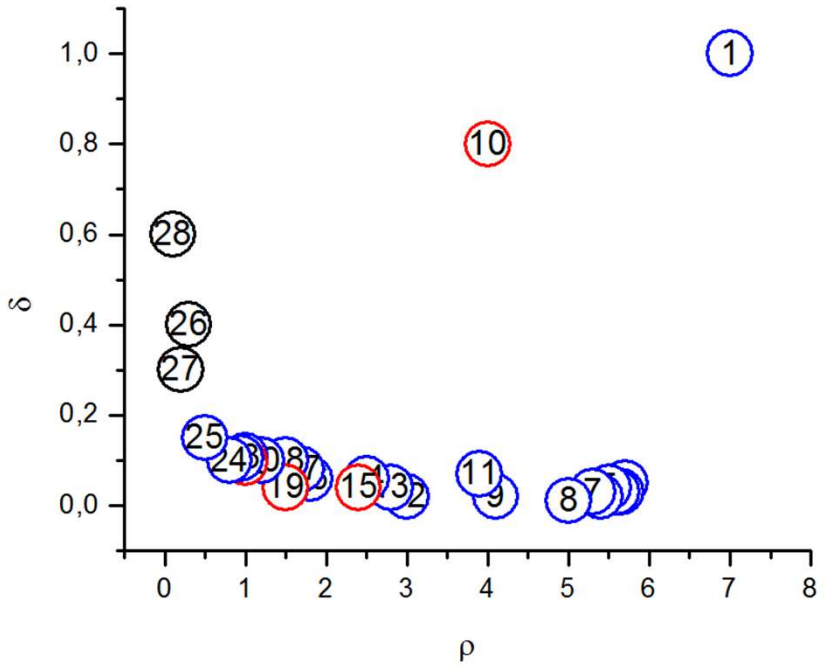


Density Peaks decision graph

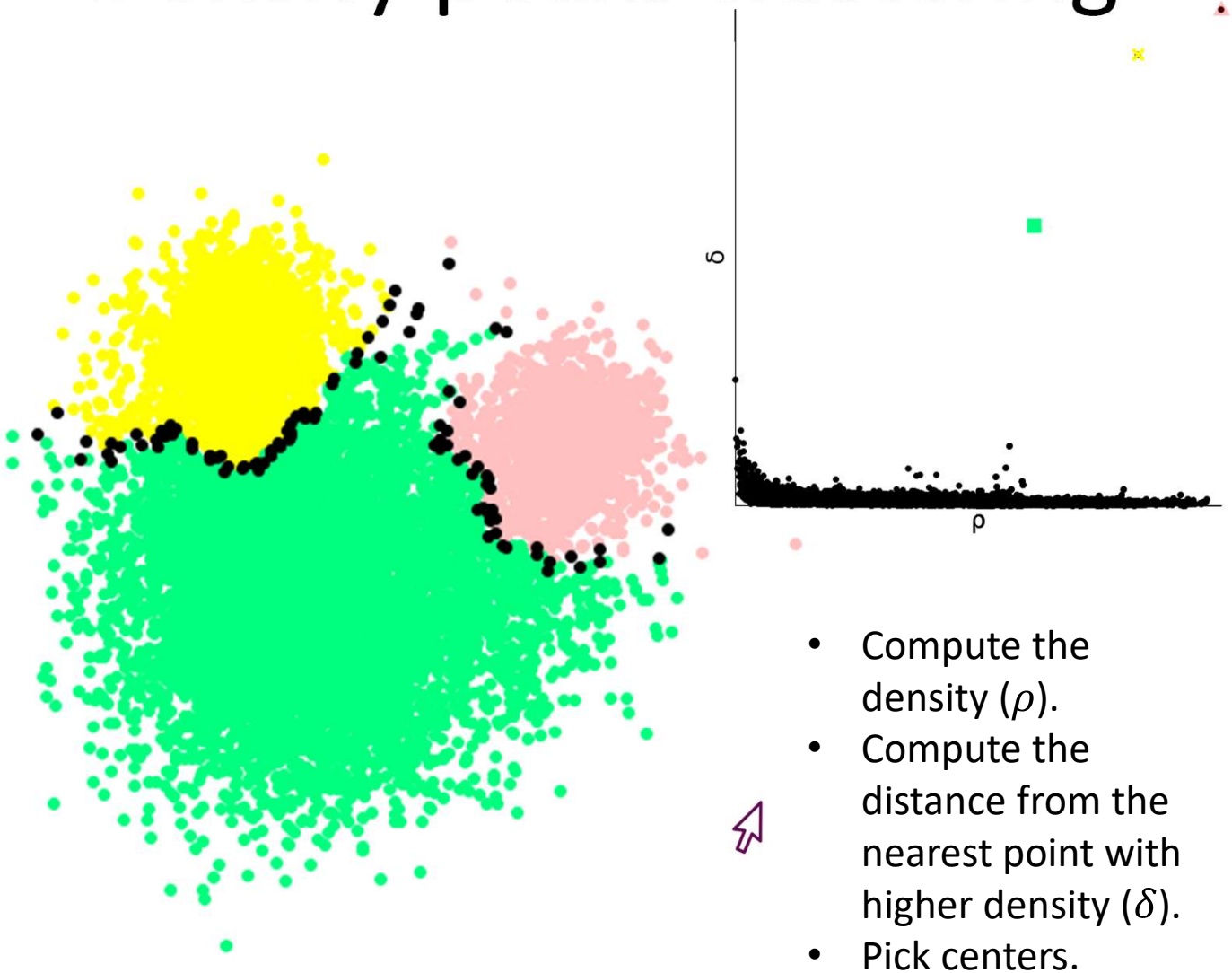
4)) the “outliers” in this graph are the cluster centers

5) Assign each point to the same cluster of its nearest neighbor of higher density





Density peaks clustering



- Compute the density (ρ).
- Compute the distance from the nearest point with higher density (δ).
- Pick centers.
- Assign points.

Density Peaks algorithm

Given a distance matrix d_{ij}, for each data point i compute:

$$\rho_i = \sum_j \chi(d_{ij} - d_c) \sim \sum_j e^{-\left(d_{ij}/d_c\right)^2}$$

(number of data points within a distance d_c or density estimation)

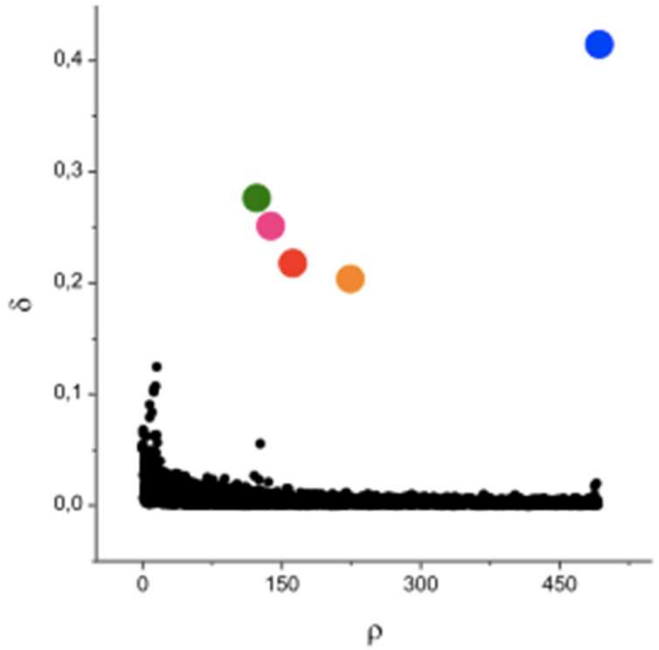
$$\delta_i = \min_{\rho_i < \rho_j} \left(d_{ij}\right)$$

(distance of the closest data point of higher density)

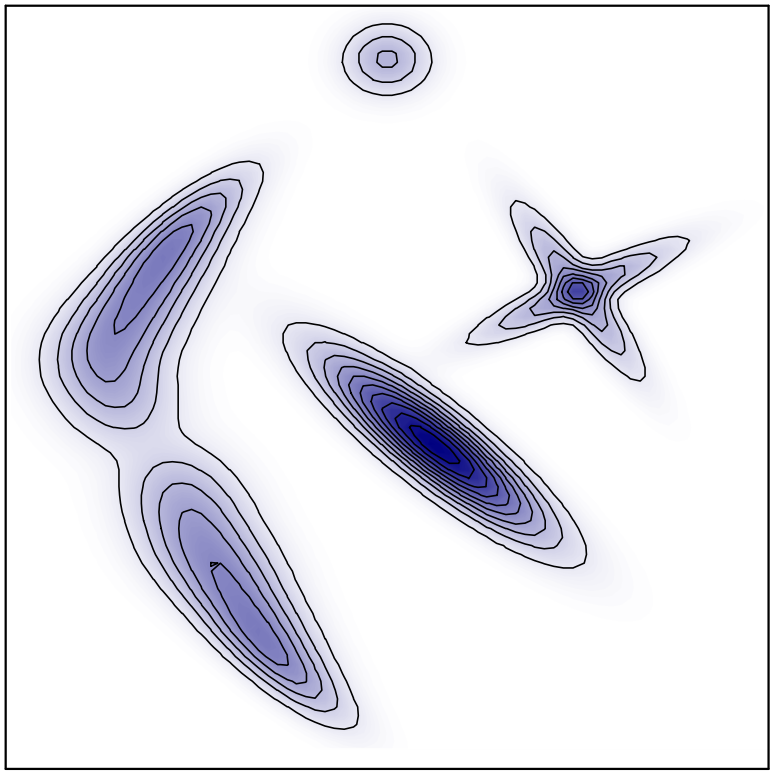
Make the Decision graph:
Plot δ as a function of ρ

One free parameter: the cutoff distance d_c

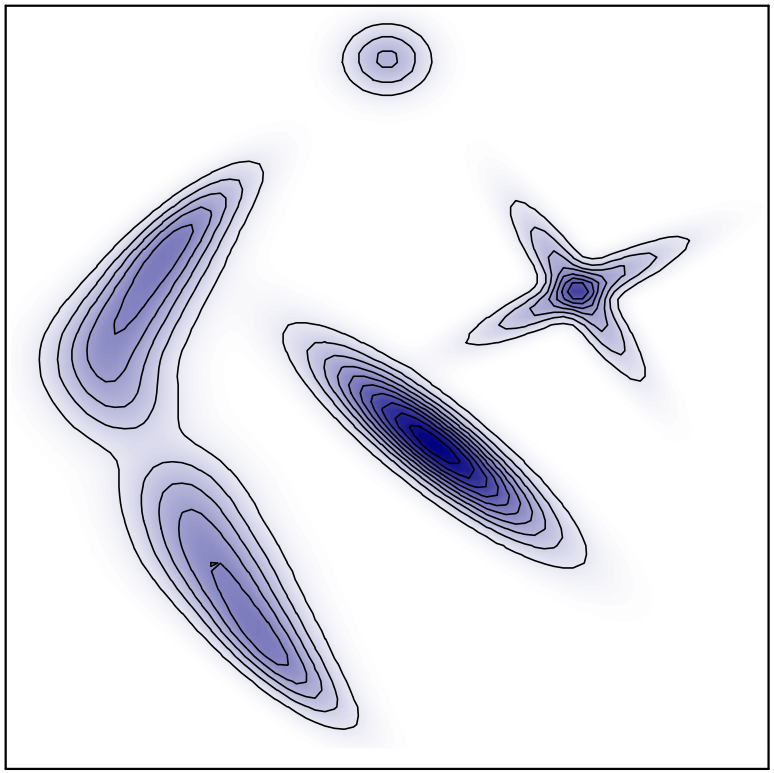
However: the method is only sensitive to the relative density of two data points, not to the absolute value of ρ

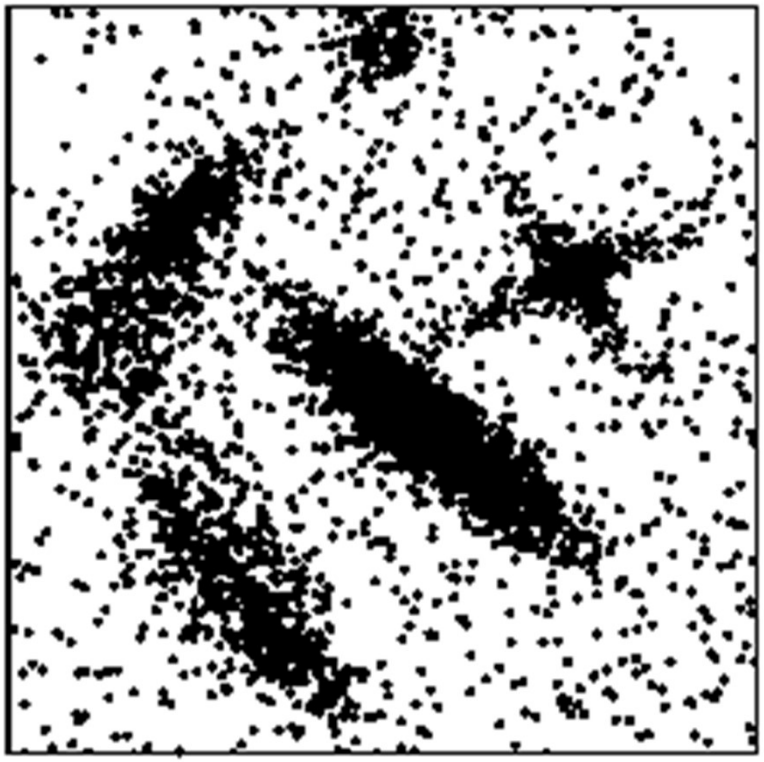


Density Peaks clustering example

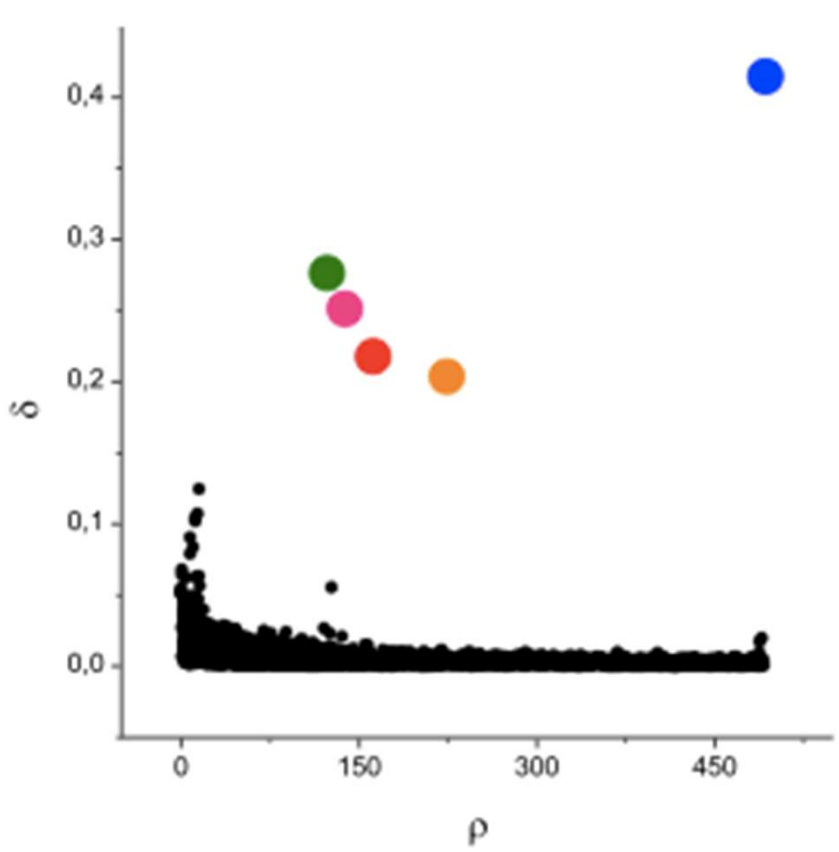


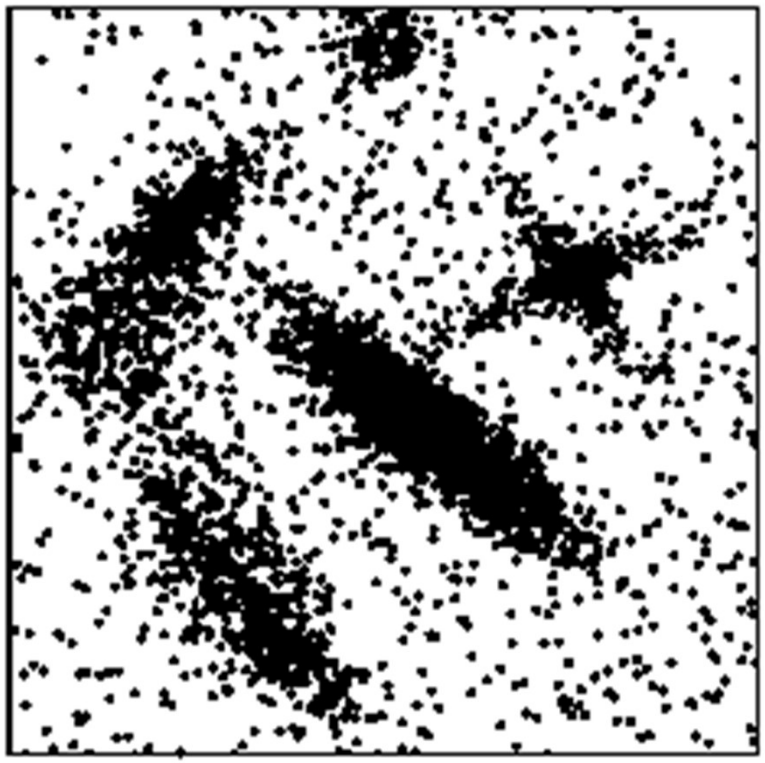
Density Peaks clustering example



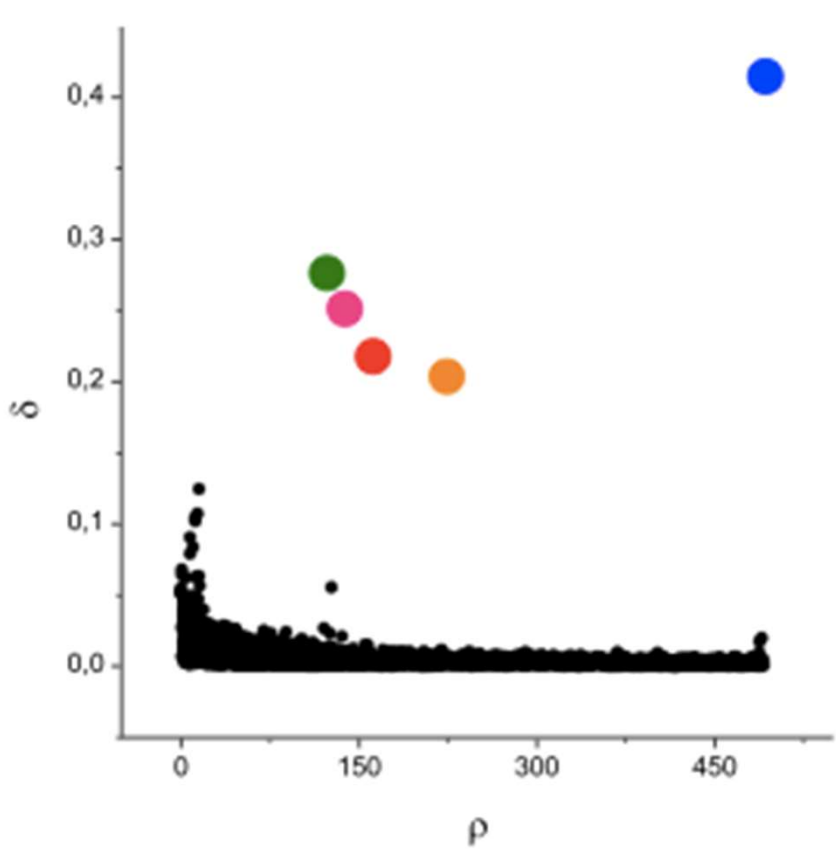


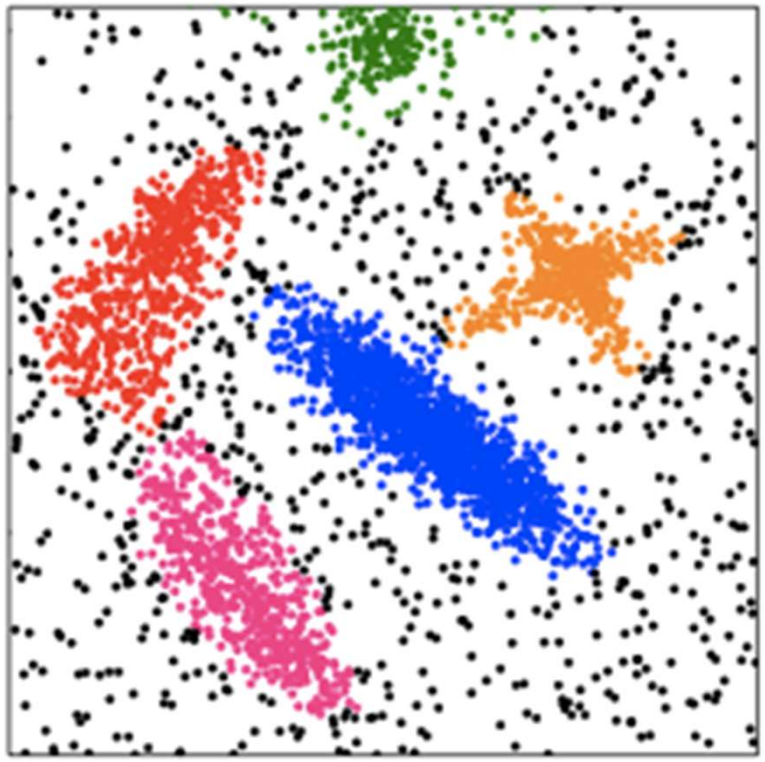
Density Peaks clustering example



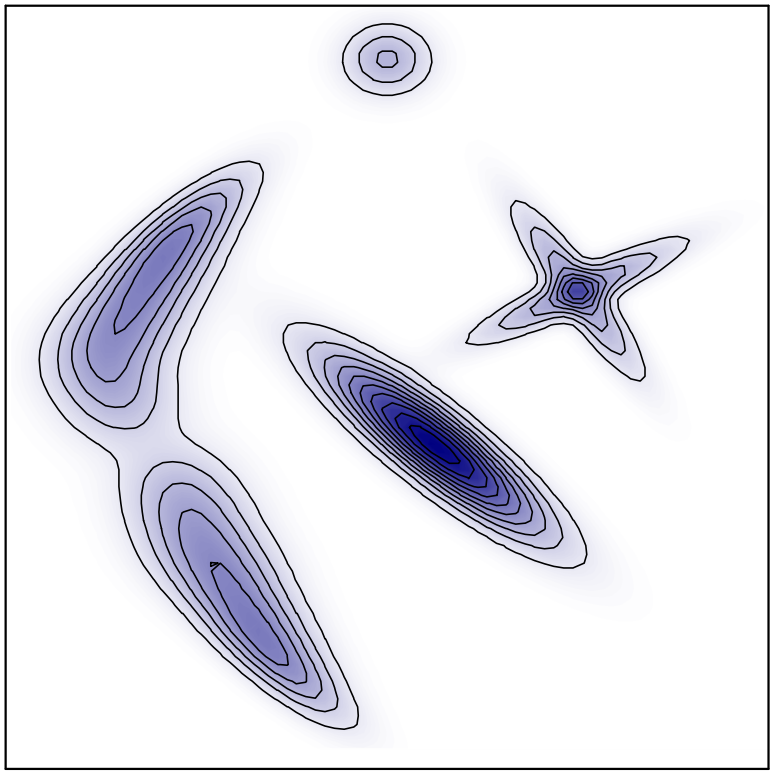


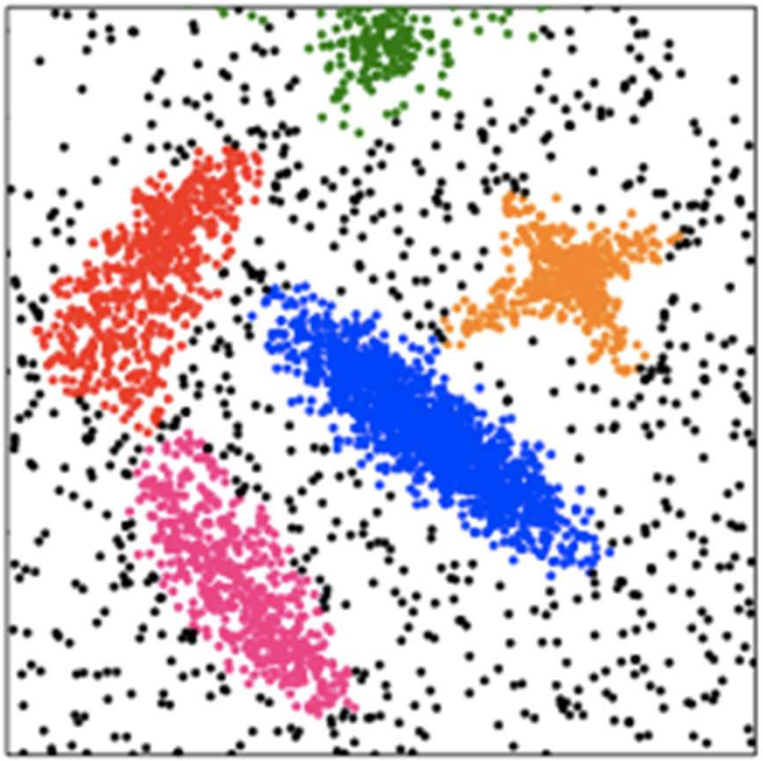
Density Peaks clustering example





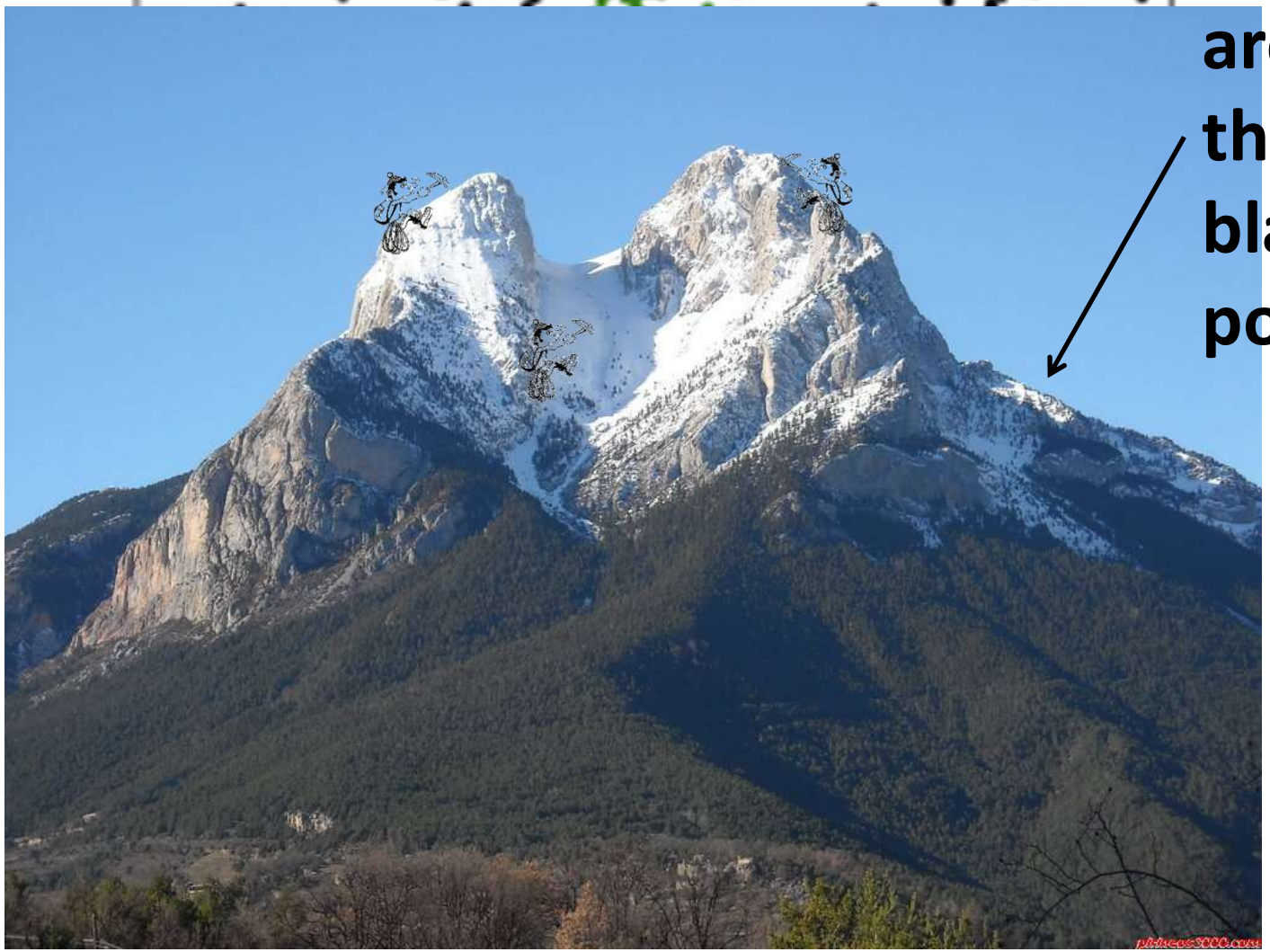
Density Peaks clustering example





Density Peaks: Halo

Halo



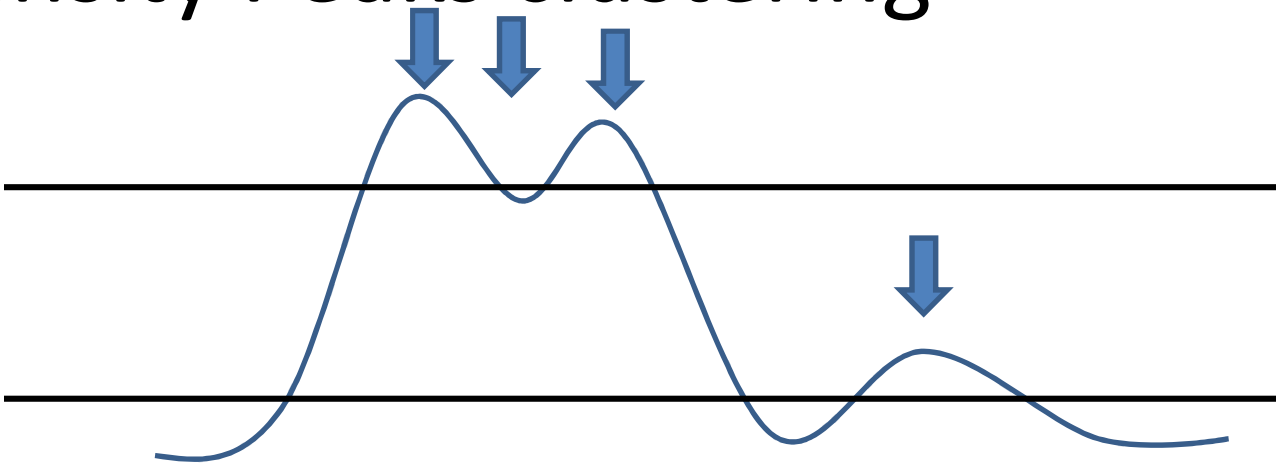
What are these black points?

Border definition

- A point i of cluster A is border point if it has, within d_c, a point j belonging to another cluster.
- The border density of A is, then, the higher density among all the border points belonging to A.
- The border point definition is strongly dependent on d_c, but it can be easily correct if we add the condition that j does not have any nearest point belonging to A.

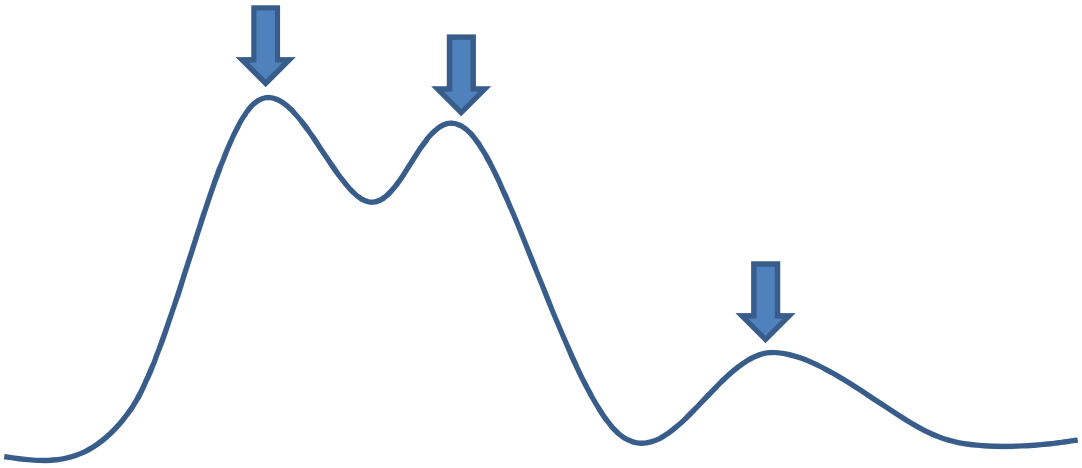
Differences between DBSCAN and Density Peaks clustering

DBSCAN



Threshold in the density: minPts/ε

Density Peaks



Directly find the peaks (But depends on the decision graph)

Some ideas about Cluster Validation

Master in High Performance Computing

Cluster validation

Supervised classification:

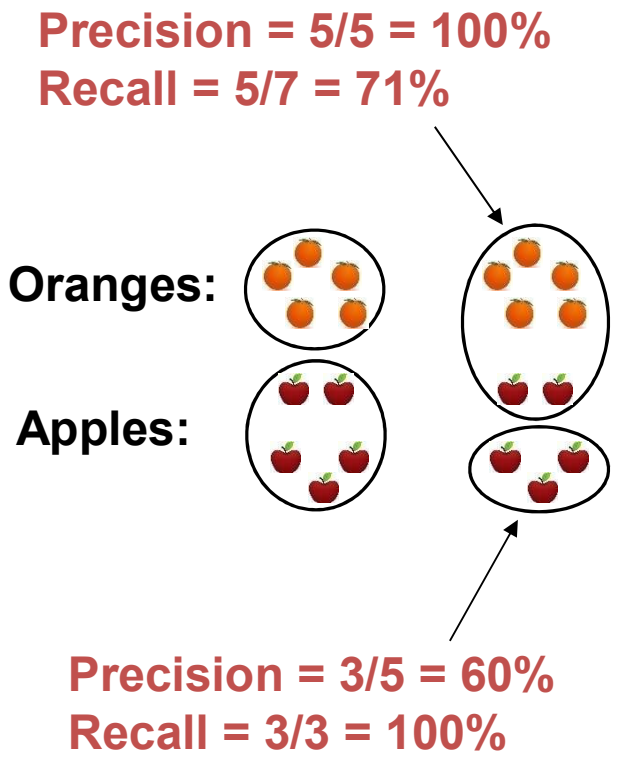
- Class labels known for ground truth
- Accuracy, precision, recall

Cluster analysis

- No class labels

Validation need to:

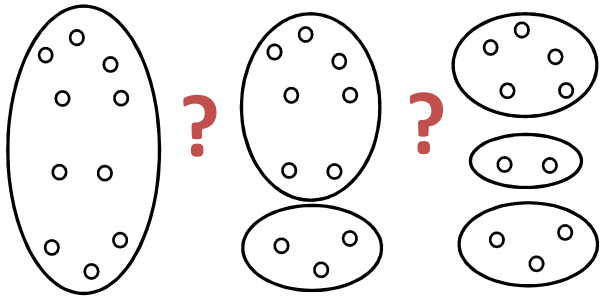
- Compare clustering algorithms
- Solve number of clusters
- Avoid finding patterns in noise



Measuring clustering validity

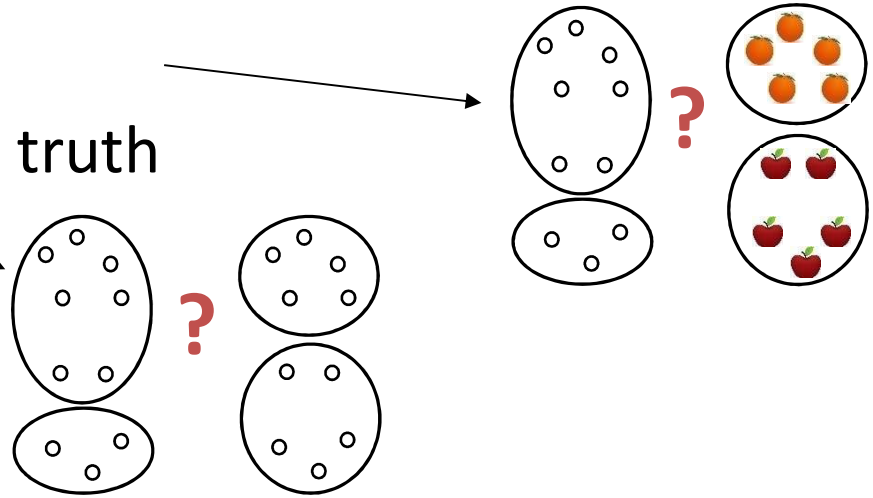
Internal Index:

- Validate *without* external info
- With different number of clusters
- Solve the number of clusters

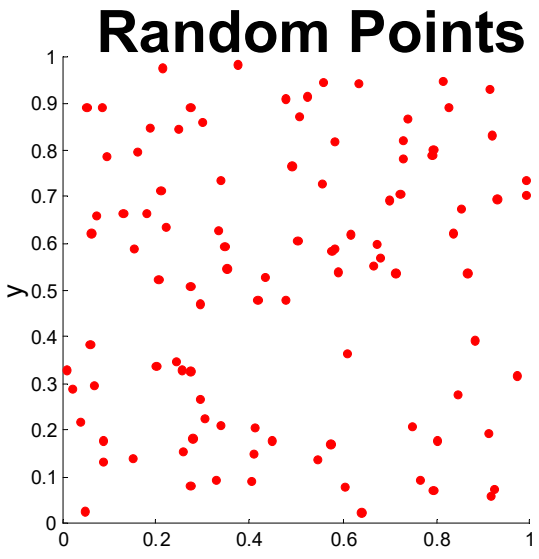


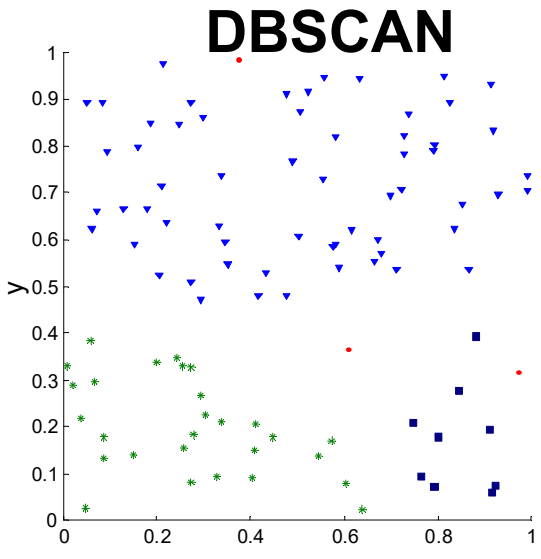
External Index

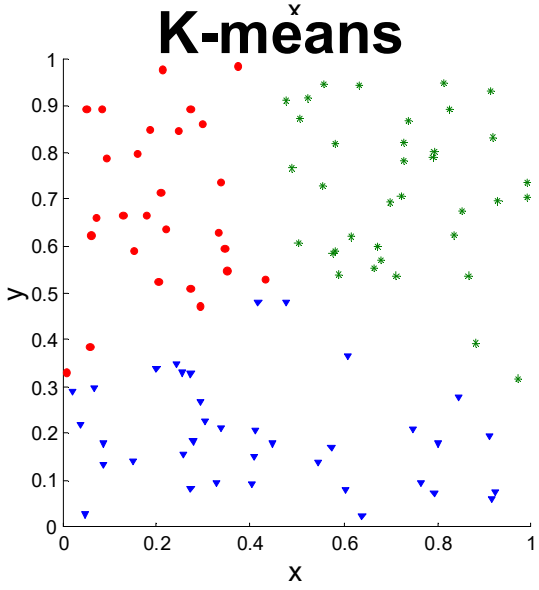
- Validate against ground truth
- Compare two clusters: (how similar)

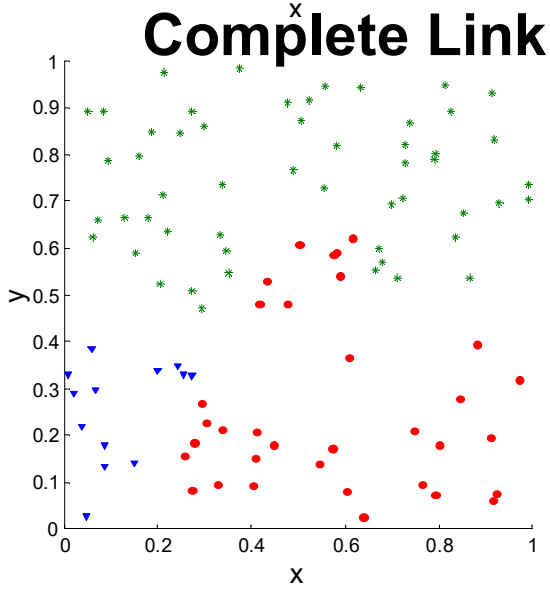


Clustering of random data







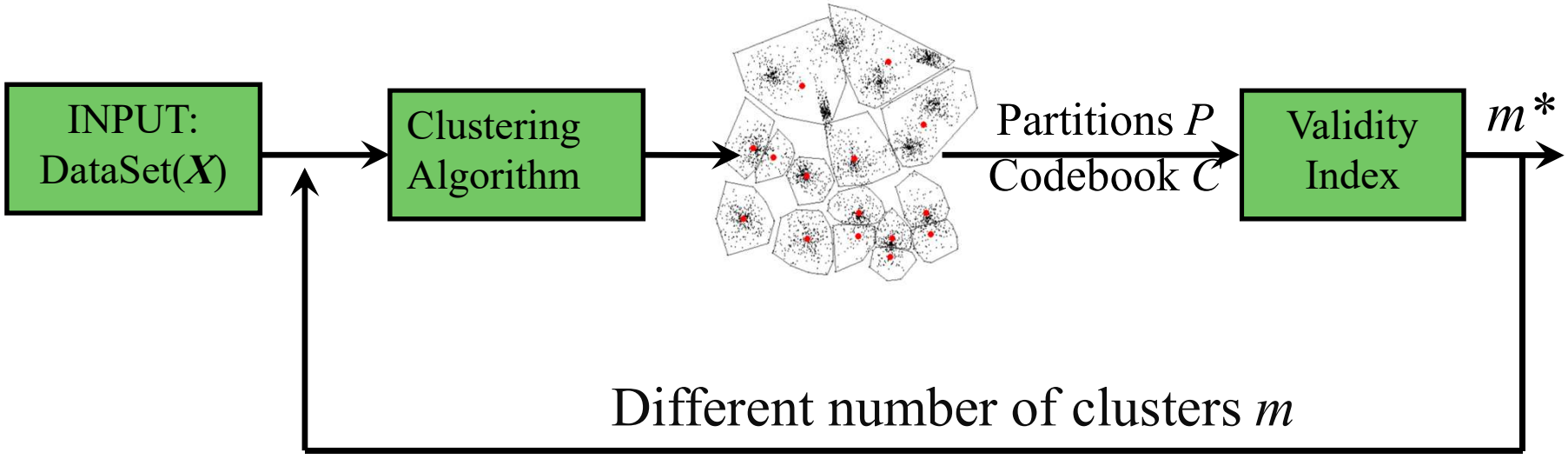


Cluster validation process

1. Distinguishing whether non-random structure actually exists in the data (one cluster).
2. Comparing the results of a cluster analysis to externally known results, e.g., to externally given class labels.
3. Evaluating how well the results of a cluster analysis fit the data *without* reference to external information.
4. Comparing the results of two different sets of cluster analyses to determine which is better.
5. Determining the number of clusters.

Cluster validation process

- **Cluster validation** refers to procedures that evaluate the results of clustering in a **quantitative** and **objective** fashion. [Jain & Dubes, 1988]
  - How to be "quantitative": To employ the measures.
  - How to be "objective": To validate the measures!

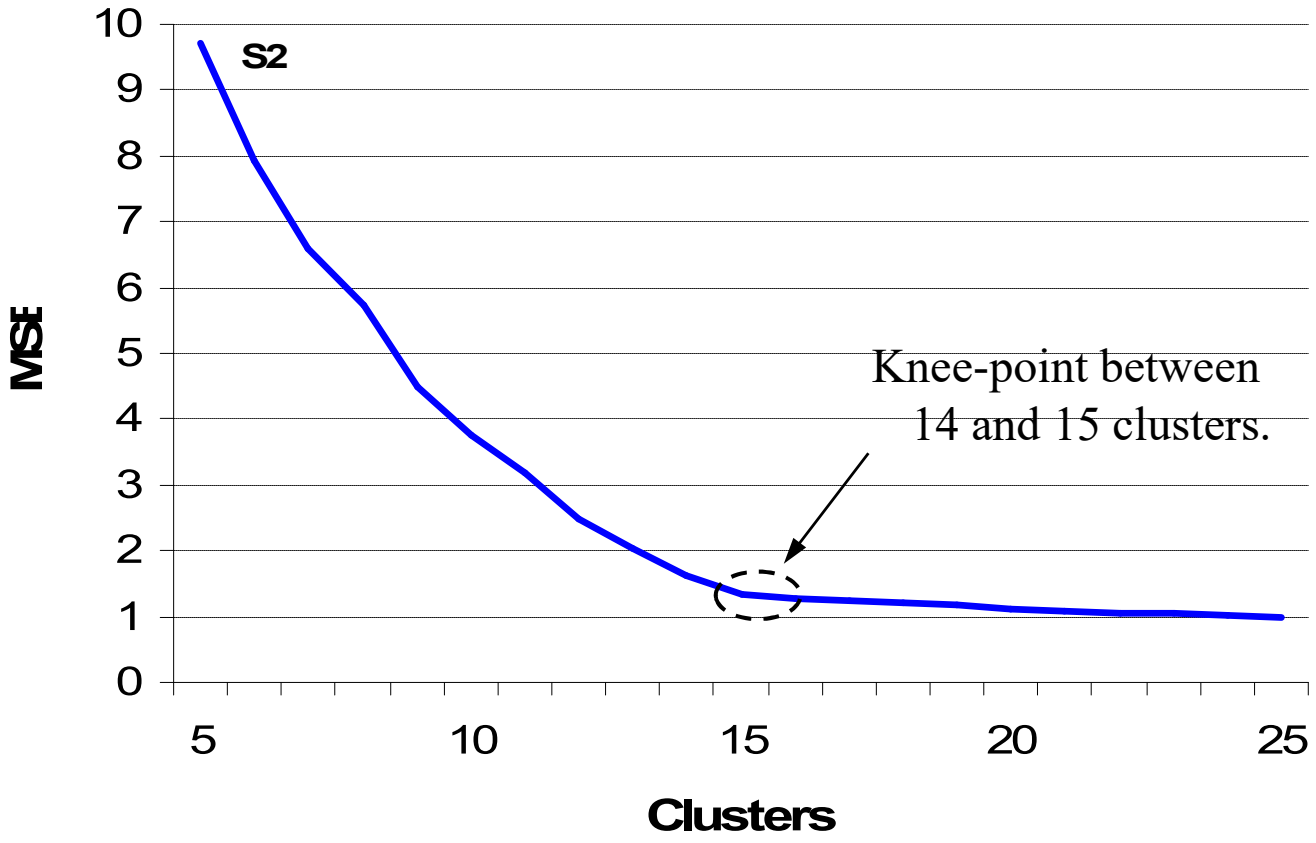


Internal indexes

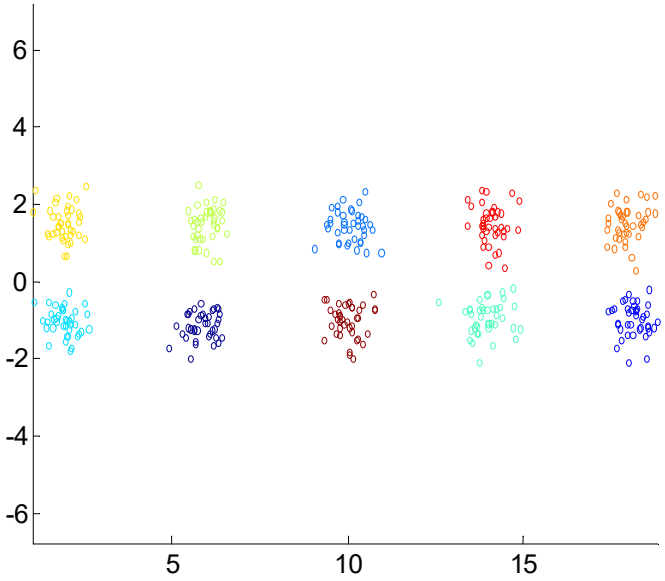
- Ground truth is rarely available but unsupervised validation must be done.
- Minimizes (or maximizes) internal index:
  - Variances of within cluster and between clusters
  - Rate-distortion method
  - F-ratio
  - Davies-Bouldin index (DBI)
  - Bayesian Information Criterion (BIC)
  - Silhouette Coefficient
  - Minimum description principle (MDL)
  - Stochastic complexity (SC)

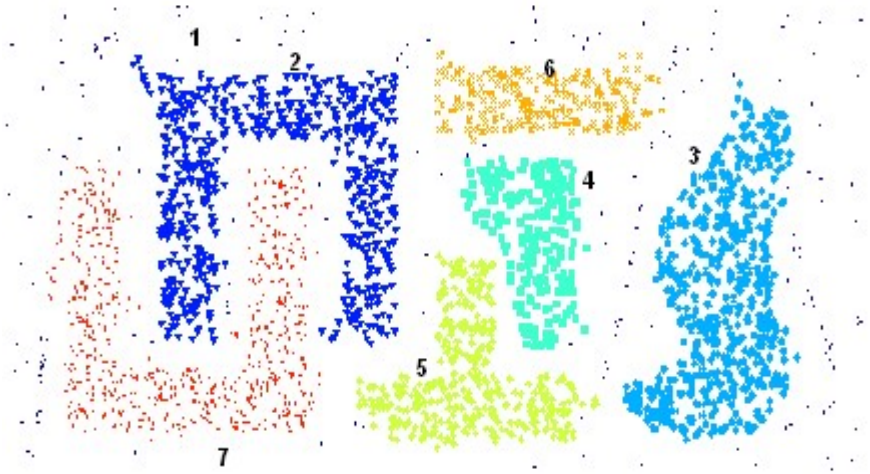
Mean square error (MSE)

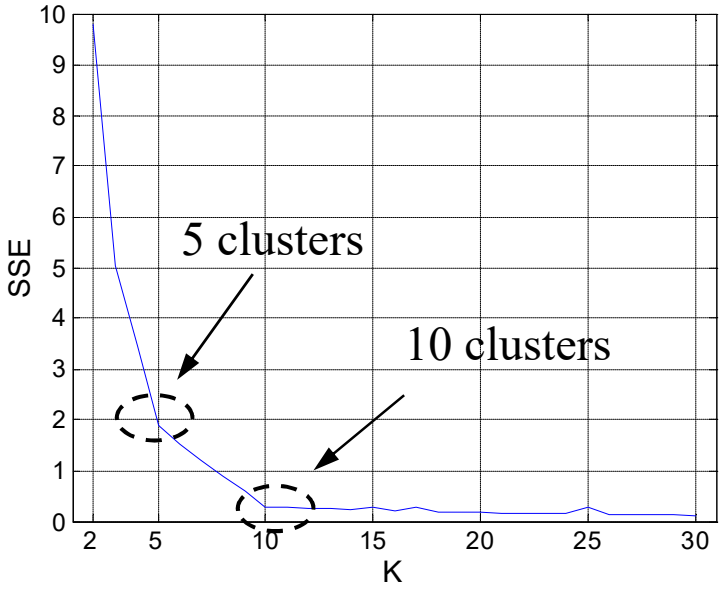
- The more clusters the smaller the MSE.
- Small knee-point near the correct value.
- But how to detect?

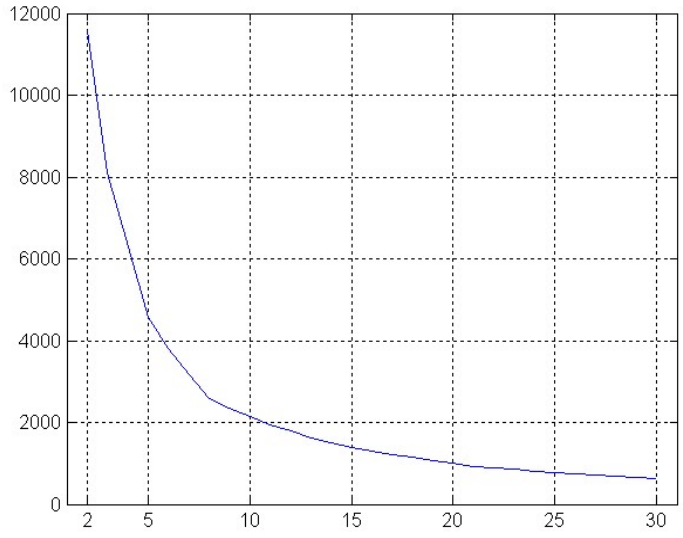


Mean square error (MSE)







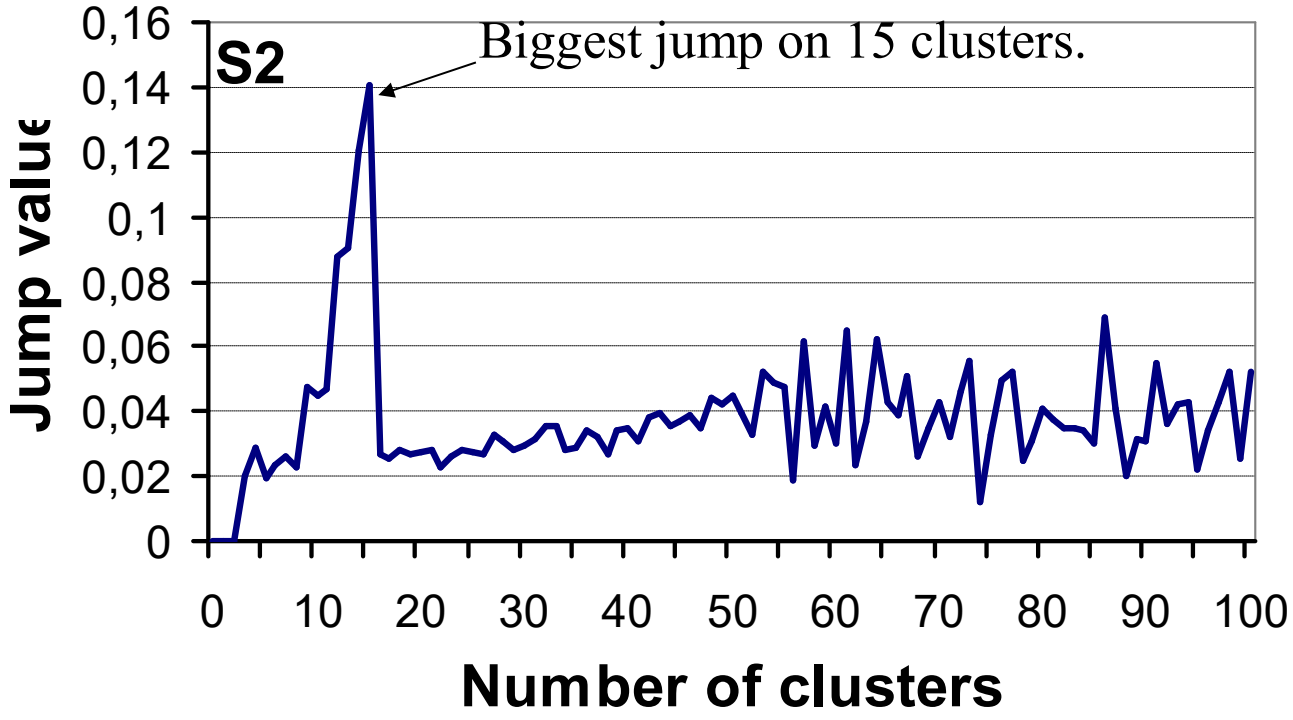


Jump point of MSE

(rate-distortion approach)

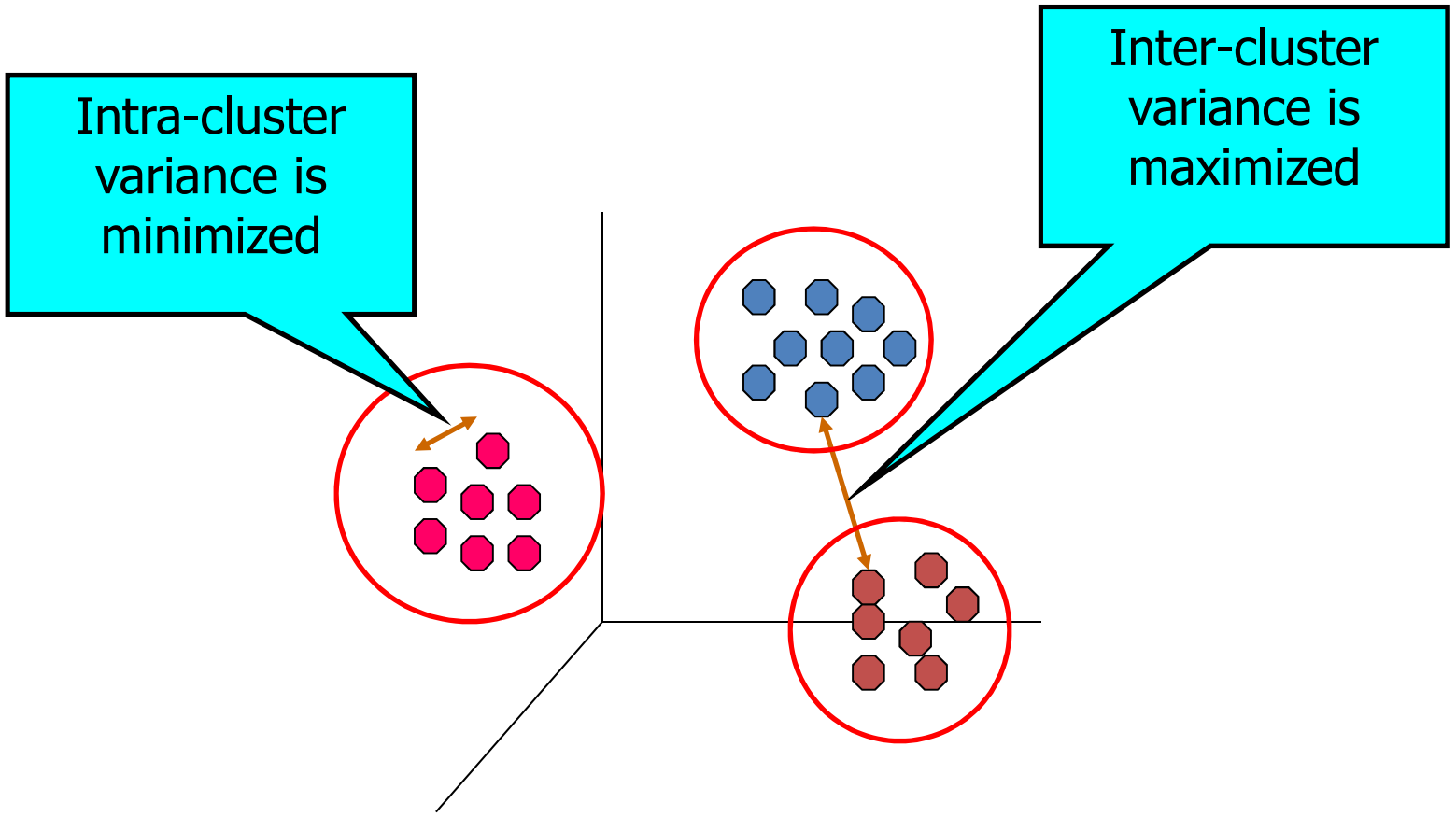
First derivative of powered MSE values:

$$J(k) = MSE(k)^{-d/2} - MSE(k-1)^{-d/2}$$



From MSE to cluster validity

- Minimize within cluster variance (MSE)
- Maximize between cluster variance



Variances

Within cluster: $$SSW(C,k) = \sum_{i=1}^{N} \| x_i - c_{p(i)} \|^2$$

Between clusters: $$SSB(C,k) = \sum_{j=1}^{k} n_j \| c_j - \bar{x} \|^2$$

Total Variance of data set:

$$\sigma(X) = \sum_{i=1}^{N} \| x_i - c_{p(i)} \|^2 + \sum_{j=1}^{k} n_j \| c_j - \bar{x} \|^2$$

SSW SSB

F-ratio variance test

- Variance-ratio F-test
- Measures ratio of between-groups variance against the within-groups variance (original f-test)
- F-ratio (WB-index):

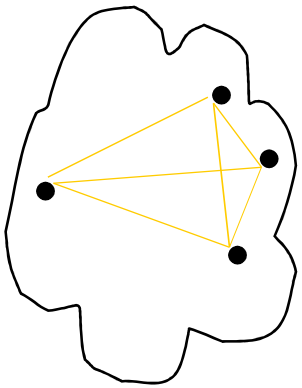
$$F = \frac{k \cdot \sum_{i=1}^{N} \| x_i - c_{p(i)} \|^2}{\sum_{j=1}^{k} n_j \| c_j - \overline{x} \|^2} = \frac{k \cdot SSW}{\sigma(X) - SSW}$$

SSB

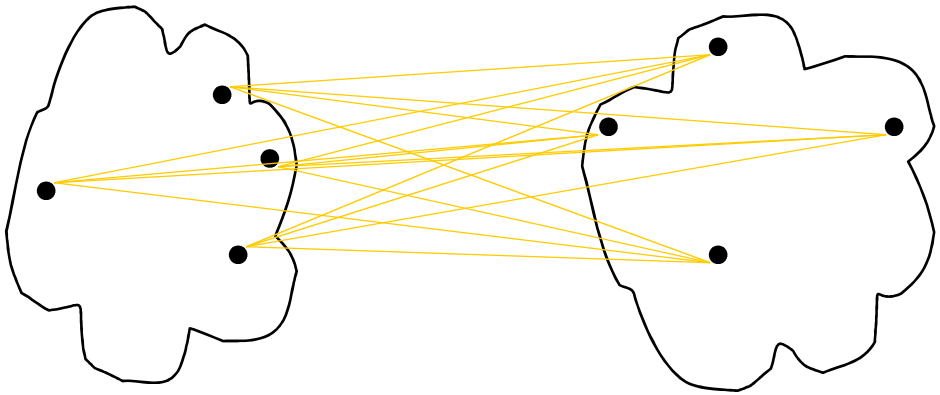
Silhouette coefficient

[Kaufman&Rousseeuw, 1990]

- Cohesion: measures how closely related are objects in a cluster
- Separation: measure how distinct or well-separated a cluster is from other clusters



cohesion



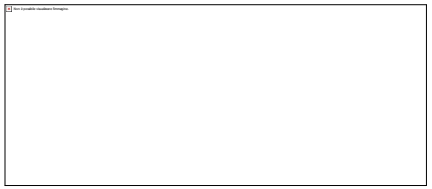
separation

Silhouette coefficient

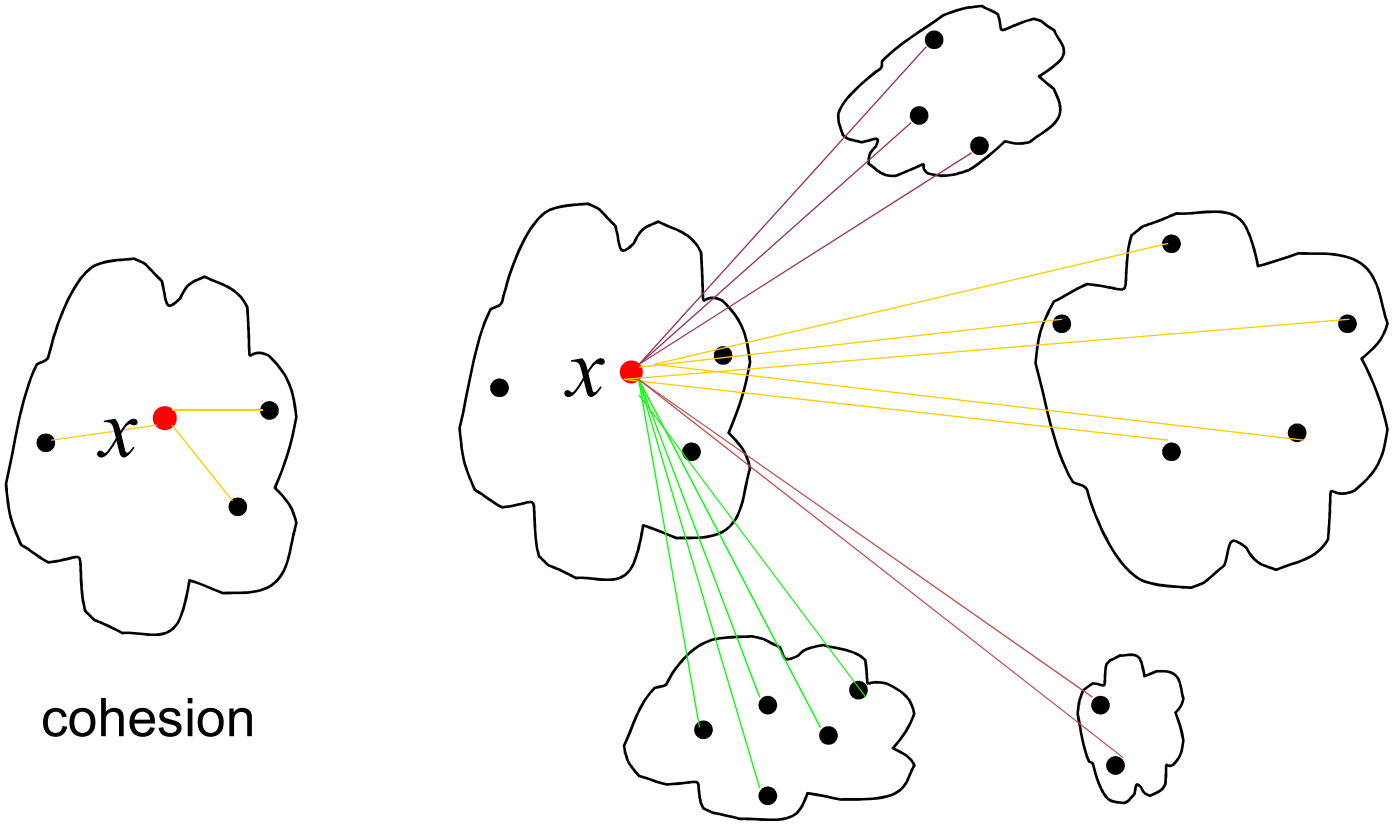
- *Cohesion* $a(x)$: average distance of x to all other vectors in the same cluster.
- *Separation* $b(x)$: average distance of x to the vectors in other clusters. Find the minimum among the clusters.
- *silhouette* $s(x)$:

$$s(x) = \frac{b(x) - a(x)}{\max\{a(x), b(x)\}}$$

- $s(x) = [-1, +1]$: -1=bad, 0=indifferent, 1=good
- Silhouette coefficient (SC):



Silhouette coefficient



$a(x)$: average distance in the cluster

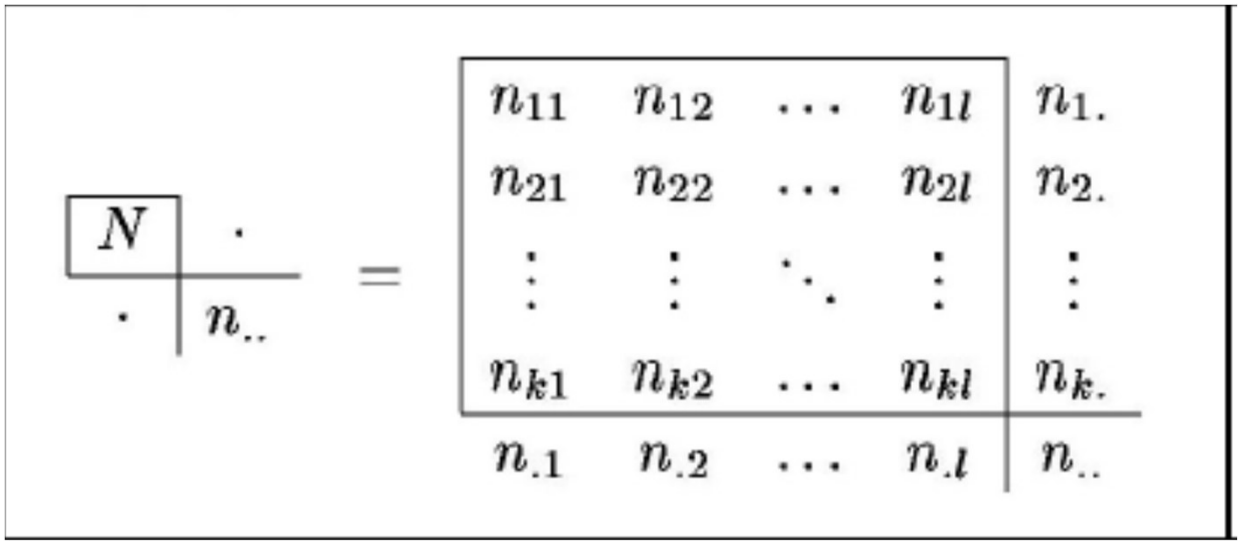
b(x): average distances to others clusters, find minimal

External indexes

- Pair counting
- Information theoretic
- Set matching

External indexes

If true class labels (*ground truth*) are known, the validity of a clustering can be verified by comparing the class labels and clustering labels.



n_{ij} = number of objects in class i and cluster j

Pair-counting measures

Measure the number of pairs that are in:

Same class **both** in *P* and *G*.

$$a = \frac{1}{2}\sum_{i=1}^{K}\sum_{j=1}^{K'} n_{ij}(n_{ij}-1)$$

Same class in *P* but different in *G*.

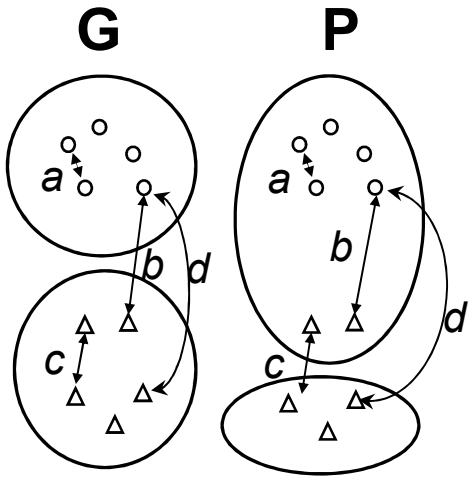
$$b = \frac{1}{2}\left(\sum_{j=1}^{K'} n_{.j}^2 - \sum_{i=1}^{K}\sum_{j=1}^{K'} n_{ij}^2\right)$$

Different classes in *P* but same in *G*.

$$c = \frac{1}{2}\left(\sum_{i=1}^{K} n_{i.}^2 - \sum_{i=1}^{K}\sum_{j=1}^{K'} n_{ij}^2\right)$$

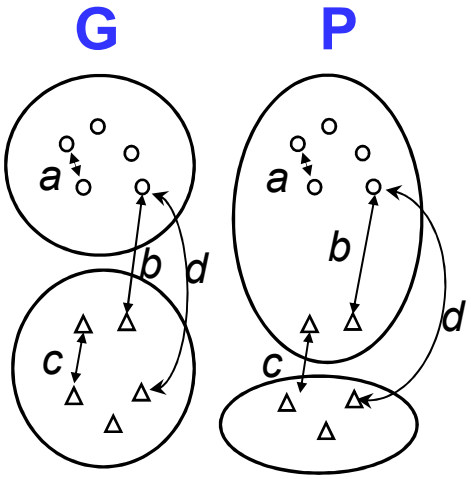
Different classes **both** in *P* and *G*.

$$d = \frac{1}{2}\left(N^2 + \sum_{i=1}^{K}\sum_{j=1}^{K'} n_{ij}^2 - \left(\sum_{i=1}^{K} n_{i.}^2 + \sum_{j=1}^{K'} n_{.j}^2\right)\right)$$



Rand and Adjusted Rand index

[Rand, 1971] [Hubert and Arabie, 1985]



Agreement: a, d

Disagreement: b, c

$$RI(P,G) = \frac{a+d}{a+b+c+d}$$

$$ARI = \frac{RI - E(RI)}{1 - E(RI)} = \frac{\sum_{i,j}\binom{n_{ij}}{2} - \left[\sum_i \binom{n_{i\bullet}}{2}\sum_j \binom{n_{\bullet j}}{2}\right] \Big/ \binom{n}{2}}{\frac{1}{2}\left[\sum_i \binom{n_{i\bullet}}{2} + \sum_j \binom{n_{\bullet j}}{2}\right] - \left[\sum_i \binom{n_{i\bullet}}{2}\sum_j \binom{n_{\bullet j}}{2}\right] \Big/ \binom{n}{2}}$$

Rand index

(example)

Vectors assigned to:	Same cluster	Different clusters
Same cluster in ground truth	20	24
Different clusters in ground truth	20	72

Rand index = (20+72) / (20+24+20+72) = 92/136 = **0.68**

Adjusted Rand = (to be calculated) = **0.xx**

Normalized Mutual information

$$MI(k, G) = \sum_{l=1}^{k} \sum_{j=1}^{G} p(l, j) \log \frac{p(l, j)}{p(l)p(j)}$$

However, it does not take into account our intuitive preference for few clusters, so we normalized it:

$$NMI(k, G) = \frac{2MI}{H(k) + H(G)}$$

$$H(k) = -\sum_{l=1}^{k} p(l) \log[p(l)]$$

Spectral Clustering

Modern clustering algorithms (III)

Spectral clustering

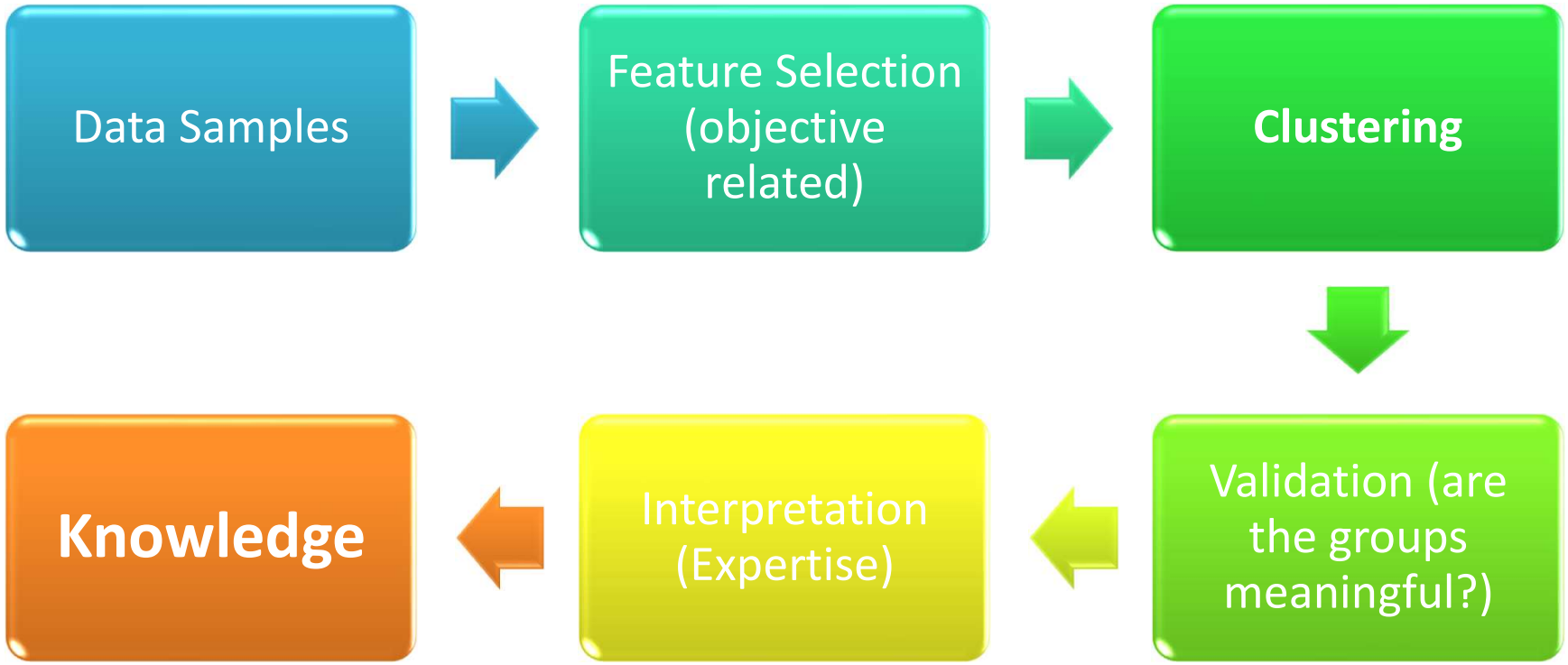
- Is a clustering method based in connectivity (instead of compactness).

HCS clustering

CLIQUE clustering

FC clustering

Clustering procedure



Objectives

- The students should be able to implement a clustering algorithm method and to choose the one that fits better the problem that they want to solve.